Puneetraj Makhija

B22 -104

R Programming Mini Project Output

Code Output:-

****

Data Cleaning / Data Wrangling

install.packages("tidyverse")

```
https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/Puneetraj Makhija/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/tidyverse_1.3.0.zip'
Content type 'application/zip' length 440148 bytes (429 KB)
downloaded 429 KB
```

library(tidyverse)

```
> library(tidyverse)
-- Attaching packages --------------------------------------- tidyverse 1.3.0 --
v ggplot2 3.3.0     v purrr   0.3.3
v tibble  2.1.3     v dplyr   0.8.5
v tidyr   1.0.2     v stringr 1.4.0
v readr   1.3.1     v forcats 0.5.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

df<- read.csv("C:/Users/Puneetraj Makhija/Desktop/R Programming/MiniProject/104_ICCWorldCup.csv", stringsAsFactors = FALSE)

str(df)

```
> df<- read.csv("C:/Users/Puneetraj Makhija/Desktop/R Programming/MiniProject/
                104_ICCWorldCup.csv", stringsAsFactors = FALSE)
> str(df)
'data.frame':  2587 obs. of  16 variables:
 $ Player                    : chr  "GS Chappell " "RW Marsh " "G Boycott "  ...
 $ Country                   : chr  "AUS" "AUS" "ENG" "ENG" ...
 $ Starting_Year             : int  1971 1971 1971 1971 1971 1971 1971 1971 1971 1971 ...
 $ Ending_Year               : int  1983 1984 1981 1982 1980 1981 1974 1975 1977 1972 ...
 $ Span_in_Years             : int  12 13 10 11 9 10 3 4 6 1 ...
 $ Matches_Played            : int  74 92 36 24 16 28 6 7 20 3 ...
 $ Innings_Batted            : chr  "72" "76" "34" "22" ...
 $ Not_Outs                  : chr  "14" "15" "4" "3" ...
 $ Runs_Scored               : chr  "2331" "1225" "1082" "757" ...
 $ Highest_Innings_Score     : chr  "138*" "66" "105" "131" ...
 $ Batting_Average           : chr  "40.18" "20.08" "36.06" "39.84" ...
 $ Balls_Faced               : chr  "3079" "1489" "2020" "1134" ...
 $ Batting_Strike_Rate       : chr  "75.7" "82.26" "53.56" "66.75" ...
 $ Hundreds_Scored           : chr  "3" "0" "1" "1" ...
 $ Runs_Scored_above_50_or._50: chr  "14" "4" "9" "5" ...
 $ Ducks_Scored              : chr  "7" "7" "1" "0" ...
```

```
#Cleaning data by converting character to Numeric in Not_Outs
df$Innings_Batted = as.numeric(substring(df$Innings_Batted,1))

#Cleaning data by converting character to Numeric in Runs_Scored
df$Runs_Scored = as.numeric(substring(df$Runs_Scored,1))

#Cleaning data by converting character to Numeric in Highest_Innings_Score
df$Highest_Innings_Score  = as.numeric(substring(df$Highest_Innings_Score ,1))

#Cleaning data by converting character to Numeric in Batting_Average
df$ Batting_Average  = as.numeric(substring(df$Batting_Average ,1))

#Cleaning data by converting character to Numeric in Balls_Faced
df$ Balls_Faced  = as.numeric(substring(df$Balls_Faced ,1))

#Cleaning data by converting character to Numeric in Batting_Strike_Rate
df$ Batting_Strike_Rate  = as.numeric(substring(df$Batting_Strike_Rate ,1))

#Cleaning data by converting character to Numeric in Hundreds_Scored
df$ Hundreds_Scored  = as.numeric(substring(df$Hundreds_Scored ,1))

#Cleaning data by converting character to Numeric in Runs_Scored_above_50_or._50
df$ Runs_Scored_above_50_or._50  = as.numeric(substring(df$
Runs_Scored_above_50_or._50 ,1))

#Cleaning data by converting character to integer in Ducks_Scored
df$ Ducks_Scored  = as.numeric(substring(df$Ducks_Scored ,1))


> #Cleaning data by converting character to Numeric in Not_Outs
> df$Not_Outs = as.numeric(substring(df$Not_Outs,1))
Warning message:
```

```
NAs introduced by coercion
> #Cleaning data by converting character to Numeric in Highest_Innings_Score
> df$Highest_Innings_Score  = as.numeric(substring(df$Highest_Innings_Score ,1
))
Warning message:
NAs introduced by coercion
> #Cleaning data by converting character to Numeric in Not_Outs
> df$Not_Outs = as.numeric(substring(df$Not_Outs,1))
>
> #Cleaning data by converting character to Numeric in Runs_Scored
> df$Runs_Scored = as.numeric(substring(df$Runs_Scored,1))
Warning message:
NAs introduced by coercion
>
> #Cleaning data by converting character to Numeric in Highest_Innings_Score
> df$Highest_Innings_Score  = as.numeric(substring(df$Highest_Innings_Score ,1
))
>
> #Cleaning data by converting character to Numeric in Batting_Average
> df$ Batting_Average  = as.numeric(substring(df$Batting_Average ,1))
Warning message:
NAs introduced by coercion
>
> #Cleaning data by converting character to Numeric in Balls_Faced
> df$ Balls_Faced  = as.numeric(substring(df$Balls_Faced ,1))
Warning message:
NAs introduced by coercion
>
> #Cleaning data by converting character to Numeric in Batting_Strike_Rate
> df$ Batting_Strike_Rate  = as.numeric(substring(df$Batting_Strike_Rate ,1))
Warning message:
NAs introduced by coercion
>
> #Cleaning data by converting character to Numeric in Hundreds_Scored
> df$ Hundreds_Scored  = as.numeric(substring(df$Hundreds_Scored ,1))
Warning message:
NAs introduced by coercion
>
> #Cleaning data by converting character to Numeric in Runs_Scored_above_50_or
._50
> df$ Runs_Scored_above_50_or._50  = as.numeric(substring(df$ Runs_Scored_abov
e_50_or._50 ,1))
Warning message:
NAs introduced by coercion
>
> #Cleaning data by converting character to integer in Ducks_Scored
> df$ Ducks_Scored  = as.numeric(substring(df$Ducks_Scored ,1))
Warning message:
NAs introduced by coercion
```

#check if any missing values in Starting_Year column of df

```
is.na(df$Starting_Year)
```

#how many missing values

```
sum(is.na(df$Starting_Year))
```

```
> #how many  missing values
> sum(is.na(df$Starting_Year))
[1] 0
```

#summarise function for distinct values

```
df %>% summarise(n= n_distinct(Starting_Year))
```

```
> #summarise function for distinct values
> df %>% summarise(n= n_distinct(Starting_Year))
   n
1 50
```

#To Check the no. of missing values in Ending_Year

```
df %>% summarise(count = sum(is.na(Ending_Year)))
```

```
> #To Check the no. of missing values in Ending_Year
> df %>% summarise(count = sum(is.na(Ending_Year)))
  count
1     0
```

#To Check the no. of missing values in Span_in_Years

```
df %>% summarise(count = sum(is.na(Span_in_Years)))
```

```
> #To Check the no. of missing values in Span_in_Years
> df %>% summarise(count = sum(is.na(Span_in_Years)))
  count
1     0
```

#To Check the no. of missing values in Matches_Played

```
df %>% summarise(count = sum(is.na(Matches_Played)))
```

```
> #To Check the no. of missing values in Matches_Played
> df %>% summarise(count = sum(is.na(Matches_Played)))
  count
1     0
```

#To Check the no. of missing values in Innings_Batted

```
df %>% summarise(count = sum(is.na(Innings_Batted)))
```

#There are 93 missing/null values in Innings_Batted

#Therefore we replace the missing values with mean

```
df <- df %>% mutate(Innings_Batted
=replace(Innings_Batted,is.na(Innings_Batted),mean(Innings_Batted,na.rm = TRUE)))
```

#To Check the no. of missing values in Not_Outs

```
df %>% summarise(count = sum(is.na(Not_Outs)))
```

```
   #There are 93 missing/null values in Not_Outs
   #Therefore we replace the missing values with mean

   df <- df %>% mutate(Not_Outs
                       =replace(Not_Outs,is.na(Not_Outs),mean(Not_Outs,na.rm = TRUE)))
#To Check the no. of missing values in  Highest_Innings_Score
df %>% summarise(count = sum(is.na( Highest_Innings_Score)))

   #There are 882 missing/null values in  Highest_Innings_Score
   #Therefore we replace the missing values with mean

   df <- df %>% mutate( Highest_Innings_Score
                        =replace( Highest_Innings_Score,is.na( Highest_Innings_Score),mean(
Highest_Innings_Score,na.rm = TRUE)))



#To Check the no. of missing values in  Batting_Average
df %>% summarise(count = sum(is.na( Batting_Average)))

   #There are 213 missing/null values in  Batting_Average
   #Therefore we replace the missing values with mean

   df <- df %>% mutate( Batting_Average
                        =replace( Batting_Average,is.na( Batting_Average),mean(
Batting_Average,na.rm = TRUE)))

#To Check the no. of missing values in  Balls_Faced
df %>% summarise(count = sum(is.na( Balls_Faced)))

   #There are 93 missing/null values in  Balls_Faced
   #Therefore we replace the missing values with mean
```

```
   df <- df %>% mutate( Balls_Faced
                        =replace( Balls_Faced,is.na( Balls_Faced),mean( Balls_Faced,na.rm =
TRUE)))



#To Check the no. of missing values in  Balls_Faced
df %>% summarise(count = sum(is.na( Balls_Faced)))

   #There are 93 missing/null values in  Balls_Faced
   #Therefore we replace the missing values with mean

   df <- df %>% mutate( Balls_Faced
                        =replace( Balls_Faced,is.na( Balls_Faced),mean( Balls_Faced,na.rm =
TRUE)))



#To Check the no. of missing values in  Batting_Strike_Rate
df %>% summarise(count = sum(is.na( Batting_Strike_Rate)))

   #There are 102 missing/null values in  Batting_Strike_Rate
   #Therefore we replace the missing values with mean

   df <- df %>% mutate( Batting_Strike_Rate
                        =replace( Batting_Strike_Rate,is.na( Batting_Strike_Rate),mean(
Batting_Strike_Rate,na.rm = TRUE)))



#To Check the no. of missing values in  Hundreds_Scored
df %>% summarise(count = sum(is.na( Hundreds_Scored)))
```

```
   #There are 93 missing/null values in  Hundreds_Scored
   #Therefore we replace the missing values with mean



 df <- df %>% mutate( Hundreds_Scored
                      =replace( Hundreds_Scored,is.na( Hundreds_Scored),mean(
Hundreds_Scored,na.rm = TRUE)))
   getwd()
#To Check the no. of missing values in  Runs_Scored_above_50_or._50
df %>% summarise(count = sum(is.na( Runs_Scored_above_50_or._50)))

   #There are 93 missing/null values in  Runs_Scored_above_50_or._50
   #Therefore we replace the missing values with mean

   df <- df %>% mutate( Runs_Scored_above_50_or._50
                      =replace( Runs_Scored_above_50_or._50,is.na(
Runs_Scored_above_50_or._50),mean( Runs_Scored_above_50_or._50,na.rm = TRUE)))



#To Check the no. of missing values in  Ducks_Scored
df %>% summarise(count = sum(is.na( Ducks_Scored )))

   #There are 93 missing/null values in  Ducks_Scored
   #Therefore we replace the missing values with mean

   df <- df %>% mutate( Ducks_Scored
                      =replace( Ducks_Scored ,is.na( Ducks_Scored ),mean( Ducks_Scored ,na.rm
= TRUE)))
**********************************************************************
```

```
#EDA
#set the wkd dir
setwd("C:/Users/Puneetraj Makhija/Desktop/R Programming/MiniProject")
getwd



# PART 1 - Summary Analysis
# An approach to unearth, summarize
# and visualize the important characteristics of a dataset.
# Important properties to look at:
# - Dimensions and size of dataset
# - Structure and variables
# - Types of variables
# - Frequencies and Mode
# - Percentiles
# - Measures of location/central tendency: Mean, Median
# - Measures of spread: Range, Variation
# - Measures of shape: Skewness, Kurtosis
# - etc


df
class(df)
> class(df)
[1] "data.frame"


dim(df)
> dim(df)
[1] 2587   16
```

names(df) #or

colnames(df)

```
> colnames(df)
 [1] "Player"                    "Country"             "Starting_Year"
 [4] "Ending_Year"               "Span_in_Years"       "Matches_Played"
 [7] "Innings_Batted"            "Not_Outs"            "Runs_Scored"
[10] "Highest_Innings_Score"     "Batting_Average"     "Balls_Faced"
[13] "Batting_Strike_Rate"       "Hundreds_Scored"     "Runs_Scored_above_50_
or._50"
[16] "Ducks_Scored"
```

object.size(df)

```
> object.size(df)
473096 bytes
```

head(df)

```
> head(df)
          Player Country Starting_Year Ending_Year Span_in_Years Matches_Played Innings
_Batted Not_Outs
1  GS Chappell      AUS          1971        1983            12             74
72        14
2     RW Marsh      AUS          1971        1984            13             92
76        15
3    G Boycott      ENG          1971        1981            10             36
34         4
4 KWR Fletcher      ENG          1971        1982            11             24
22         3
5  IM Chappell      AUS          1971        1980             9             16
16         2
6   KD Walters      AUS          1971        1981            10             28
24         6
  Runs_Scored Highest_Innings_Score Batting_Average Balls_Faced Batting_Strike_Rate Hu
ndreds_Scored
1        2331                    NA           40.18        3079                75.70
3
2        1225                    66           20.08        1489                82.26
0
3        1082                   105           36.06        2020                53.56
1
4         757                   131           39.84        1134                66.75
1
5         673                    86           48.07         874                77.00
0
6         513                    59           28.50         732                70.08
0
  Runs_Scored_above_50_or._50 Ducks_Scored
1                          14            7
2                           4            7
3                           9            1
4                           5            0
5                           8            0
6                           2            1
```

summary(df)

```
> summary(df)
    Player           Country          Starting_Year   Ending_Year   Span_in_Years    Matches_Played
 Length:2587        Length:2587        Min.   :1971   Min.   :1971   Min.   : 0.000   Min.   :  1.00
 Class :character   Class :character   1st Qu.:1991   1st Qu.:1996   1st Qu.: 0.000   1st Qu.:  4.00
 Mode  :character   Mode  :character   Median :2003   Median :2007   Median : 3.000   Median : 12.00
                                       Mean   :2000   Mean   :2004   Mean   : 4.124   Mean   : 36.05
                                       3rd Qu.:2010   3rd Qu.:2016   3rd Qu.: 7.000   3rd Qu.: 41.00
                                       Max.   :2020   Max.   :2020   Max.   :23.000   Max.   :463.00
 Innings_Batted       Not_Outs      Runs_Scored      Highest_Innings_Score Batting_Average
 Length:2587        Min.   : 0.0   Min.   :    0.0   Min.   :  0.00        Min.   :  0.00
 Class :character   1st Qu.: 0.0   1st Qu.:   24.0   1st Qu.: 24.00        1st Qu.:  9.09
```

```
 Mode  :character   Median : 2.0   Median :  114.0   Median : 47.79        Median : 17.75
                    Mean   : 5.5   Mean   :  676.8   Mean   : 47.79        Mean   : 18.29
                    3rd Qu.: 6.0   3rd Qu.:  582.0   3rd Qu.: 51.00        3rd Qu.: 24.63
                    Max.   :84.0   Max.   :18426.0   Max.   :264.00        Max.   :145.00
  Balls_Faced      Batting_Strike_Rate Hundreds_Scored   Runs_Scored_above_50_or._50  Ducks_Scored
 Min.   :    0.0   Min.   :  0.00      Min.   : 0.0000   Min.   : 0.000              Min.   : 0.000
 1st Qu.:   45.0   1st Qu.: 50.00      1st Qu.: 0.0000   1st Qu.: 0.000              1st Qu.: 0.000
 Median :  181.0   Median : 63.55      Median : 0.0000   Median : 0.000              Median : 1.000
 Mean   :  905.8   Mean   : 63.55      Mean   : 0.7269   Mean   : 3.571              Mean   : 2.474
 3rd Qu.:  800.5   3rd Qu.: 77.06      3rd Qu.: 0.0000   3rd Qu.: 2.000              3rd Qu.: 3.000
 Max.   :21367.0   Max.   :328.57      Max.   :49.0000   Max.   :96.000              Max.   :34.000
```

str

```
> str
function (object, ...)
UseMethod("str")
<bytecode: 0x000001f9021fb650>
<environment: namespace:utils>
```

#Starting Year

#returns mean, missing values are removed, if #na.rm=TRUE.

mean(df$Starting_Year, na.rm=TRUE)

```
> mean(df$Starting_Year, na.rm=TRUE)
[1] 2000.043
```

median(df$Starting_Year, na.rm=TRUE)

```
> median(df$Starting_Year, na.rm=TRUE)
[1] 2003
```

range(df$Starting_Year,na.rm=TRUE)

```
> range(df$Starting_Year,na.rm=TRUE)
[1] 1971 2020
```

var(df$Starting_Year,na.rm=TRUE)

```
> var(df$Starting_Year,na.rm=TRUE)
[1] 176.5144
```

sd(df$Starting_Year, na.rm=TRUE)

```
> sd(df$Starting_Year, na.rm=TRUE)
[1] 13.28587
```

quantile(df$Starting_Year, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Starting_Year, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
1971 1991 2003 2010 2020
```

fivenum(df$Starting_Year)

```
> fivenum(df$Starting_Year)
[1] 1971 1991 2003 2010 2020
```

#Ending Year

mean(df$Ending_Year, na.rm=TRUE)

```
> mean(df$Ending_Year, na.rm=TRUE)
[1] 2004.167
```

median(df$Ending_Year, na.rm=TRUE)

```
> median(df$Ending_Year, na.rm=TRUE)
[1] 2007
```

range(df$Ending_Year,na.rm=TRUE)

```
> range(df$Ending_Year,na.rm=TRUE)
[1] 1971 2020
```

var(df$Ending_Year,na.rm=TRUE)

```
> var(df$Ending_Year,na.rm=TRUE)
[1] 170.4439
```

sd(df$Ending_Year, na.rm=TRUE)

```
> sd(df$Ending_Year, na.rm=TRUE)
[1] 13.05542
```

quantile(df$Ending_Year, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Ending_Year, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
1971 1996 2007 2016 2020
```

fivenum(df$Ending_Year)

```
> fivenum(df$Ending_Year)
[1] 1971 1996 2007 2016 2020
```

#Span in Years

mean(df$Span_in_Years, na.rm=TRUE)

```
> mean(df$Span_in_Years, na.rm=TRUE)
[1] 4.124082
```

median(df$Span_in_Years, na.rm=TRUE)

```
> median(df$Span_in_Years, na.rm=TRUE)
[1] 3
```

range(df$Span_in_Years,na.rm=TRUE)

```
> range(df$Span_in_Years,na.rm=TRUE)
[1]  0 23
```

var(df$Span_in_Years,na.rm=TRUE)

```
> var(df$Span_in_Years,na.rm=TRUE)
[1] 19.54492
```

sd(df$Span_in_Years, na.rm=TRUE)

```
> sd(df$Span_in_Years, na.rm=TRUE)
[1] 4.420964
```

quantile(df$Span_in_Years, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Span_in_Years, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
   0    0    3    7   23
```

fivenum(df$Span_in_Years)

```
> fivenum(df$Span_in_Years)
[1]  0  0  3  7 23
```

#Matches Played

mean(df$Matches_Played, na.rm=TRUE)

```
> mean(df$Matches_Played, na.rm=TRUE)
[1] 36.04639
```

median(df$Matches_Played, na.rm=TRUE)

```
> median(df$Matches_Played, na.rm=TRUE)
[1] 12
```

range(df$Matches_Played,na.rm=TRUE)

```
> range(df$Matches_Played,na.rm=TRUE)
[1]   1 463
```

var(df$Matches_Played,na.rm=TRUE)

```
> var(df$Matches_Played,na.rm=TRUE)
[1] 3392.76
```

sd(df$Matches_Played, na.rm=TRUE)

```
> sd(df$Matches_Played, na.rm=TRUE)
[1] 58.2474
```

quantile(df$Matches_Played, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Matches_Played, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
   1    4   12   41  463
```

fivenum(df$Matches_Played)

```
> fivenum(df$Matches_Played)
[1]   1   4  12  41 463
```

#Not Outs

mean(df$Not_Outs, na.rm=TRUE)

```
> mean(df$Not_Outs, na.rm=TRUE)
[1] 5.499599
```

median(df$Not_Outs, na.rm=TRUE)

```
> median(df$Not_Outs, na.rm=TRUE)
[1] 2
```

range(df$Not_Outs,na.rm=TRUE)

```
> range(df$Not_Outs,na.rm=TRUE)
[1]  0 84
```

var(df$Not_Outs,na.rm=TRUE)

```
> var(df$Not_Outs,na.rm=TRUE)
[1] 84.42904
```

sd(df$Not_Outs, na.rm=TRUE)

```
> sd(df$Not_Outs, na.rm=TRUE)
[1] 9.188528
```

quantile(df$Not_Outs, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Not_Outs, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
   0    0    2    6   84
```

fivenum(df$Not_Outs)

```
> fivenum(df$Not_Outs)
[1]  0  0  2  6 8
```

#Runs Scored

mean(df$Runs_Scored, na.rm=TRUE)

```
> mean(df$Runs_Scored, na.rm=TRUE)
[1] 676.7823
```

median(df$Runs_Scored, na.rm=TRUE)

```
> median(df$Runs_Scored, na.rm=TRUE)
[1] 114
```

range(df$Runs_Scored,na.rm=TRUE)

```
> range(df$Runs_Scored,na.rm=TRUE)
[1]     0 18426
```

var(df$Runs_Scored,na.rm=TRUE)

```
> var(df$Runs_Scored,na.rm=TRUE)
[1] 2521937
```

sd(df$Runs_Scored, na.rm=TRUE)

```
> sd(df$Runs_Scored, na.rm=TRUE)
[1] 1588.061
```

quantile(df$Runs_Scored, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Runs_Scored, probs=seq(0,1,0.25),na.rm=TRUE)
   0%   25%   50%   75%  100%
    0    24   114   582 18426
```

fivenum(df$Runs_Scored)

```
> fivenum(df$Runs_Scored)
[1]     0    24   114   582 18426
```

Highest Innings Score

mean(df$Highest_Innings_Score, na.rm=TRUE)

```
> mean(df$Highest_Innings_Score, na.rm=TRUE)
[1] 47.78592
```

median(df$Highest_Innings_Score, na.rm=TRUE)

```
> median(df$Highest_Innings_Score, na.rm=TRUE)
[1] 47.78592
```

range(df$Highest_Innings_Score,na.rm=TRUE)

```
> range(df$Highest_Innings_Score,na.rm=TRUE)
[1]   0 264
```

var(df$Highest_Innings_Score,na.rm=TRUE)

```
> var(df$Highest_Innings_Score,na.rm=TRUE)
[1] 1176.754
```

sd(df$Highest_Innings_Score, na.rm=TRUE)

```
> sd(df$Highest_Innings_Score, na.rm=TRUE)
[1] 34.30385
```

quantile(df$Highest_Innings_Score, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Highest_Innings_Score, probs=seq(0,1,0.25),na.rm=TRUE)
      0%       25%       50%       75%      100%
 0.00000  24.00000  47.78592  51.00000 264.00000
```

fivenum(df$Highest_Innings_Score)

```
> fivenum(df$Highest_Innings_Score)
[1]   0.00000  24.00000  47.78592  51.00000 264.00000
```

#Batting Average

mean(df$Batting_Average, na.rm=TRUE)

```
> mean(df$Batting_Average, na.rm=TRUE)
[1] 18.28585
```

median(df$Batting_Average, na.rm=TRUE)

```
> median(df$Batting_Average, na.rm=TRUE)
[1] 17.75
```

range(df$Batting_Average,na.rm=TRUE)

```
> range(df$Batting_Average,na.rm=TRUE)
[1]   0 145
```

var(df$Batting_Average,na.rm=TRUE)

```
> var(df$Batting_Average,na.rm=TRUE)
[1] 149.6689
```

sd(df$Batting_Average, na.rm=TRUE)

```
> sd(df$Batting_Average, na.rm=TRUE)
[1] 12.23392
```

quantile(df$Batting_Average, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Batting_Average, probs=seq(0,1,0.25),na.rm=TRUE)
    0%    25%    50%    75%   100%
  0.00   9.09  17.75  24.63 145.00
```

fivenum(df$Batting_Average)

```
> fivenum(df$Batting_Average)
[1]   0.00   9.09  17.75  24.63 145.00
```

#Balls Faced

mean(df$Balls_Faced, na.rm=TRUE)

```
> mean(df$Balls_Faced, na.rm=TRUE)
[1] 905.8027
```

median(df$Balls_Faced, na.rm=TRUE)

```
> median(df$Balls_Faced, na.rm=TRUE)
[1] 181
```

range(df$Balls_Faced,na.rm=TRUE)

```
> range(df$Balls_Faced,na.rm=TRUE)
[1]     0 21367
```

var(df$Balls_Faced,na.rm=TRUE)

```
> var(df$Balls_Faced,na.rm=TRUE)
[1] 4100631
```

sd(df$Balls_Faced, na.rm=TRUE)

```
> sd(df$Balls_Faced, na.rm=TRUE)
[1] 2025.001
```

quantile(df$Balls_Faced, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Balls_Faced, probs=seq(0,1,0.25),na.rm=TRUE)
    0%    25%    50%    75%   100%
   0.0   45.0  181.0  800.5 21367.0
```

fivenum(df$Balls_Faced)

```
> fivenum(df$Balls_Faced)
[1]     0.0    45.0   181.0   800.5 21367.0
```

Batting Strike

mean(df$Batting_Strike_Rate, na.rm=TRUE)

```
> mean(df$Batting_Strike_Rate, na.rm=TRUE)
[1] 63.55337
```

median(df$Batting_Strike_Rate, na.rm=TRUE)

```
> median(df$Batting_Strike_Rate, na.rm=TRUE)
```

```
[1] 63.55337
```

range(df$Batting_Strike_Rate,na.rm=TRUE)

```
> range(df$Batting_Strike_Rate,na.rm=TRUE)
[1]   0.00 328.57
```

var(df$Batting_Strike_Rate,na.rm=TRUE)

```
> var(df$Batting_Strike_Rate,na.rm=TRUE)
[1] 675.7612
```

sd(df$Batting_Strike_Rate, na.rm=TRUE)

```
> sd(df$Batting_Strike_Rate, na.rm=TRUE)
[1] 25.99541
```

quantile(df$Batting_Strike_Rate, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Batting_Strike_Rate, probs=seq(0,1,0.25),na.rm=TRUE)
      0%       25%       50%       75%      100%
 0.00000  50.00000  63.55337  77.06500 328.57000
```

fivenum(df$Batting_Strike_Rate)

```
> fivenum(df$Batting_Strike_Rate)
[1]   0.00000  50.00000  63.55337  77.06500 328.57000
```

Hundreds Scored

mean(df$Hundreds_Scored, na.rm=TRUE)

median(df$Hundreds_Scored, na.rm=TRUE)

```
> median(df$Hundreds_Scored, na.rm=TRUE)
[1] 0
```

range(df$Hundreds_Scored,na.rm=TRUE)

```
> range(df$Hundreds_Scored,na.rm=TRUE)
[1]  0 49
```

var(df$Hundreds_Scored,na.rm=TRUE)

```
> var(df$Hundreds_Scored,na.rm=TRUE)
[1] 8.361581
```

sd(df$Hundreds_Scored, na.rm=TRUE)

```
> sd(df$Hundreds_Scored, na.rm=TRUE)
[1] 2.89164
```

quantile(df$Hundreds_Scored, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Hundreds_Scored, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
   0    0    0    0   49
```

fivenum(df$Hundreds_Scored

```
> fivenum(df$Hundreds_Scored)
[1]  0  0  0  0 49
```

#Ducks Scored

mean(df$Ducks_Scored, na.rm=TRUE)

```
> mean(df$Ducks_Scored, na.rm=TRUE)
[1] 2.474338
```

median(df$Ducks_Scored, na.rm=TRUE)

```
> median(df$Ducks_Scored, na.rm=TRUE)
[1] 1
```

range(df$Ducks_Scored,na.rm=TRUE)

```
> range(df$Ducks_Scored,na.rm=TRUE)
[1]  0 34
```

var(df$Ducks_Scored,na.rm=TRUE)

```
> var(df$Ducks_Scored,na.rm=TRUE)
[1] 14.22732
```

sd(df$Ducks_Scored, na.rm=TRUE)

```
> sd(df$Ducks_Scored, na.rm=TRUE)
[1] 3.771912
```

quantile(df$Ducks_Scored, probs=seq(0,1,0.25),na.rm=TRUE)

```
> quantile(df$Ducks_Scored, probs=seq(0,1,0.25),na.rm=TRUE)
  0%  25%  50%  75% 100%
   0    0    1    3   34
```

fivenum(df$Ducks_Scored)

```
> fivenum(df$Ducks_Scored)
[1]  0  0  1  3 34
```

PART 2 - Descriptive Analysis

Extension of Summary Analysis.

Generally, both overlap each other.

```
# Helps in analysing large amounts of data
# in simple and structured manner.
# Involves numerical and graphical methods
# to analyse the dataset.
# Refers to measures of distribution, shape, central tendency
# and variability of a dataset with respect
# to continuous variables mainly.
# Skewness: Refers to the symmetry (or asymmetry) of a distribution.
# - Can be positive or negative.
# - Positive value: Distribution is right-skewed
# i.e. mean is greater than median.
# - Negative value: Distribution is left-skewed
# i.e. mean is less than median.
#Calculating skewness

SIY <- df$Span_in_Years
MP <- df$Matches_Played
IS<- df$Innings_Batted
NO<- df$Not_Outs
RS<- df$Runs_Scored
HIS<- df$Highest_Innings_Score
BA<- df$Batting_Average
BF<- df$Balls_Faced
BSR<- df$Batting_Strike_Rate
HS<- df$Hundreds_Scored
RSA<- df$Runs_Scored_above_50_or._50
DS<- df$Ducks_Scored

install.packages("moments")
```

library(moments)

skewness(SIY)

```
> skewness(SIY)
[1] 1.053648
```

skewness(MP)

```
> skewness(MP)
[1] 2.968275
```

skewness(IS)

```
> skewness(IS)
[1] 5.130879
```

skewness(NO)

```
> skewness(NO)
[1] 3.130879
```

skewness(RS)

```
> skewness(RS)
[1] 4.447134
```

skewness(HIS)

```
> skewness(HIS)
[1] 1.366548
```

skewness(BA)

```
> skewness(BA)
[1] 1.42563
```

skewness(BF)

```
> skewness(BF)
[1] 4.228021
```

skewness(BSR)

```
> skewness(BSR)
[1] 0.7584338
```

skewness(HS)

```
> skewness(HS)
[1] 7.472802
```

skewness(RSA)

```
> skewness(RSA)
[1] 4.566868
```

skewness(DS)

```
> skewness(DS)
[1] 2.866744
```

```
# Kurtosis: Refers to the tailedness (heavy-tailed or light-tailed)
# of data relative to a normal distribution.
# - Can be positive or negative.
#
# - Positive value: Positive or high value kurtosis
# indicates tails or outliers. Said to be leptokurtic.
#
# - Negative value: Negative kurtosis indicates a
# flat data distribution (light tails or lack of outliers).
# Said to be platykurtic.
#
# - The normal distribution has zero kurtosis
# (or Pearson's kurtosis as 3). Said to be mesokurtic.
```

```
#calculation of kurtosis
```

kurtosis(SIY) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(SIY) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 3.412724
```

kurtosis(MP) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(MP) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 13.68285
```

kurtosis(IS) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(IS) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 12.68285
```

kurtosis(NO) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(NO) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 15.42966
```

kurtosis(RS) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(RS) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 27.94169
```

kurtosis(HIS) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(HIS) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 5.828927
```

kurtosis(BA) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(BA) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 9.523326
```

kurtosis(BF) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(BF) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 25.06735
```

kurtosis(BSR) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(BSR) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 9.410634
```

kurtosis(HS) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(HS) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 79.5539
```

kurtosis(RSA) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(RSA) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 28.35597
```

kurtosis(DS) #reports Pearson's (proper) kurtosis (> or < 3)

```
> kurtosis(DS) #reports Pearson's (proper) kurtosis (> or < 3)
[1] 13.93426
```

****

#Data visualization in R

R has 3 plotting systems
#1. Base plotting system
#2. Lattice plotting system
#3. GGplot2

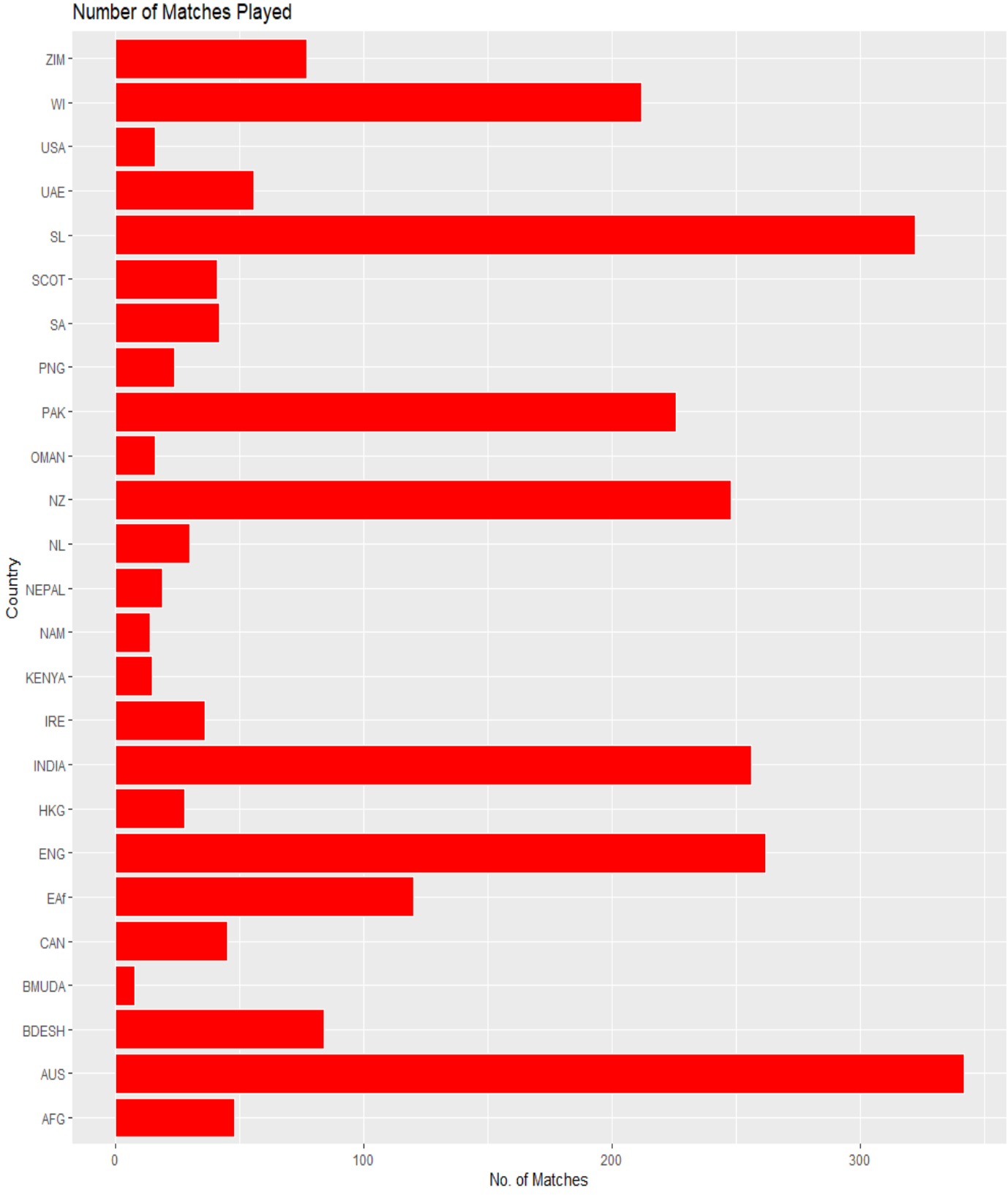
```
install.packages("ggplot2")
library(ggplot2)
install.packages("ggthemes")
library(ggthemes)
library(tidyverse)


#ggplot2
#gg stands for grammer of graphics

#components of graphics:
#1. Data : The dataset being summarized
#2. Aesthetics: Variables mapped to visual cues, such as x-axis and y-axis values and color, shape, size etc.
#3. Geometry: The type of plot (scatterplot, boxplot, barplot, histogram, qqplot, s mooth density, etc.)
#4. Facets: Groups by which we divide the data

df<- read.csv("C:/Users/Puneetraj Makhija/Desktop/R Programming/MiniProject /104_ICCWorldCup.csv", stringsAsFactors = FALSE)
str(df)
```

```
> str(df)
'data.frame': 2587 obs. of  16 variables:
 $ Player                     : chr  "GS Chappell ""RW Marsh""G Boycott "  ...
 $ Country                    : chr  "AUS" "AUS" "ENG" "ENG" ...
 $ Starting_Year              : int  1971 1971 1971 1971 1971 1971 1971 1971..
 $ Ending_Year                : int  1983 1984 1981 1982 1980 1981 1974  1977
 $ Span_in_Years              : int  12 13 10 11 9 10 3 4 6 1 ...
 $ Matches_Played             : int  74 92 36 24 16 28 6 7 20 3 ...
 $ Innings_Batted             : chr  "72" "76" "34" "22" ...
 $ Not_Outs                   : num  14 15 4 3 2 6 0 0 4 1 ...
 $ Runs_Scored                : num  2331 1225 1082 757 673 ...
 $ Highest_Innings_Score      : num  47.8 66 105 131 86 ...
 $ Batting_Average            : num  40.2 20.1 36.1 39.8 48.1 ...
 $ Balls_Faced                : num  3079 1489 2020 1134 874 ...
 $ Batting_Strike_Rate        : num  75.7 82.3 53.6 66.8 77 ...
 $ Hundreds_Scored            : num  3 0 1 1 0 0 0 0 0 0 ...
 $ Runs_Scored_above_50_or._50: num  14 4 9 5 8 2 3 2 1 0 ...
 $ Ducks_Scored               : num  7 7 1 0 0 1 0 0 2 0 ...
```

#BoxPlot

ggplot(data=df, aes(y=Country))+ geom_bar(col="white", fill="red") + labs(title ="Number of Matches Played", y = "Country", x = "No. of Matches")

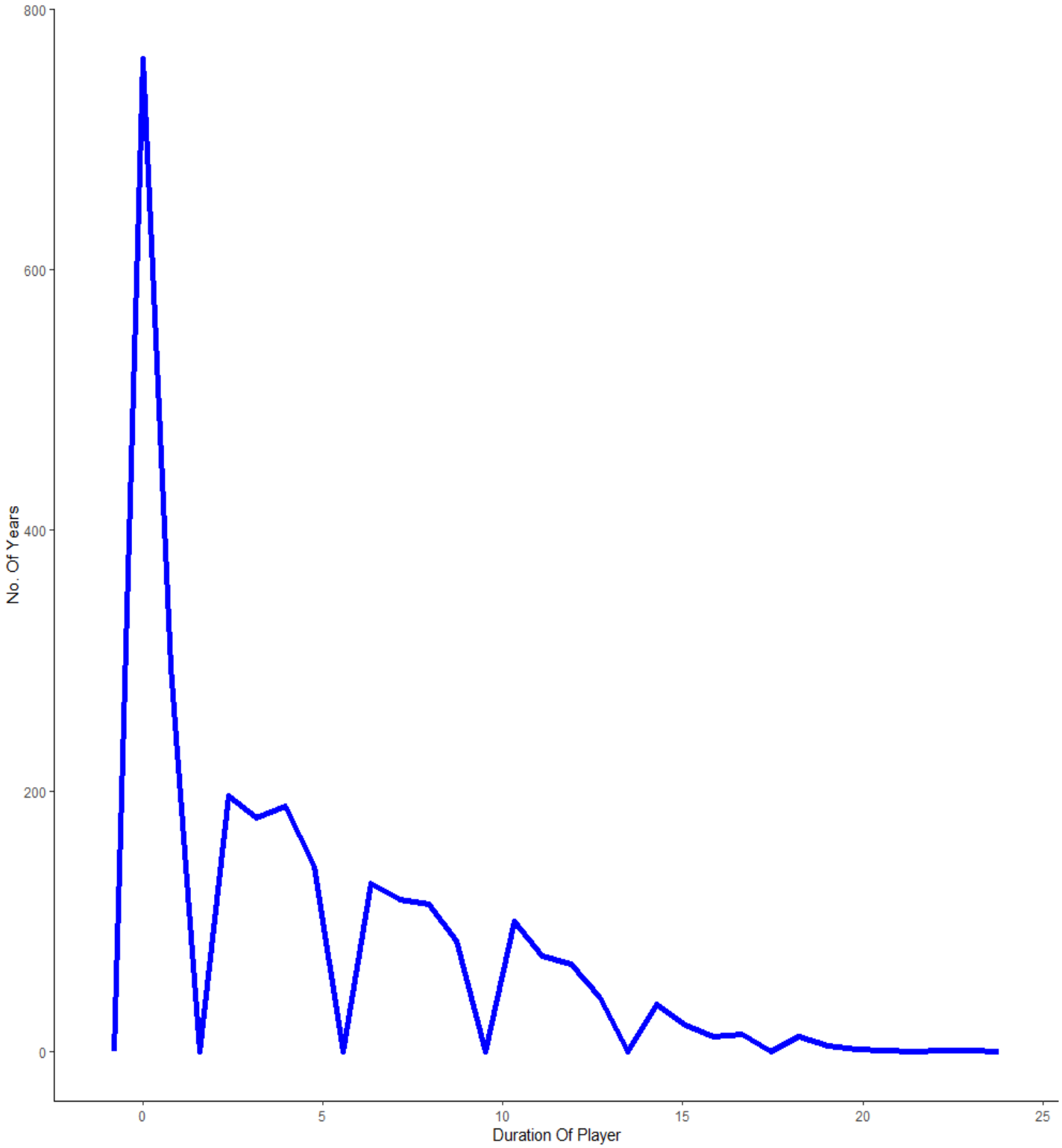
```
> ggplot(data=df, aes(y=Country))+ geom_bar(col="white", fill="red") + labs(ti
tle ="Number of Matches Played", y = "Country", x = "No. of Matches")
```



#Frequency Plot

ggplot(data=df, aes(Span_in_Years))+ geom_freqpoly(color="blue",size=2) + labs(y = "No. Of Years", x = "Duration Of Player") + theme_classic()

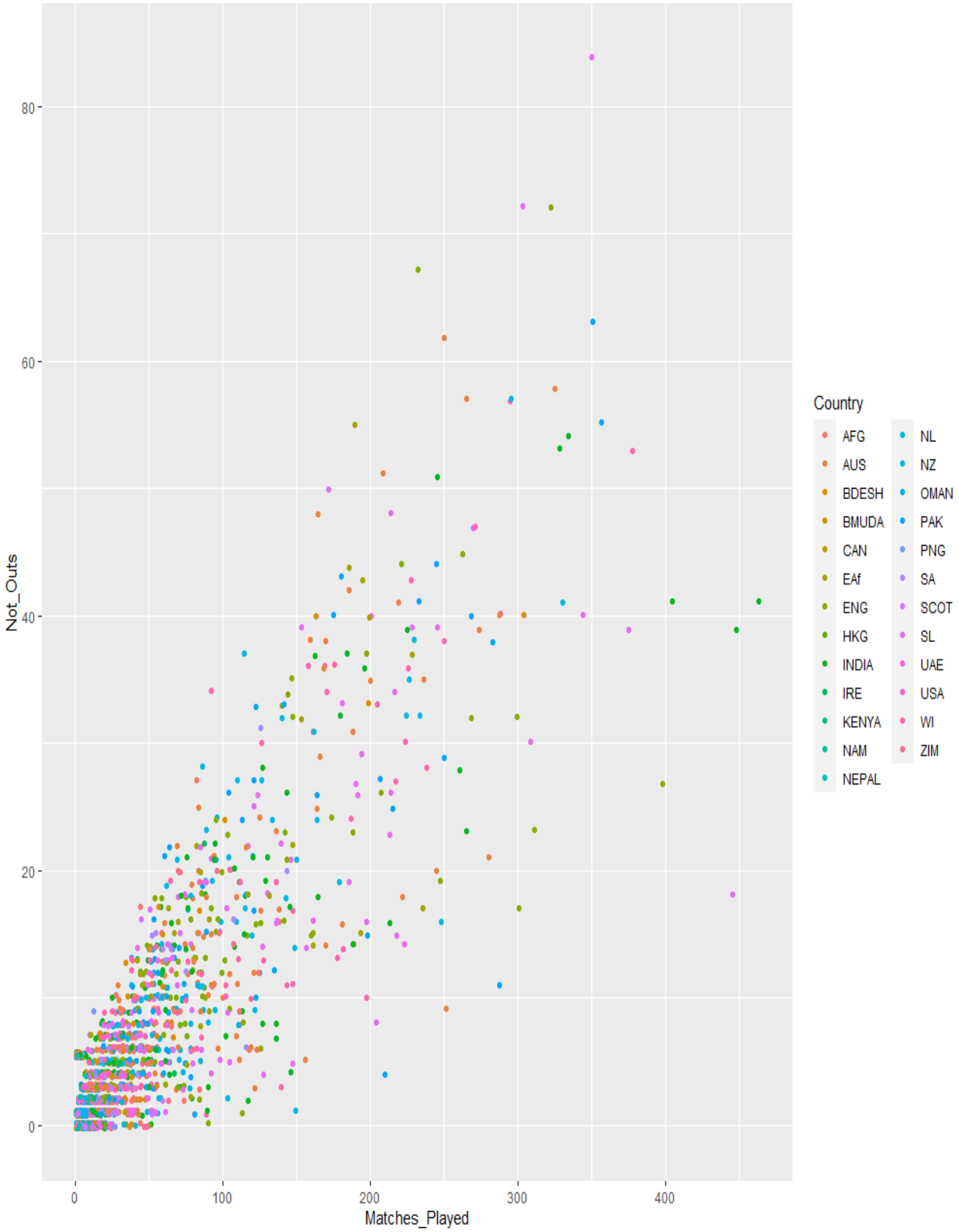
```
> ggplot(data=df, aes(Span_in_Years))+ geom_freqpoly(color="blue",size=2)  + l
abs( y = "No. Of Years", x = "Duration Of Player")  + theme_classic()
```



#Jitter

ggplot(data=df, aes(x=Matches_Played,y=Not_Outs, color= Country))+ geom_jitter()

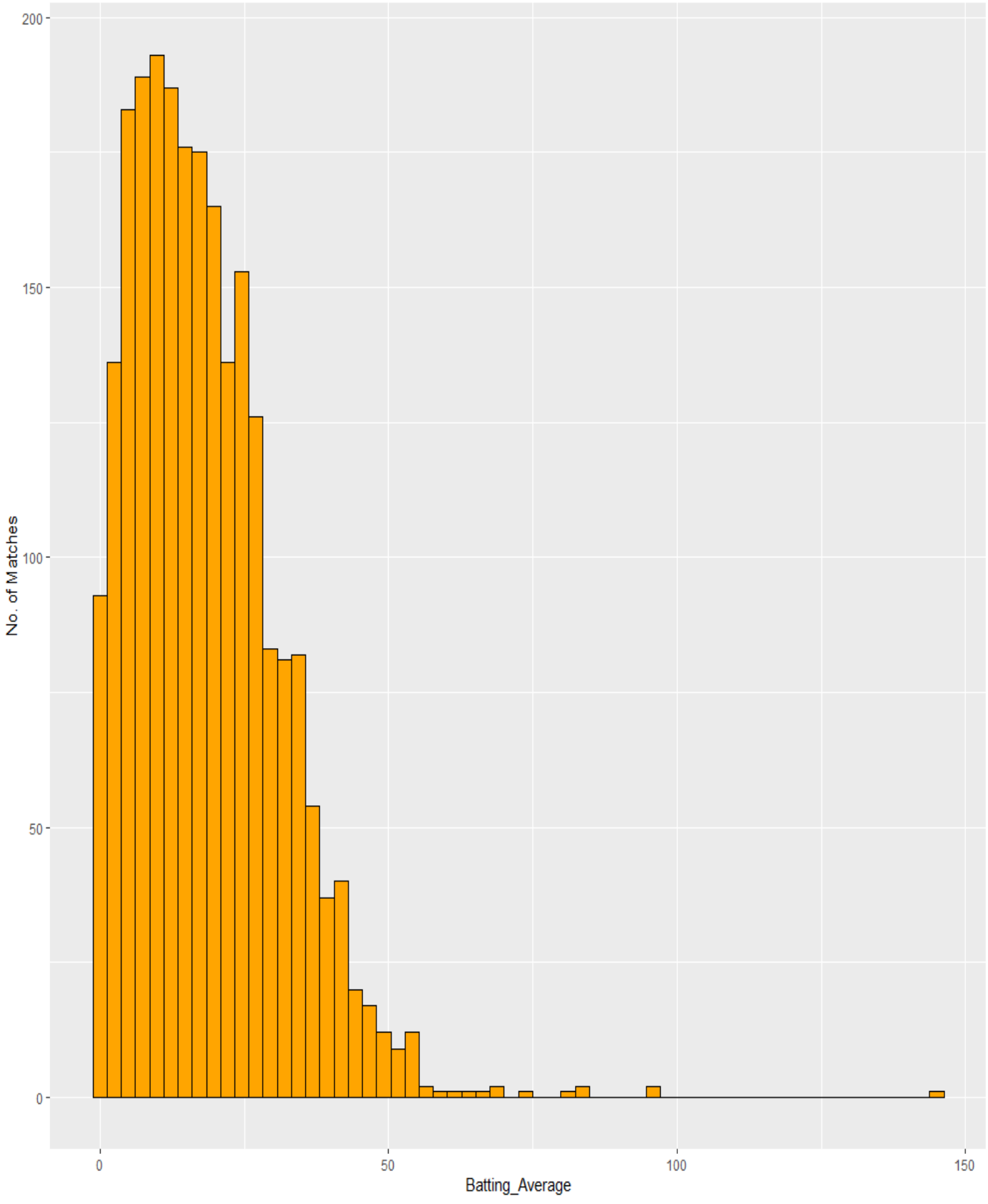
```
> ggplot(data=df, aes(x=Matches_Played,y=Not_Outs, color= Country))+ geom_jitter()
```



#Histogram

ggplot(data=df, aes(x=as.numeric(Batting_Average))) + geom_histogram(bins=60, fill="orange",color="Black") + labs(x="Batting_Average",y="No. of Matches")

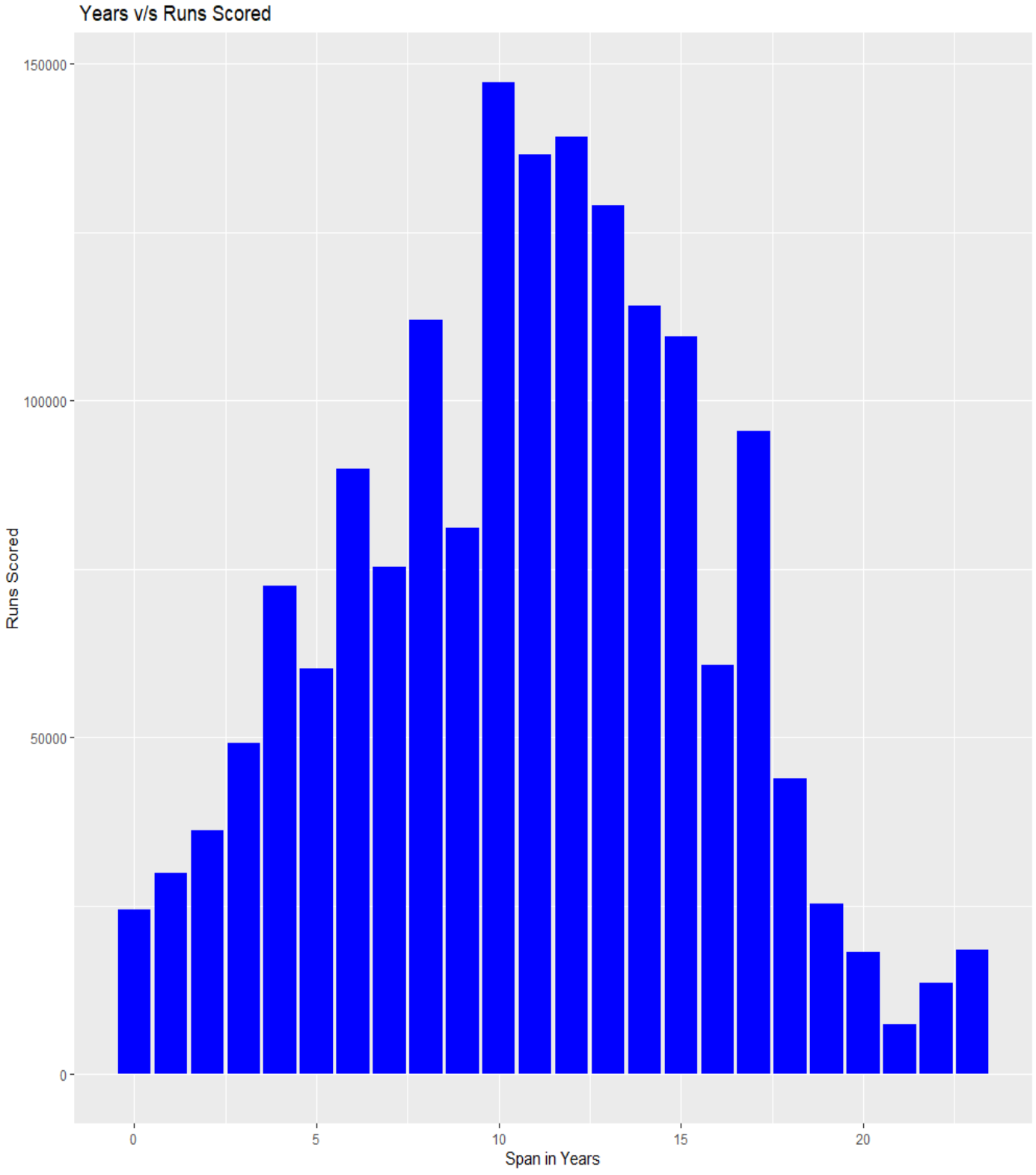
```
> ggplot(data=df, aes(x=as.numeric(Batting_Average))) + geom_histogram(bins=60
, fill="orange",color="Black") + labs(x="Batting_Average",y="No. of Matches")
```



#Barplot

ggplot(data=df, aes(x=as.numeric(Span_in_Years),y=as.numeric(Runs_Scored)))+ geom_bar(stat = "identity",fill="blue") + labs(title=" Years v/s Runs Scored",x="Span in Years",y="Runs Scored")

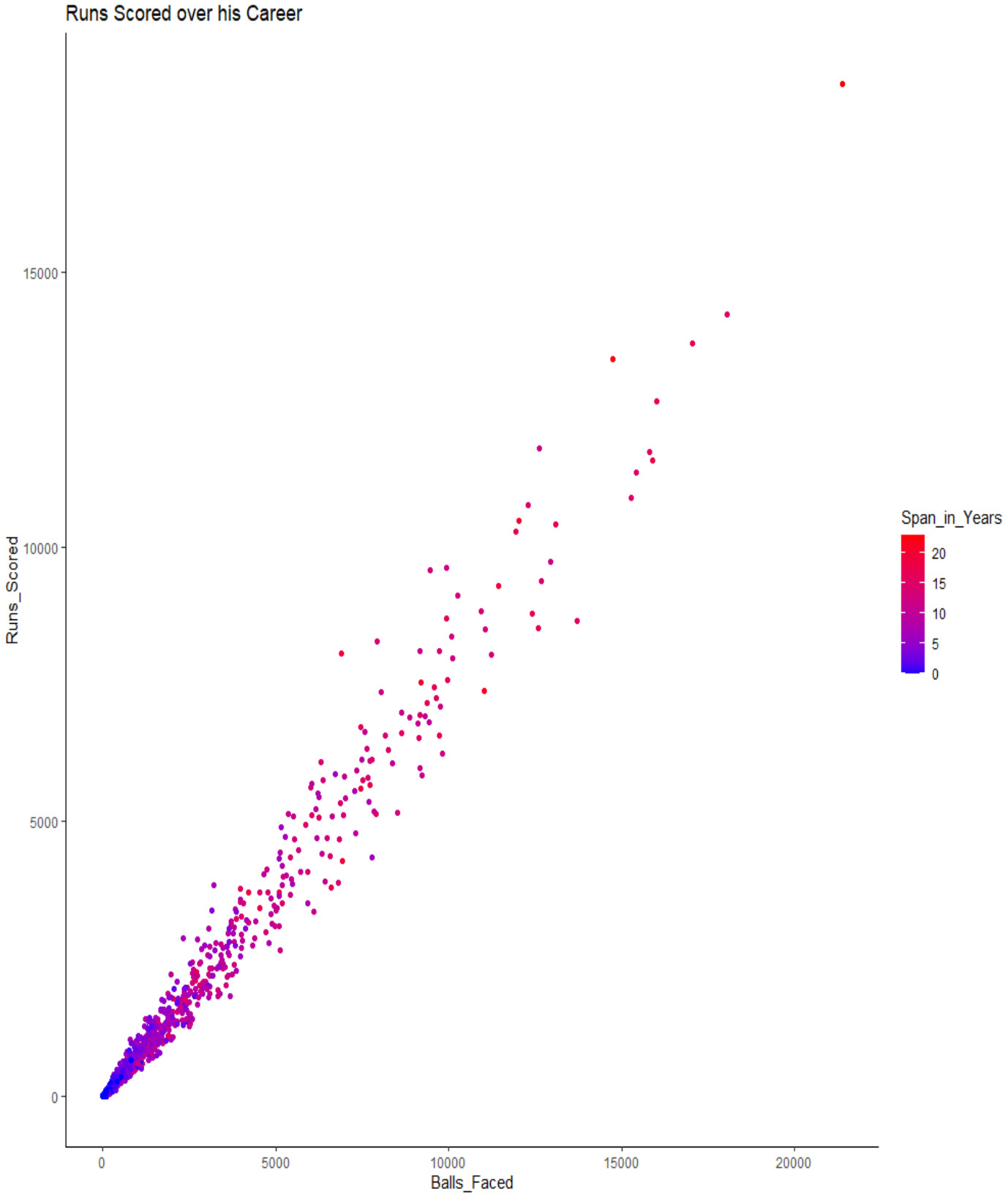
```
> ggplot(data=df, aes(x=as.numeric(Span_in_Years),y=as.numeric(Runs_Scored)))+
geom_bar(stat = "identity",fill="blue") + labs(title=" Years v/s Runs Scored",
x="Span in Years",y="Runs Scored")
```



#Scatter Plot

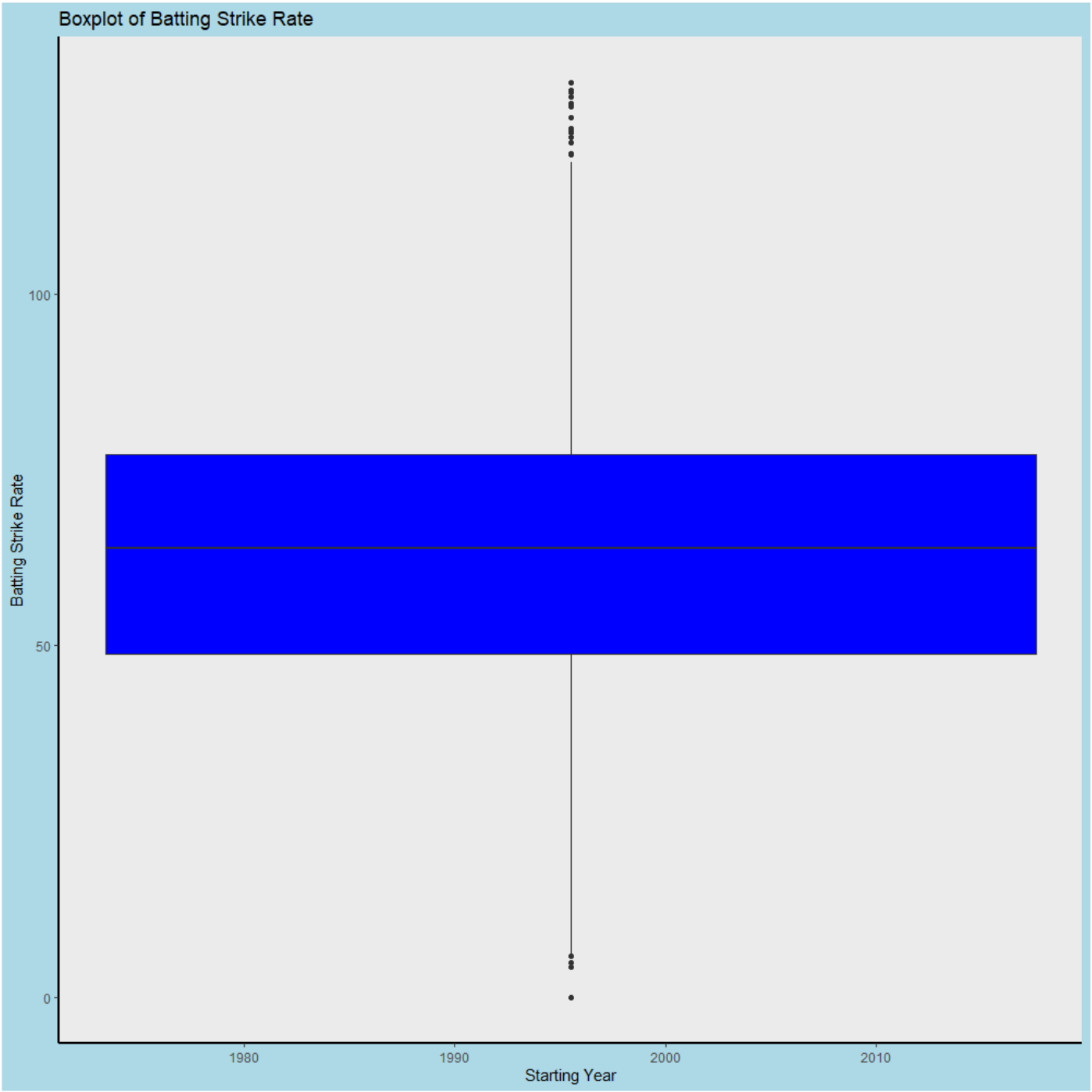
ggplot(data=df, aes(x=Balls_Faced,y=Runs_Scored,color= Span_in_Years)) + geom_point() +scale_color_gradient(low="blue", high="red") + ggtitle("Runs Scored over his Career") + theme_classic()

```
>  ggplot(data=df, aes(x=Balls_Faced,y=Runs_Scored,color= Span_in_Years)) + ge
om_point() +scale_color_gradient(low="blue", high="red") + ggtitle("Runs Score
d over his Career") + theme_classic()
```

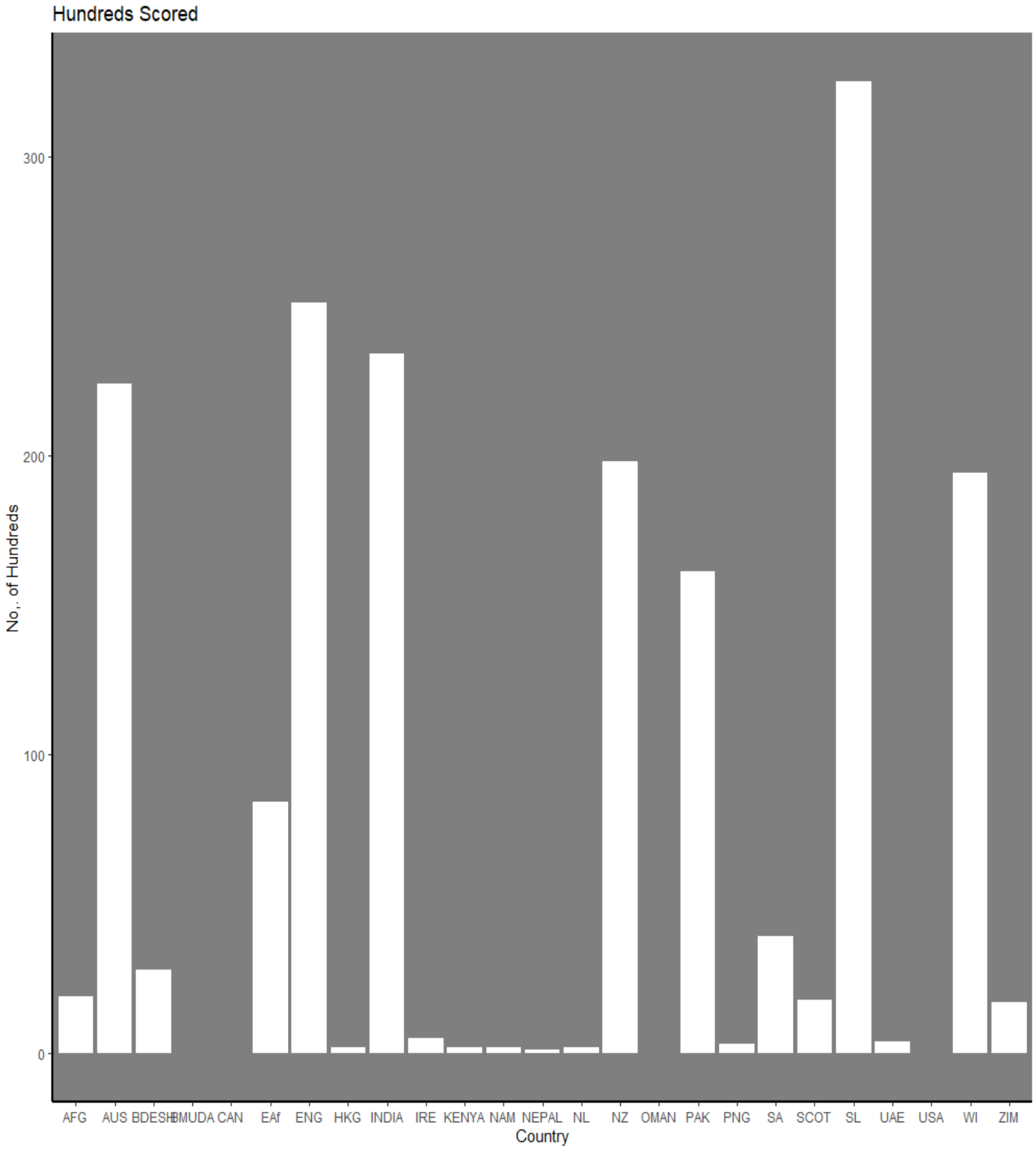


#Box Plot

```
ggplot(data=df, aes(x=Starting_Year,y=Batting_Strike_Rate)) + geom_boxplot(fill="blue")
+ ylim(0,130) + theme(plot.background = element_rect(fill = "lightblue")) + theme(panel.bor
der = element_blank(),
panel.grid.major = element_blank(),
panel.grid.minor = element_blank(),
axis.line = element_line(size = 1,
colour = "black")) + labs(title = "Boxplot of Batting Strike Rate",x="Starting Year",y="Batin
g Strike Rate")
```



```
ggplot(data=df, aes(x=Country, y=Hundreds_Scored))+ geom_bar(stat = "identity", fill="white")  + theme_dark() +  theme(panel.border = element_blank(), panel.grid.major = element_blank(),panel.grid.minor = element_blank(),  axis.line = element_line(size = 1colour = "black")) + labs(title = "Hundreds Scored",y="No. of Hundreds")
```



****

```
# Multiple Linear Regression

input <- df[,c("Innings_Batted","Runs_Scored","Batting_Average")]
# Create the relationship model.
model <- lm(formula = Runs_Scored~Innings_Batted*Batting_Average , data = i
nput)
# Show the model.
print(model)
```

```
Call:
lm(formula = Runs_Scored ~ Innings_Batted * Batting_Average,
    data = input)

Coefficients:
(Intercept)         Innings_Batted                Batting_Average
-6.9887                 -0.7157                        -0.3039

Innings_Batted:Batting_Average
                        0.8838
```

```
# Get the Intercept and coefficients as vector
# elements.
a<- coef(model)[1]
XIB <- coef(model)[2]
XBA <- coef(model)[3]
```

```
print(a)
> print(a)
(Intercept)
  -6.988657
```

```
print(XIB)
> print(XIB)
Innings_Batted
    -0.7157235
```

```
print(XBA)
> print(XBA)
Batting_Average
     -0.3038917
```

```
newdata <- data.frame(Innings_Batted=29,Batting_Average=100.000)
Y <- predict(model,newdata)
print(Y)
```

```
> print(Y)
         1
  2504.811
```

```
#Expected value should be 2900 but the actual value is calculated and observed t
o be 2504.811
#Hence the error is 396

install.packages("caTools")
library(caTools)
split <- sample.split(df,SplitRatio=0.8)
split
```

```
> split
 [1]  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
TRUE FALSE  TRUE  TRUE FALSE
```

```
train_data <- subset(df,split==TRUE)
test_data <- subset(df, split==FALSE)

#to check correlation
dt <- as.data.frame(apply(df,2,as.numeric))
cr <- cor(dt)
cr
```

```
    cr
```

	Player	Country	Starting_Year	Ending_Year	Span_in_Years	Matches_Played	Innings_Batted
Player	1	NA	NA	NA	NA	NA	NA
Country	NA	1	NA	NA	NA	NA	NA
Starting_Year	NA	NA	1.000000000	0.94381227	-0.2180544	-0.1080891	-0.09824892
Ending_Year	NA	NA	0.943812274	1.00000000	0.1167271	0.1535579	0.14630110
Span_in_Years	NA	NA	-0.218054422	0.11672710	1.0000000	0.7782964	0.72729484
Matches_Played	NA	NA	-0.108089065	0.15355794	0.7782964	1.0000000	0.96315049
Innings_Batted	NA	NA	-0.098248922	0.14630110	0.7272948	0.9631505	1.00000000
Not_Outs	NA	NA	-0.098588139	0.12675886	0.6706047	0.8666750	0.78452635
Runs_Scored	NA	NA	-0.070847443	0.13271791	0.6048359	0.8591427	0.94679852
Highest_Innings_Score	NA	NA	0.011187368	0.16107748	0.4420528	0.4699850	0.52628100
Batting_Average	NA	NA	0.002151976	0.09286446	0.2677681	0.3340959	0.41006236
Balls_Faced	NA	NA	-0.098398595	0.10878319	0.6169516	0.8551511	0.94402867
Batting_Strike_Rate	NA	NA	0.182682248	0.25572607	0.2061806	0.2279646	0.23186655
Hundreds_Scored	NA	NA	-0.013325394	0.12714438	0.4155117	0.6416517	0.73152306
Runs_Scored_above_50_or._50	NA	NA	-0.068308046	0.11741879	0.5520251	0.8040178	0.90163261
Ducks_Scored	NA	NA	-0.068167200	0.17025585	0.7076334	0.8827396	0.85612211

	Not_Outs	Runs_Scored	Highest_Innings_Score	Batting_Average	Balls_Faced	Batting_Strike_Rate
Player	NA	NA	NA	NA	NA	NA
Country	NA	NA	NA	NA	NA	NA
Starting_Year	-0.09858814	-0.07084744	0.01118737	0.002151976	-0.0983986	0.1826822
Ending_Year	0.12675886	0.13271791	0.16107748	0.092864459	0.1087832	0.2557261
Span_in_Years	0.67060466	0.60483586	0.44205279	0.267768122	0.6169516	0.2061806
Matches_Played	0.86667497	0.85914265	0.46998495	0.334095932	0.8551511	0.2279646
Innings_Batted	0.78452635	0.94679852	0.52628100	0.410062359	0.9440287	0.2318666
Not_Outs	1.00000000	0.62852888	0.28435001	0.233679549	0.6253451	0.1887402

```
Runs_Scored                 0.62852888  1.00000000           0.53757608     0.481526268   0.9899316          0.2133494
Highest_Innings_Score       0.28435001  0.53757608           1.00000000     0.637123738   0.5273394          0.3604263
Batting_Average             0.23367955  0.48152627           0.63712374     1.000000000   0.4797925          0.4316178
Balls_Faced                 0.62534507  0.98993164           0.52733935     0.479792463   1.0000000          0.1851710
Batting_Strike_Rate         0.18874022  0.21334939           0.36042626     0.431617762   0.1851710          1.0000000
Hundreds_Scored             0.39392093  0.86757568           0.46405435     0.406512964   0.8308300          0.1619106
Runs_Scored_above_50_or._50 0.57324936  0.97973106           0.50714152     0.483596752   0.9787727          0.1863005
Ducks_Scored                0.73506199  0.71589569           0.42500042     0.214470319   0.7117517          0.1840740
                      Hundreds_Scored Runs_Scored_above_50_or._50 Ducks_Scored
Player                             NA                          NA           NA
Country                            NA                          NA           NA
Starting_Year              -0.01332539                -0.06830805   -0.0681672
Ending_Year                 0.12714438                 0.11741879    0.1702559
Span_in_Years               0.41551170                 0.55202509    0.7076334
Matches_Played              0.64165167                 0.80401783    0.8827396
Innings_Batted              0.73152306                 0.90163261    0.8561221
Not_Outs                    0.39392093                 0.57324936    0.7350620
Runs_Scored                 0.86757568                 0.97973106    0.7158957
Highest_Innings_Score       0.46405435                 0.50714152    0.4250004
Batting_Average             0.40651296                 0.48359675    0.2144703
Balls_Faced                 0.83082998                 0.97877269    0.7117517
Batting_Strike_Rate         0.16191063                 0.18630054    0.1840740
Hundreds_Scored             1.00000000                 0.82208328    0.5259969
Runs_Scored_above_50_or._50 0.82208328                 1.00000000    0.6541400
Ducks_Scored                0.52599693                 0.65413997    1.0000000

>
```

```
model <- lm(Runs_Scored~Innings_Batted*Batting_Average,dt)
summary(model)
Call:
lm(formula = Runs_Scored ~ Innings_Batted * Batting_Average,
    data = dt)

Residuals:
     Min       1Q   Median       3Q      Max
-2265.73   -15.24     9.27    26.13  1373.39

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                    -6.988657   6.316782  -1.106 0.268682
Innings_Batted                 -0.715723   0.207019  -3.457 0.000555 ***
Batting_Average                -0.303892   0.293885  -1.034 0.301219
Innings_Batted:Batting_Average  0.883774   0.005928 149.090  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 156.8 on 2370 degrees of freedom
  (213 observations deleted due to missingness)
Multiple R-squared:  0.991,     Adjusted R-squared:  0.991
F-statistic: 8.684e+04 on 3 and 2370 DF,  p-value: < 2.2e-16


predicted <- predict(model, test_data)
predicted

          4           9          13          16          20          25
29          32
  739.7687479   224.3701427     7.9524467     4.6888716            NA   256.5728035
29.1977174          NA
          36          41          45          48          52          57
61          64
 2342.7762039   842.4729981   270.5396075   341.4906310   230.9177421   130.9267598
175.7383045    52.7010437
          68          73          77          80          84          89
93          96
   49.4757511    28.1713120     4.0209777    -3.9052715    -7.7043801   376.0068324
112.9342363    45.0033631
         100         105         109         112         116         121
125         128
```

```
   23.0293910    24.4729487    -5.4927903  4942.5666555   466.6102669   250.9960781
135.1292436   150.4894652
          132           137           141           144           148           153
157           160
   78.6306260    36.6390730    29.6349686     2.6013268     4.6888716    -4.7609620
-7.6882752  3881.6552607
          164           169           173           176           180           185
189           192
  532.1930910   244.5329638     1.5737395    -4.2250852  2215.3405616   483.4483617
112.7626863    64.0308992
          196           201           205           208           212           217
221           224
   26.2377069    14.9984024  8475.9444328  2922.7193803   962.5230790   116.5345019
52.1861568   191.0667002
          228           233           237           240           244           249
253           256
   46.8265527            NA            NA  6622.7572969   571.2302413   173.5988044
158.0144110    20.5826465
          260           265           269           272           276           281
285           288
    6.4980619    -1.1018204    -6.7883962    -7.1244976            NA   794.5126880
651.1176835   525.5614599
          292           297           301           304           308           313
317           320
  180.6778216   118.5336372     8.7120154    -1.1018204    -6.5446151  2355.2140176
54.5171311    29.5068288
          324           329           333           336           340           345
349           352
   -5.9647327  4650.8241099   852.5607614   752.5895266   157.9909375   133.2439313
33.1179268    -4.7609620
          356           361           365           368           372           377
381           384
 1548.2565301   677.3038545   703.0379060   265.4762247   193.6732315    80.4157839
77.1352446   171.2778475
          388           393           397           400           404           409
413           416
   28.5250272    47.0424329     4.7528060    -6.5446151  5649.1661573   690.9792887
261.7067738   226.3622837
          420           425           429           432           436           441
445           448
   78.1987835    33.7057815    -4.0291337            NA  3454.8260776   134.4526134
50.7236576    29.9567166
          452           457           461           464           468           473
477           480
   -0.1579357            NA  8150.8217543  3960.8686537   474.9200262   237.6180667
122.5534548    94.9478805
          484           489           493           496           500           505
509           512
   33.8925790    40.0627074    15.5121961    15.9980888     0.3618362            NA
NA  3413.7768000
          516           521           525           528           532           537
541           544
 1256.8832215   778.2695168   229.5959365   112.3546519    84.8612932    -4.0291337  1
999.4422697  2127.8536545
          548           553           557           560           564           569
573           576
  532.7183945   434.6379862   160.3600516    76.2391895    41.0057476    30.5385110
16.6859114    -2.8916612
          580           585           589           592           596           601
605           608
           NA  4391.6777823  1155.6951442   201.2269583    25.8750376            NA  5
290.8222973  1619.2753973
          612           617           621           624           628           633
637           640
  842.9222671   431.0208638    93.8867716    75.2474752    19.8079944    23.7282037
-4.8049677            NA
          644           649           653           656           660           665
669           672
11954.8623440  1296.8721673   582.3927854   538.4549546   184.7337001    10.2719766
NA  6169.2194916
          676           681           685           688           692           697
701           704
 6412.9042043  1192.3492988   681.0417548   339.9521221   222.1771976    94.8279760
53.1569454    37.9550237
          708           713           717           720           724           729
733           736
```

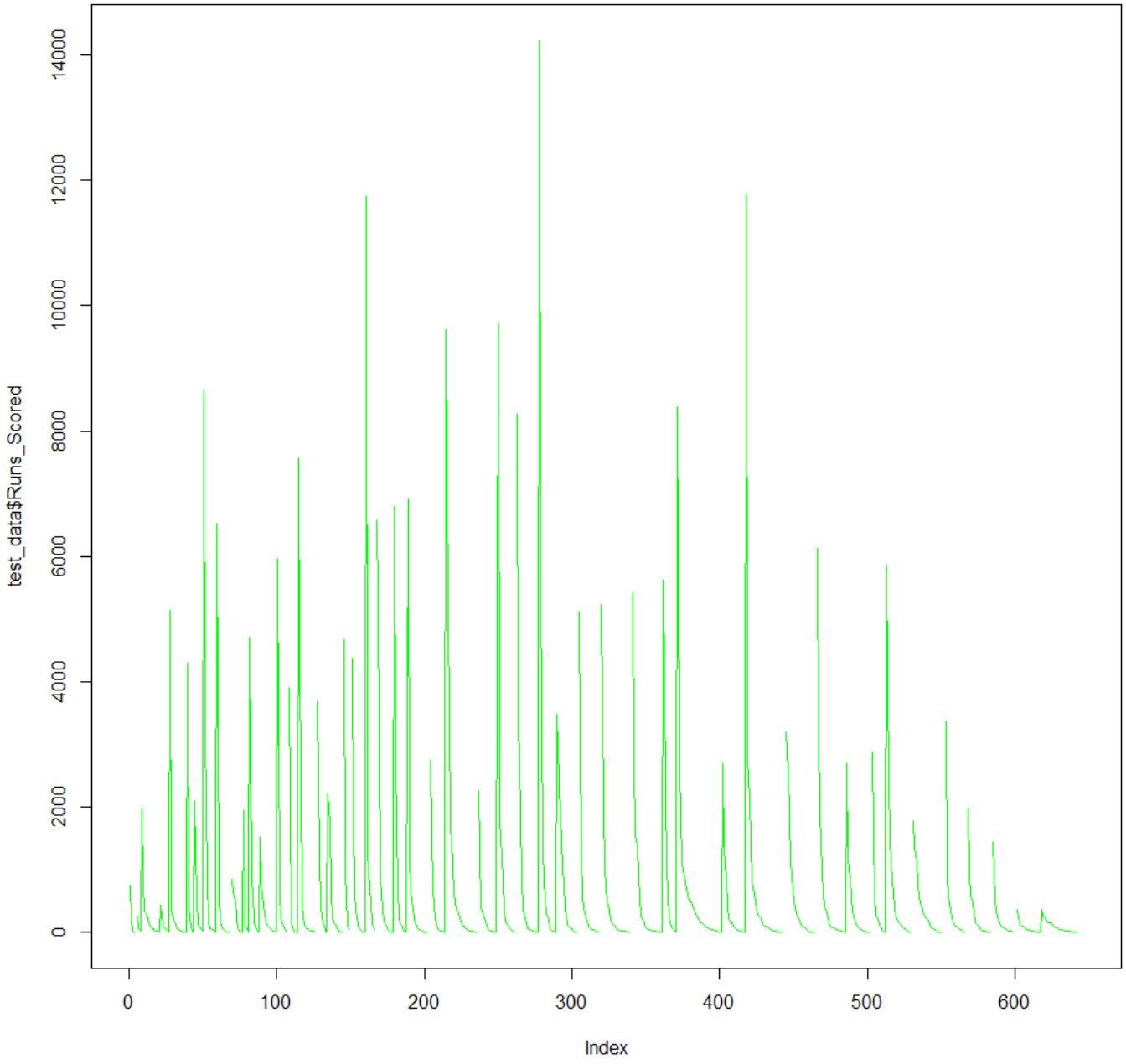
```
   14.3384807    -4.0291337    -7.7043801  6543.4101385  2713.2071320   756.7071923
174.7099850   138.5093572
          740           745           749           752           756           761
765           768
  133.9370683    21.3807730    -5.4927903    -7.7043801  9117.5603212  1631.1327217
881.9223846   511.6023297
          772           777           781           784           788           793
797           800
  330.0005342   178.4630988    89.6918175    61.8604240    76.2391895    10.3616040
4.1395668    30.5825167
          804           809           813           816           820           825
829           832
   -3.2973054    -6.5446151            NA  2524.6015314  1473.4662134   945.5962449
453.8284658   145.6602850
          836           841           845           848           852           857
861           864
  190.6326249    24.5080756     2.5573211    40.1203618     0.3618362    -7.1244976  8
631.9223362  5805.3658295
          868           873           877           880           884           889
893           896
 3228.7123087  1610.0069827  1125.5598899   611.2050358   483.4239208   287.6230304
258.6459505   191.8629938
          900           905           909           912           916           921
925           928
  201.6542146    87.0845970    99.9776040   112.7960805    30.5825167    13.7512714
156.4430933    19.3893722
          932           937           941           944           948           953
957           960
    3.2891494            NA    -9.1358270            NA  2717.9603305   503.9545832
393.9617138   309.8636940
          964           969           973           976           980           985
989           992
  284.2155242    86.2088837    70.3539910    93.5470068    21.9602274    32.7902826
2.5097340    -5.3848502
          996          1001          1005          1008          1012          1017
1021          1024
   -6.5446151  9848.3294951  1613.0416036  1300.3533310   986.3974642   325.8531938
316.0420571    99.7692989
         1028          1033          1037          1040          1044          1049
1053          1056
  154.5159515    49.8579822    22.7085244     1.8254928    -2.5654771            NA  7
396.1972417  3457.7535837
         1060          1065          1069          1072          1076          1081
1085          1088
 1647.2152757   728.3494711   310.3515702   348.2268772   112.6530135    42.0037118
30.5825167    28.5126456
         1092          1097          1101          1104          1108          1113
1117          1120
    6.4980619    -1.3256729     0.7716410    -2.5866771    -4.0291337 13806.5977697  5
553.7648315  1200.9208474
         1124          1129          1133          1136          1140          1145
1149          1152
 1017.8546333   543.2442886   389.9477047   197.4201796    67.4153985    30.7704960
NA     4.7528060
         1156          1161          1165          1168          1172          1177
1181          1184
   -6.9564469  3185.5231480  2285.8688144  1719.0237701  1316.6616265  1310.2362561
487.7170376   209.3720115
         1188          1193          1197          1200          1204          1209
1213          1216
  187.6604796   115.3206055    37.8127884    45.0789331    14.1226004    -3.2973054
NA            NA
         1220          1225          1229          1232          1236          1241
1245          1248
 5588.1935526  1083.0675552   703.4828494   350.1126973   532.8459488   169.8051712
91.9151331   101.1512469
         1252          1257          1261          1264          1268          1273
1277          1280
   31.0986252    48.2971845    21.1170632   163.2093948            NA            NA
NA  4893.3510866
         1284          1289          1293          1296          1300          1305
1309          1312
 1940.8716951   863.9808829   674.8094983   582.9753482   401.7698911   154.4756591
133.7000981    98.6895732
         1316          1321          1325          1328          1332          1337
1341          1344
```

```
   64.4981013    49.0999225    24.5121704    35.7110258    11.6536130    35.0614980
7.3707717    -0.2857404
         1348          1353          1357          1360          1364          1369
1373          1376
   -2.4854378     0.2538961    -8.4201035            NA  5123.6931170  2464.1596231  1
384.3087336  1845.1432148
         1380          1385          1389          1392          1396          1401
1405          1408
  881.0926503   624.0697844   197.9680098   205.1139932   216.0727139    90.3813588
86.6757291    33.6687510
         1412          1417          1421          1424          1428          1433
1437          1440
   73.0242499     9.1437759    22.3166855     4.7528060     1.8254928     4.7528060
-4.7609620            NA
         1444          1449          1453          1456          1460          1465
1469          1472
   -7.7043801  5897.4072168  2337.0633765   907.6261647   792.3532130   199.2003538
57.1035297    43.6488472
         1476          1481          1485          1488          1492          1497
1501          1504
   25.6513342    -7.1244976  8793.7890248  6317.7662517  2512.2759094  1162.2914022
845.1527702   917.9222663
         1508          1513          1517          1520          1524          1529
1533          1536
  719.7235866   559.3205435   513.5116574   416.0966538   439.9126226   421.8159761
349.0078971   242.7817261
         1540          1545          1549          1552          1556          1561
1565          1568
  205.3281442   149.2450313   207.2710149   212.9826840    78.4937636   170.0598839
102.4934675   131.9966390
         1572          1577          1581          1584          1588          1593
1597          1600
   76.8407249    38.9865037    35.6064530   118.1287806     0.4139745    19.0333422
NA    -6.9564469
         1604          1609          1613          1616          1620          1625
1629          1632
   -7.7043801  2855.3592427  1159.3845363  1091.9159791   446.2796754   232.8007242
134.2593567   107.2989340
         1636          1641          1645          1648          1652          1657
1661          1664
  197.4114071    47.0024650    56.6303342    40.2562394    22.4703235     6.4980619
0.3618362    -5.3848502
         1668          1673          1677          1680          1684          1689
1693          1696
   -6.5446151 12288.8690177  2523.3555747  2056.8670585  1285.7717134   933.7995876
581.4439381   592.2059817
         1700          1705          1709          1712          1716          1721
1725          1728
  283.6711989   451.4432589   206.4037177   257.8452133   110.4131105    89.8382152
71.1964306    75.8472864
         1732          1737          1741          1744          1748          1753
1757          1760
   67.6973033    37.1531977    71.7323073            NA     5.4846344     1.8254928
NA    -4.8049677
         1764          1769          1773          1776          1780          1785
1789          1792
   -6.2246186    -9.1358270            NA            NA  3242.2145879  2255.9736161  1
840.8333485  1100.4557367
         1796          1801          1805          1808          1812          1817
1821          1824
  819.7313613   633.7770036   383.8098282   235.9297111   210.2420373   138.4175037
105.1148794    84.9196897
         1828          1833          1837          1840          1844          1849
1853          1856
   41.3442215    43.6488472    40.3541233            NA    -1.8336487     0.2538961
NA            NA
         1860          1865          1869          1872          1876          1881
1885          1888
          NA  5888.0771988  3158.2314470  1737.8861615   911.2554385   683.0061053
424.7385104   274.6167254
         1892          1897          1901          1904          1908          1913
1917          1920
  183.0426598   185.2628942    69.1040405    93.6908200    63.3675529    35.3553700
67.6973675    31.2665154
         1924          1929          1933          1936          1940          1945
1949          1952
```

```
   12.7657021     3.0732692    19.9895079     5.2839176    -7.1244976  2831.2178576  1
421.2230819   931.9355267
         1956           1961          1965          1968          1972          1977
1981          1984
  476.5491547   308.6545024   170.6655771   214.6143230    93.6708913   102.2254843
20.1212006    91.8400646
         1988           1993          1997          2000          2004          2009
2013          2016
    5.6222327             NA    38.7045347    -4.2250852    -6.6203456            NA  2
767.9752635  1578.9804447
         2020           2025          2029          2032          2036          2041
2045          2048
  479.5845150   349.7759045   193.6732315   293.7490749   215.3184329    -0.1659079
1.4576667    -7.7043801
         2052           2057          2061          2064          2068          2073
2077          2080
 6008.5172306  2079.4400875  1399.9567320  1054.7997821   726.5931417   383.9760986
209.3720115   174.6067839
         2084           2089          2093          2096          2100          2105
2109          2112
  198.3234113    91.2750072    76.2391895    78.1097920    56.2385965            NA
1.0936645    -3.2973054
         2116           2121          2125          2128          2132          2137
2141          2144
   -6.5446151             NA  1781.0394274  1813.9917230   976.6309705   646.9882936
514.3116404   385.6505611
         2148           2153          2157          2160          2164          2169
2173          2176
  309.6975937   268.0338916   241.3299022   255.8950738   197.2166618   128.7960929
55.0194559    66.2429738
         2180           2185          2189          2192          2196          2201
2205          2208
  101.1934191    61.4448597    16.6859114     2.6013268            NA            NA
NA            NA
         2212           2217          2221          2224          2228          2233
2237          2240
 3371.9487242   899.7894457   540.9981572   291.1114163   212.0089462   113.5005997
90.6915681   143.2668229
         2244           2249          2253          2256          2260          2265
2269          2272
   57.5915778             NA    22.1554251     0.9938570    11.2113920    -7.7043801
NA  1753.6975297
         2276           2281          2285          2288          2292          2297
2301          2304
  960.8716773   571.7411068   408.0958430   263.4790543   146.9731930   160.7980297
101.1375022    41.3442215
         2308           2313          2317          2320          2324          2329
2333          2336
   25.2439987    28.8805797    11.5313885     9.1437759    11.9910499            NA
-7.7043801            NA
         2340           2345          2349          2352          2356          2361
2365          2368
 1312.5750199   411.6476189   473.3519249   284.2963604   404.2226511   108.4855304
97.7801248    47.9771636
         2372           2377          2381          2384          2388          2393
2397          2400
   74.1597773    88.7800783    67.6173283            NA            NA    -4.8049677
NA            NA
         2404           2409          2413          2416          2420          2425
2429          2432
  318.0251668   238.8407172   171.1714948    78.6945604    59.6193552    68.0036512
42.9274539   129.3778873
         2436           2441          2445          2448          2452          2457
2461          2464
   19.8079944     9.8756042     4.0209777     8.0421182     4.7528060    -3.6452028
-5.3848502    -7.6882752
         2468           2473          2477          2480          2484          2489
2493          2496
   -7.7043801   282.3217570   235.1972910   178.4335732   143.7788869   142.8882245
203.8101064   107.2989340
         2500           2505          2509          2512          2516          2521
2525          2528
   95.3404086    68.5051687    81.2402576    42.0760499   102.5946503    20.1212006
29.5967807   112.8820980
         2532           2537          2541          2544          2548          2553
2557          2560
```

```
    6.2127993    19.3893722     3.2891494     7.1194930    -0.3699921            NA
6.3044741    -5.9647327
         2564          2569          2573          2576          2580          2585
   -6.5446151            NA            NA            NA            NA    -4.8049677
```

```
plot(test_data$Runs_Scored,type = "l",lty=1.8, col="green")
```



```
lines(predicted,type="l", col="red")
```