Test Matches Record

Bunty Patil

2024-03-15

Dataset Summary

The test matches record dataset is offering insights into the performance of teams and players in the longest format of international cricket. The dataset contains the data of 106 players from year 1908 to year 2024 and from nine test cricket playing countries. The dataset includes information such as the country names, matches played, total innings, total runs, averages etc.

Installing and loading packages

To start with our report we need to install some of the packages as follows:

```r
install.packages("tidyverse", repos = "http://cran.us.r-project.org");
```

```
## Installing package into 'C:/Users/raahu/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\raahu\AppData\Local\Temp\Rtmpsv7Qwh\downloaded_packages
```

```r
install.packages("janitor", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/raahu/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'janitor' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\raahu\AppData\Local\Temp\Rtmpsv7Qwh\downloaded_packages
```

Then load the packages:

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(janitor)
```

```
## Warning: package 'janitor' was built under R version 4.3.3
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

Data Collection

The collection of raw data is the first step of the data analysis. The raw test matches record dataset is downloaded in csv format from the platform named **Kaggle** and stored in local drive.

To import the csv file, we use read_csv function

```
test_matches <- read_csv("test_matches_records.csv")
```

```
## New names:
## Rows: 107 Columns: 17
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (8): Names, Country, Span, Matches, Highest Score, Fours, Balls_Faced, S... dbl
## (9): ...1, Innings, Not Outs, Total Runs, Average, Strike Rate, Hundreds...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

Data cleaning

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.

First lets get all columns name of our data

```
colnames(test_matches)
```

```
##  [1] "...1"          "Names"         "Country"       "Span"
##  [5] "Matches"       "Innings"       "Not Outs"      "Total Runs"
##  [9] "Highest Score" "Average"       "Fours"         "Balls_Faced"
## [13] "Strike Rate"   "Hundreds"      "Fifty"         "Zeroes"
## [17] "Sixes"
```

As our output shows, first column is not valid and other columns are not properly named. To rename our column we use rename function as follows

```r
test_matches_records <- rename(test_matches, "sr_no" = "...1")
```

Now to name our all columns properly, we use clean_names() function

```r
test_matches_records <- clean_names(test_matches_records)
```

Now, lets check our values in all columns

```r
glimpse(test_matches_records)
```

```
## Rows: 107
## Columns: 17
## $ sr_no         <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16~
## $ names         <chr> "SR Tendulkar", "RT Ponting", "JH Kallis", "R Dravid", "~
## $ country       <chr> "IND", "AUS", "SA", "IND", "ENG", "SL", "WI", "WI", "SL"~
## $ span          <chr> "1989-2013", "1995-2012", "1995-2013", "1996-2012", "200~
## $ matches       <chr> "200", "168", "166", "164", "161", "134", "131", "164", ~
## $ innings       <dbl> 329, 287, 280, 286, 291, 233, 232, 280, 252, 249, 265, 2~
## $ not_outs      <dbl> 33, 29, 40, 32, 16, 17, 6, 49, 15, 20, 44, 46, 16, 19, 2~
## $ total_runs    <dbl> 15921, 13378, 13289, 13288, 12472, 12400, 11953, 11867, ~
## $ highest_score <chr> "248*", "257", "224", "270", "294", "319", "400*", "203*~
## $ average       <dbl> 53.78, 51.85, 55.37, 52.31, 45.35, 57.40, 52.88, 51.37, ~
## $ fours         <chr> "2058+", "1509", "1488", "1654", "1442", "1491", "1559",~
## $ balls_faced   <chr> "29437+", "22782", "28903", "31258", "26562", "22882", "~
## $ strike_rate   <dbl> 54.04, 58.72, 45.97, 42.51, 46.95, 54.19, 60.51, 43.31, ~
## $ hundreds      <dbl> 51, 41, 45, 36, 33, 38, 34, 30, 34, 30, 27, 32, 34, 34, ~
## $ fifty         <dbl> 68, 62, 58, 63, 57, 52, 48, 66, 50, 60, 63, 50, 45, 33, ~
## $ zeroes        <dbl> 14, 17, 16, 8, 9, 11, 17, 15, 15, 12, 11, 22, 12, 19, 9,~
## $ sixes         <chr> "69", "73", "97", "21", "11", "51", "88", "36", "61", "4~
```

As we can see from the output that our sr no is starting from 0 instead of 1. To fix this

```r
test_matches_records$sr_no <- test_matches_records$sr_no + 1
```

Also, there are special characters like '*' and '+' in matches, balls_faced, fours and sixes column. To remove it

```r
testmatches <- test_matches_records %>%
  mutate(
    matches = as.numeric(str_remove(matches, "[*+]")),
    fours = as.numeric(str_remove(fours, "[*+]")),
    sixes = as.numeric(str_remove(sixes, "[*+]")),
    balls_faced = as.numeric(str_remove(balls_faced, "[*+]"))
    )
```

Lets check the result

```
head(testmatches)
```

```
## # A tibble: 6 x 17
##   sr_no names    country span  matches innings not_outs total_runs highest_score
##   <dbl> <chr>    <chr>   <chr>   <dbl>   <dbl>    <dbl>      <dbl> <chr>
## 1     1 SR Tend~ IND     1989~     200     329       33      15921 248*
## 2     2 RT Pont~ AUS     1995~     168     287       29      13378 257
## 3     3 JH Kall~ SA      1995~     166     280       40      13289 224
## 4     4 R Dravid IND     1996~     164     286       32      13288 270
## 5     5 AN Cook  ENG     2006~     161     291       16      12472 294
## 6     6 KC Sang~ SL      2000~     134     233       17      12400 319
## # i 8 more variables: average <dbl>, fours <dbl>, balls_faced <dbl>,
## #   strike_rate <dbl>, hundreds <dbl>, fifty <dbl>, zeroes <dbl>, sixes <dbl>
```

Now lets check the duplicates in our data

```
get_dupes(testmatches)
```

```
## No variable names specified - using all columns.

## No duplicate combinations found of: sr_no, names, country, span, matches, innings, not_outs, total_ru

## # A tibble: 0 x 18
## # i 18 variables: sr_no <dbl>, names <chr>, country <chr>, span <chr>,
## #   matches <dbl>, innings <dbl>, not_outs <dbl>, total_runs <dbl>,
## #   highest_score <chr>, average <dbl>, fours <dbl>, balls_faced <dbl>,
## #   strike_rate <dbl>, hundreds <dbl>, fifty <dbl>, zeroes <dbl>, sixes <dbl>,
## #   dupe_count <int>
```

There are no duplicates in our data.

Data organization

Data organization is a process of organizing raw data, by classifying them into different categories.

Now we want to seperate years from span column into 'from' and 'to' column for our easy analysis

```
test_matches_clean <- separate(testmatches, col = span,
                               into = c("from", "to"), sep = "-")
```

Changing data type of 'from' and 'to' columns to numeric data type

```
test_matches_clean$to <- as.numeric(as.character(test_matches_clean$to))
test_matches_clean$from <- as.numeric(as.character(test_matches_clean$from))
```

```
colnames(test_matches_clean)
```

```
##  [1] "sr_no"         "names"         "country"       "from"
##  [5] "to"            "matches"       "innings"       "not_outs"
##  [9] "total_runs"    "highest_score" "average"       "fours"
## [13] "balls_faced"   "strike_rate"   "hundreds"      "fifty"
## [17] "zeroes"        "sixes"
```

Data Analysis

Lets start with our calculation and finding answers to our questions.

Which are the test matches playing countries

```r
test_matches_clean %>%
  distinct(country)
```

```
## # A tibble: 9 x 1
##   country
##   <chr>
## 1 IND
## 2 AUS
## 3 SA
## 4 ENG
## 5 SL
## 6 WI
## 7 PAK
## 8 NZ
## 9 BAN
```

There are total 9 test cricket playing countries.

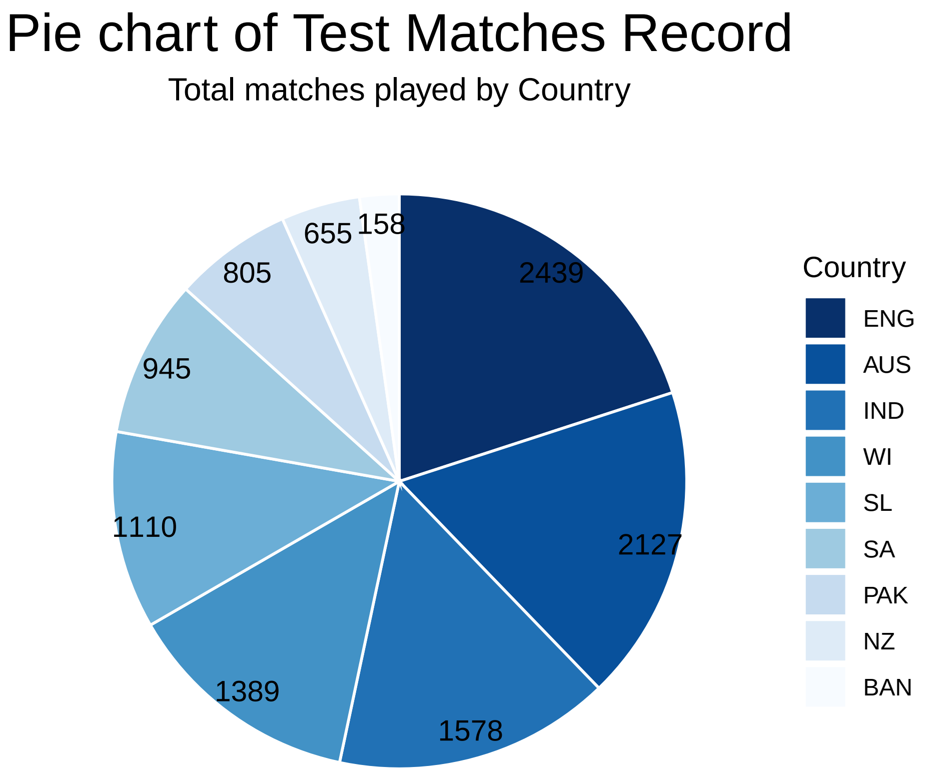
Total Matches played by each country

```r
matches_played_by_country <- test_matches_clean %>%
  group_by(country) %>%
  summarise(total_matches = sum(matches)) %>%
  arrange(desc(total_matches))
print(matches_played_by_country)
```

```
## # A tibble: 9 x 2
##   country total_matches
##   <chr>           <dbl>
## 1 ENG              2439
## 2 AUS              2127
## 3 IND              1578
## 4 WI               1389
## 5 SL               1110
## 6 SA                945
## 7 PAK               805
## 8 NZ                655
## 9 BAN               158
```

Visualization:

```r
matches_pie_chart <- matches_played_by_country %>%
  ggplot(aes(x = '', y = reorder(total_matches, country ) ,
             fill = factor(reorder(country, total_matches)) )) +
  geom_bar(stat = "identity", width = 1, color = 'white') +
  geom_text(aes(x = 1.4,label = total_matches),
            position =  position_stack(vjust= 0.5), color = 'black') +
  theme_void() +
  theme_classic() +
  theme(legend.position = "right") +
  coord_polar("y", start = 0)  +
  theme(plot.title = element_text(hjust = 0.5, size = 20)) +
  theme(plot.subtitle = element_text(hjust = 0.5, size = 12)) +
  theme(axis.line = element_blank()) +
  theme(axis.text = element_blank()) +
  theme(axis.ticks = element_blank()) +
  guides(fill = guide_legend(reverse = TRUE)) +
  labs(x = NULL, y = NULL,
       title = 'Pie chart of Test Matches Record',
       subtitle = 'Total matches played by Country') +
  scale_fill_brewer(palette = "Blues", name = "Country")
print(matches_pie_chart)
```



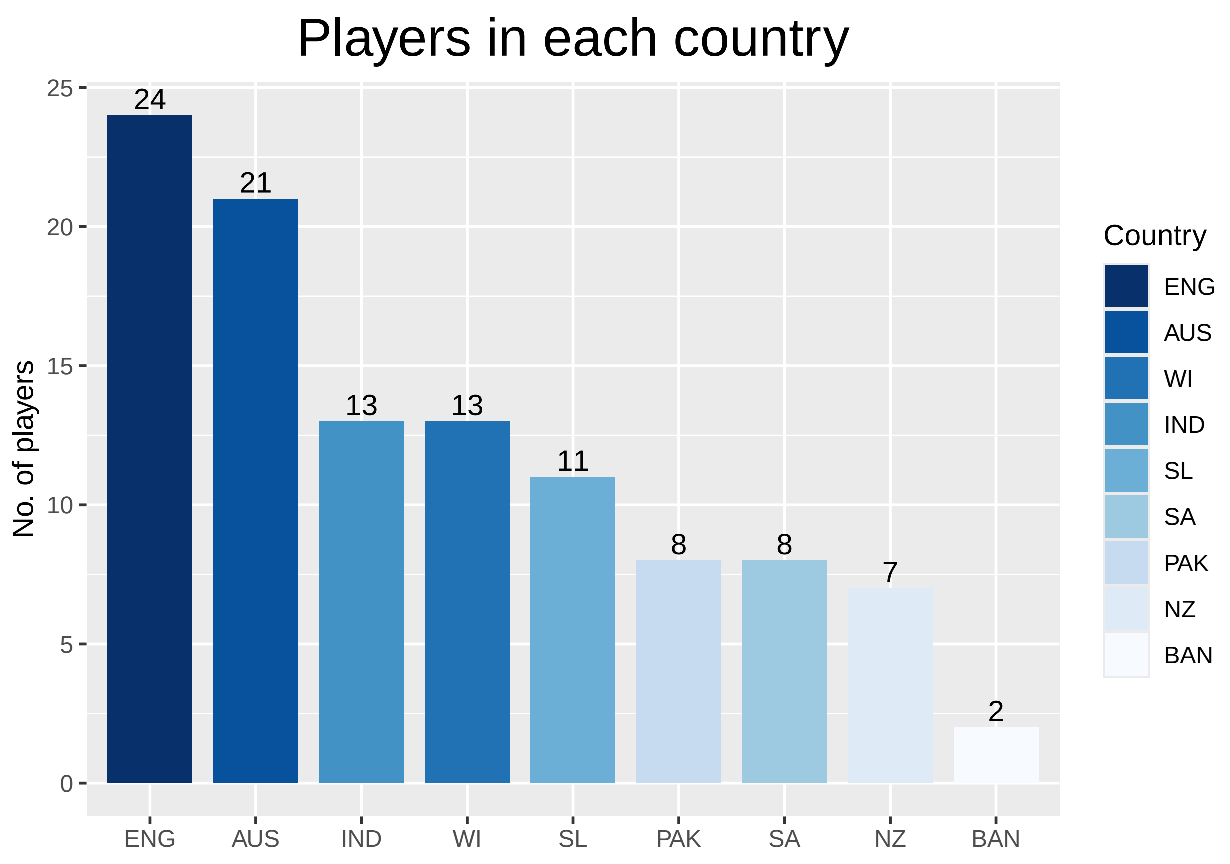
Total players in each country

```
players_in_country <- test_matches_clean %>%
  group_by(country) %>%
  summarise(players_count = n()) %>%
  arrange(desc(players_count))
print(players_in_country)
```

```
## # A tibble: 9 x 2
##   country players_count
##   <chr>           <int>
## 1 ENG                24
## 2 AUS                21
## 3 IND                13
## 4 WI                 13
## 5 SL                 11
## 6 PAK                 8
## 7 SA                  8
## 8 NZ                  7
## 9 BAN                 2
```

Visualization:

```
players_bar_chart <-players_in_country %>%
  ggplot(aes(x = reorder(country, desc(players_count)),
             y = players_count,
             fill = factor(reorder(country, players_count)))) +
  geom_bar (stat="identity", width = 0.8) +
  geom_text(aes(label = players_count),
            vjust = -0.3, size = 4) +
  labs(title = "Players in each country",
       x= NULL, y="No. of players") +
  theme_get() +
  theme(plot.title = element_text(hjust = 0.5, size = 20)) +
  guides(fill = guide_legend(reverse = TRUE)) +
  scale_fill_brewer(palette = "Blues",
                    name = "Country")
print(players_bar_chart)
```



Top 10 players with max runs and there highest scores

```
## # A tibble: 10 x 3
##    names             total_runs highest_score
##    <chr>                  <dbl> <chr>
##  1 SR Tendulkar           15921 248*
##  2 RT Ponting             13378 257
##  3 JH Kallis              13289 224
##  4 R Dravid               13288 270
##  5 AN Cook                12472 294
##  6 KC Sangakkara          12400 319
##  7 BC Lara                11953 400*
##  8 S Chanderpaul          11867 203*
##  9 DPMD Jayawardene       11814 374
## 10 JE Root                11447 254
```

Players from IND having max Average

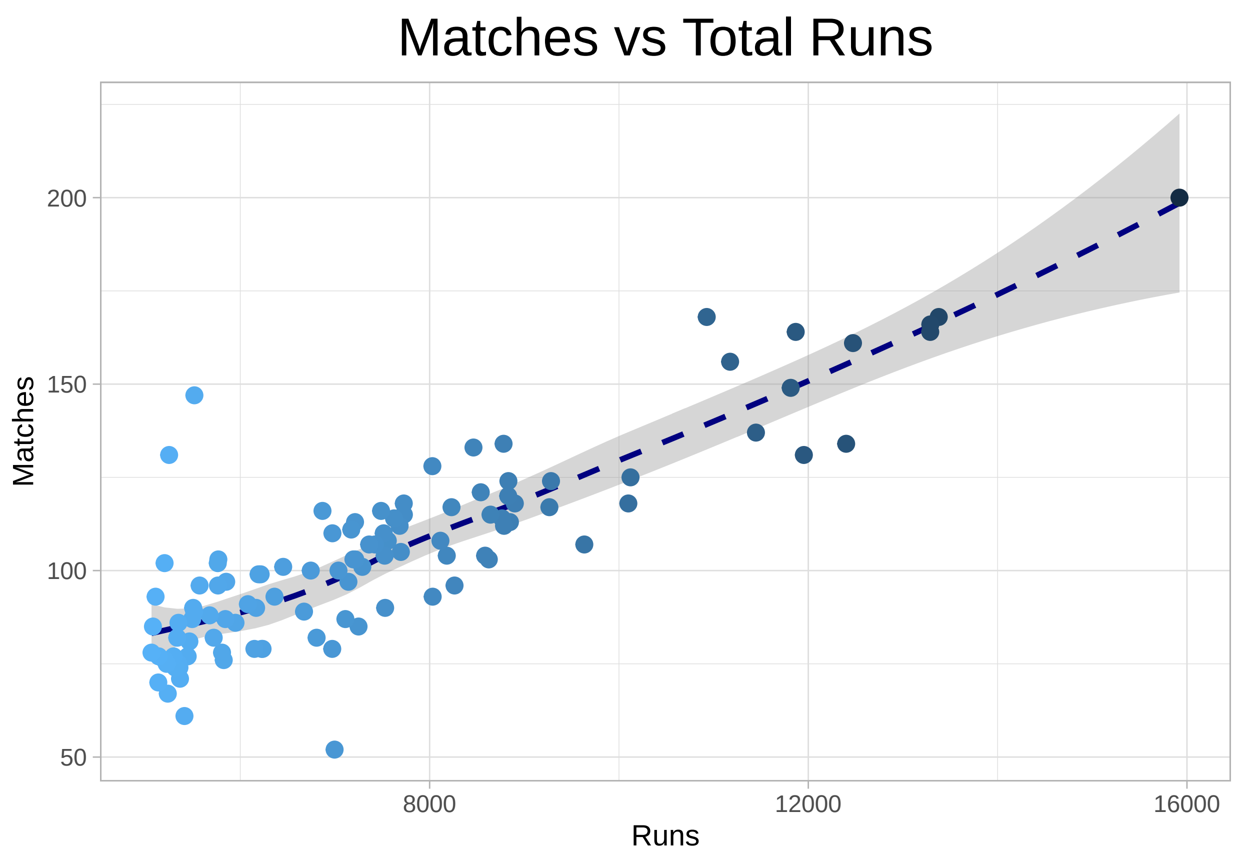
```
## # A tibble: 13 x 2
##    names          average
##    <chr>            <dbl>
##  1 SR Tendulkar      53.8
##  2 R Dravid          52.3
##  3 SM Gavaskar       51.1
##  4 V Sehwag          49.3
##  5 V Kohli           49.2
##  6 VVS Laxman       46.0
##  7 M Azharuddin     45.0
##  8 CA Pujara        43.6
##  9 SC Ganguly       42.2
## 10 DB Vengsarkar    42.1
## 11 GR Viswanath     41.9
## 12 AM Rahane        38.5
## 13 N Kapil Dev      31.0
```

Players with max strike rate

```
## # A tibble: 10 x 2
##    names         strike_rate
##    <chr>               <dbl>
##  1 V Sehwag             82.2
##  2 AC Gilchrist         82.0
##  3 N Kapil Dev          79.3
##  4 DA Warner            70.2
##  5 IVA Richards         69.8
##  6 TM Dilshan           65.5
##  7 ST Jayasuriya        65.2
##  8 BB McCullum          64.6
##  9 KP Pietersen         61.7
## 10 IT Botham            60.7
```

Finding relation between Total Runs and Total Matches

```
runs_matches <- test_matches_clean %>%
  select(matches, total_runs, balls_faced) %>%
  ggplot(mapping = aes(x = total_runs, y = matches,
                      color = -total_runs)) +
  geom_smooth(formula = y ~ x, method = "loess",
              linetype = 'dashed', color = 'navy') +
  geom_point(size = 2.5) +
  labs(title = "Matches vs Total Runs",
       x = 'Runs', y = 'Matches')+
  theme_light() +
  theme(plot.title = element_text(hjust = 0.5, size = 20)) +
  theme(legend.position = "")
print(runs_matches)
```



Players from India with max hundreds

```
## # A tibble: 13 x 2
##    names          hundreds
##    <chr>             <dbl>
##  1 SR Tendulkar         51
##  2 R Dravid             36
##  3 SM Gavaskar          34
##  4 V Kohli              29
##  5 V Sehwag             23
##  6 M Azharuddin         22
##  7 CA Pujara            19
##  8 VVS Laxman           17
##  9 DB Vengsarkar        17
## 10 SC Ganguly           16
## 11 GR Viswanath         14
## 12 AM Rahane            12
## 13 N Kapil Dev           8
```

Player with earliest debut

```
## # A tibble: 5 x 2
##   names        from
##   <chr>       <dbl>
## 1 JB Hobbs     1908
## 2 WR Hammond   1927
## 3 DG Bradman   1928
## 4 L Hutton     1937
## 5 DCS Compton  1937
```

Players with long career

```
test_matches_clean %>%
  select(names, from, to) %>%
  mutate(played_years = test_matches_clean$to - test_matches_clean$from) %>%
  arrange(desc(played_years))
```

```
## # A tibble: 107 x 4
##    names           from    to played_years
##    <chr>          <dbl> <dbl>        <dbl>
##  1 SR Tendulkar   1989  2013           24
##  2 JB Hobbs       1908  1930           22
##  3 S Chanderpaul  1994  2015           21
##  4 MC Cowdrey     1954  1975           21
##  5 GA Gooch       1975  1995           20
##  6 GS Sobers      1954  1974           20
##  7 WR Hammond     1927  1947           20
##  8 DG Bradman     1928  1948           20
##  9 DCS Compton    1937  1957           20
## 10 SR Waugh       1985  2004           19
## # i 97 more rows
```

We have solved my questions with the dataset and can solve even more.