lab_02

2023-09-06

Preliminaries

Here, we are reading the CSV:

```
cal_fb <- read.csv("./data/cal_fb.csv")
```

Here, we remove the first column of the dataframe above. Then, we write a function that calculates Cal's total score and the opponent's total score from a single row of scores.

```
quarter_scores <- cal_fb[, -1]
sum_scores <- function(x) {
  row_scores <- x
  cal_scores <- sum(row_scores[1:4])
  opponent_scores <- sum(x) - cal_scores
  return(c(cal_scores, opponent_scores))
}
```

We check if our function returns the correct values for the first row:

```
## [1] 27 13
```

Now, we want to create a new dataframe that shows the final scores against all the opponents.

```
first_column <- cal_fb[ ,1]
#apply presents the results as two rows, so we have to flip the rows and columns
second_and_third_columns <- t(apply(quarter_scores, 1, sum_scores))

#storing the final scores as a data frame
final_scores <-as.data.frame(cbind(first_column, second_and_third_columns))
```

I used `dplyr` to change the column names, but have hidden the code for the PDF file. Please look at the `markdown` file for more details.

Is Cal a Loser

Here, we write a function that checks if Cal football loses against its opponent. We don't account for draws in this function, but I guess it's fine since none of the scores are a draw anyway.

```
did_cal_lose <- function(x) {
  #Not sure why I had to put it in an if-else statement
  #Previously wrote as return(x[2] < x[3]), but then it would also print
  #Cal_score row number for some reason
  #have to change the results from string to integer, or it will compare index by index
  if (strtoi(x[2]) < strtoi(x[3])) {
    return(TRUE)
  }
  else {
    return(FALSE)
```

```
  }
}
```

Checking to see if it returns the right result, for row 6:

```
## [1] TRUE
```

Carlos is Nosy

Now, we log the results that Carlos is looking for:

```
results <- apply(final_scores, 1, did_cal_lose)
final_scores_with_results <- cbind(final_scores, results)

cal_wins <- sum(final_scores_with_results[,4] == FALSE)
cal_losses <- sum(final_scores_with_results[,4] ==TRUE)
```

Let's check if the wins and losses are correct, where the first number is the number of wins and the second one is the number of losses:

```
## [1] 8
```

```
## [1] 5
```

Head or Tails (Sonic likes Tails)

Here, we write a function that tosses a coin once as well another function that tosses the coin some set amount that we want, and then stores the results.

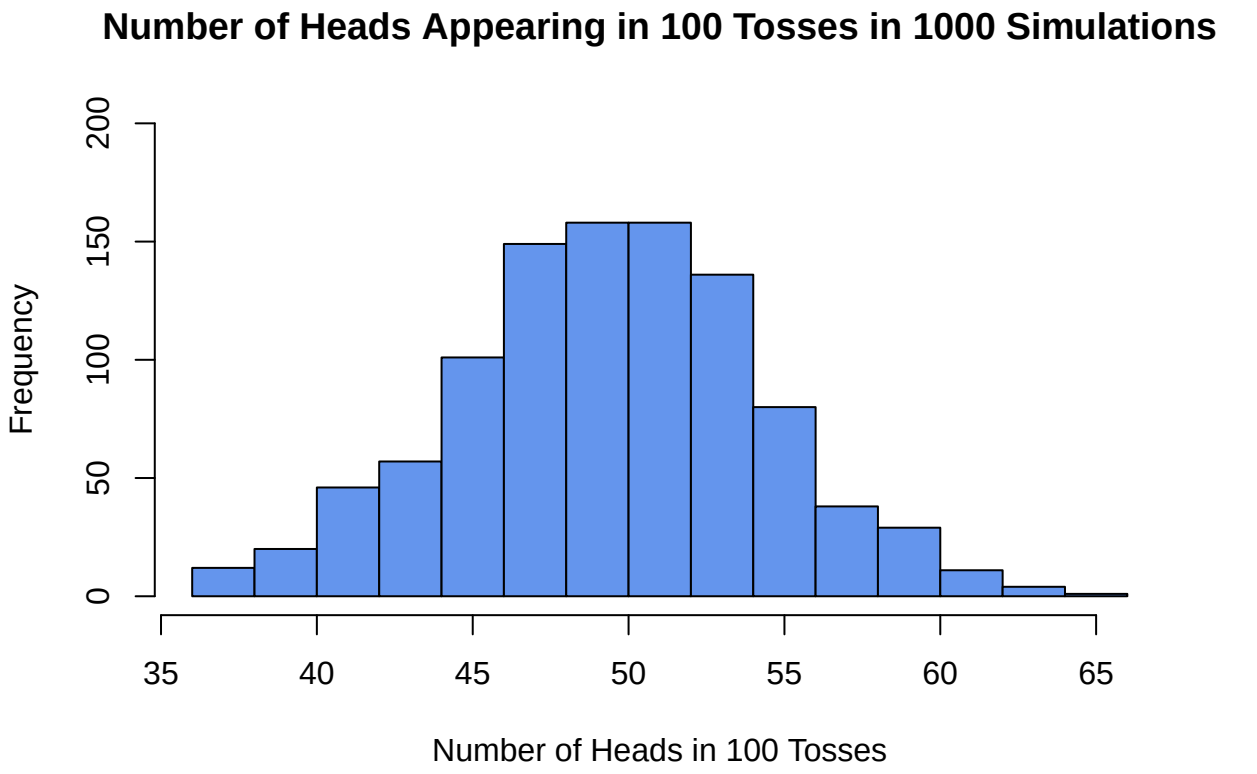
```
#tosses the coin once
toss <- function(x) {
  sample(c("H", "T"), size = 1)
}

#tosses the coin multiple times, according to the argument supplied
toss_multiple <- function(x) {
  store <- numeric(x)
  for (i in 1:x) {
    store[i] <- toss()
  }
  return(store)
}
```

Here, we have written a function that counts the number of heads in a given number of coin tosses.

```
count_heads <- function(x){
  sum(grepl("H", toss_multiple(x)))
}
```

Now, we toss the coin 100 times in 1000 simulations by using the `sapply`, and obtain the required histogram.



We see that the most frequent number of heads appearing is around 50 or so, which makes sense, since there's a 50% chance of heads appearing.