We are back with another post on the [Indian Premier League](https://www.iplt20.com/). This is the fourth post in the series. We will assume that you have already read the previous article analyzing strike rates [here](https://www.radmuzom.com/2019/04/06/analysing-strike-rates-in-the-ipl-using-the-tidyverse). One change since the last article is that [Cricsheet](https://cricsheet.org/) now has updated data available – so we have the details of all matches played up to 2019.

The initial part of the code is very similar to the previous article. We will not explain the details here, but the code has been repeated below.

batsmen\_striker <- match\_deliveries %>%

distinct(delivery\_batsman)

nrow(batsmen\_striker)

## [1] 514

batsmen\_matches <- match\_deliveries %>%

distinct(delivery\_batsman, id) %>%

group\_by(delivery\_batsman) %>%

summarise(n\_matches = n()) %>%

ungroup() %>%

arrange(desc(n\_matches))

summary(batsmen\_matches$n\_matches)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 1.00 3.00 8.00 21.97 23.00 189.00

batsmen\_matches <- batsmen\_matches %>%

filter(n\_matches >= 20)

batsmen\_match\_details <- match\_deliveries %>%

inner\_join(batsmen\_matches, by = c("delivery\_batsman" = "delivery\_batsman"))

table(batsmen\_match\_details$delivery\_extras\_type)

##

## byes legbyes noballs penalty wides

## 375 2546 608 2 4823

noballs <- batsmen\_match\_details %>%

filter(delivery\_extras\_type == "noballs", delivery\_runs\_total > 1)

byes <- batsmen\_match\_details %>%

filter(delivery\_extras\_type == "byes", delivery\_runs\_batsman > 1)

legbyes <- batsmen\_match\_details %>%

filter(delivery\_extras\_type == "legbyes", delivery\_runs\_batsman > 1)

wides <- batsmen\_match\_details %>%

filter(delivery\_extras\_type == "wides", delivery\_runs\_batsman > 1)

penalty <- batsmen\_match\_details %>%

filter(delivery\_extras\_type == "penalty")

batsmen\_match\_details <- batsmen\_match\_details %>%

filter(!(delivery\_extras\_type %in% c("wides", "penalty")))

At the end of the code above, the dataset *batsmen\_match\_details* is restricted to batsmen who have played at least 20 matches. We have also removed the deliveries which don’t contribute to runs scored by a batsman.

Anyone who follows the game of cricket knows that T20 cricket is primarily a batsman’s game. The crowd loves it when the batsmen send the ball into the stands. More elegant players might not use a lot of power in their shots, but may gracefully place it in the gaps for four runs. In this article, we examine the percentage of runs scored boundaries (sixes and fours) by top batsmen in this tournament.

We need to be careful about one particular aspect of the data. Even though four or six runs may be scored off a ball, they are not necessarily to be counted as boundaries. The dataset has a field called *delivery\_runs\_non\_boundary* to identify such cases.

table(batsmen\_match\_details$delivery\_runs\_non\_boundary)

##

## 0 1

## 153590 12

batsmen\_match\_details$delivery\_runs\_batsman[

batsmen\_match\_details$delivery\_runs\_non\_boundary == 1]

## [1] 4 4 4 6 4 4 4 6 4 6 6 4

Looking at the above results, there are only 12 such cases. In all such cases, the runs scored by a batsman in that ball was 4 or 6. For each delivery, we use the *case\_when* function from *dplyr* to derive two fields – *runs\_boundary* and *runs\_non\_boundary*. We also verify that the total runs scored by the batsman in the delivery adds up to the sum of these two fields.

batsmen\_match\_runs <- batsmen\_match\_details %>%

mutate(

runs\_boundary = case\_when(

delivery\_runs\_non\_boundary == 0 &

(delivery\_runs\_batsman %in% c(4, 6)) ~ delivery\_runs\_batsman,

TRUE ~ 0L

),

runs\_non\_boundary = case\_when(

delivery\_runs\_non\_boundary == 1 |

!(delivery\_runs\_batsman %in% c(4, 6)) ~ delivery\_runs\_batsman,

TRUE ~ 0L

)

) %>%

select(id, innings\_num, delivery\_batsman, delivery\_runs\_batsman,

runs\_boundary, runs\_non\_boundary)

nrow(subset(

batsmen\_match\_runs,

delivery\_runs\_batsman != runs\_boundary + runs\_non\_boundary

))

## [1] 0

We first summarise the data at the batsman level. We also verify that the total number of runs scored by a batsman in their IPL career adds up to the runs scored in boundaries and non-boundaries.

batsmen\_runs\_summary <- batsmen\_match\_runs %>%

group\_by(delivery\_batsman) %>%

summarise(

n\_balls = n(),

delivery\_runs\_batsman = sum(delivery\_runs\_batsman),

runs\_boundary = sum(runs\_boundary),

runs\_non\_boundary = sum(runs\_non\_boundary)

) %>%

ungroup() %>%

arrange(desc(delivery\_runs\_batsman))

nrow(subset(

batsmen\_runs\_summary,

delivery\_runs\_batsman != runs\_boundary + runs\_non\_boundary

))

## [1] 0

As there are 151 unique batsmen in the data which is too much to deal with, we restrict the data to those who have scored at least 1000 runs in their IPL career. After this step, we are left with 70 batsmen.

batsmen\_runs\_summary <- batsmen\_runs\_summary %>%

filter(delivery\_runs\_batsman > 1000)

nrow(batsmen\_runs\_summary)

## [1] 70

We now derive two fields in the data. *runs\_boundary\_pct* is the percentage of runs scored as boundaries in their IPL career. *career\_strike\_rate* is the strike rate of the batsman in their IPL career.

batsmen\_runs\_summary <- batsmen\_runs\_summary %>%

mutate(runs\_boundary\_pct = runs\_boundary / delivery\_runs\_batsman,

career\_strike\_rate = delivery\_runs\_batsman / n\_balls) %>%

arrange(desc(runs\_boundary\_pct))

summary(batsmen\_runs\_summary$runs\_boundary\_pct)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.4820 0.5508 0.5906 0.5975 0.6397 0.7886

The summary statistics show that of all batsmen who have played at least 20 matches and scored 1000 runs, no one has scored less than 48% of their runs in boundaries. There is one (or more) batsmen who have scored close to 80% of their runs in boundaries. We’ll soon see who they are.

A reasonable hypothesis is that those who have a large percentage of runs from boundaries will also have higher career strike rates. Let us take a look at a plot to see if this is indeed the case.

ggplot(batsmen\_runs\_summary,

aes(runs\_boundary\_pct, career\_strike\_rate)) +

geom\_point(colour = "blue") +

geom\_smooth(method = "lm", se = FALSE, alpha = 0.1, colour = "black") +

scale\_x\_continuous(labels = scales::percent) +

scale\_y\_continuous(labels = scales::percent) +

xlab("% Runs (Boundaries)") +

ylab("IPL Career Strike Rate") +

theme\_bw() +

theme(

panel.border = element\_blank(),

legend.position = "none",

plot.title = element\_text(size = 20, hjust = 0.5, vjust = 0.5),

axis.title = element\_text(size = 10),

axis.text = element\_text(size = 10)

)

The hypothesis appears to be true. The linear regression line, as produced by *geom\_smooth* and the *lm* method has a positive slope. There seems to be one outlier in the data – someone with a strike rate close to 190% and around 80% of their runs from boundaries. Can you guess who that is? We’ll see soon.

We finally plot the data for each of the batsmen. We use a dot plot to visualize the results.

ggplot(batsmen\_runs\_summary,

aes(reorder(delivery\_batsman, runs\_boundary\_pct),

runs\_boundary\_pct)) +

geom\_point(colour = "blue") +

scale\_y\_continuous(labels = scales::percent) +

xlab(NULL) +

ylab("% Runs (Boundaries)") +

theme\_bw() +

theme(

panel.border = element\_blank(),

legend.position = "none",

plot.title = element\_text(size = 20, hjust = 0.5, vjust = 0.5),

axis.title = element\_text(size = 10),

axis.text = element\_text(size = 10)

) +

coord\_flip()

Most people would have guessed Chris Gayle or Andre Russell for the outlier in the previous figure. I believe Russell’s recent performance in the 2019 IPL has ensured that he occupies the top spot as of the date of writing this article. I was a bit surprised to see that five Australians appear in the top 10 – Gilchrist, Maxwell, Lynn, Watson and Hayden. Considering that Dwayne Smith also appears in the top 10, the domination by the Aussies and the West Indians in this list appears complete.