Workable Coin Toss Puzzle Solution

Konstantinos Papastamos

6 June 2016

The Puzzle:

Suppose we have 4 coins, each with a different probability of throwing Heads. An unseen hand chooses a coin at random and flips it 50 times. This experiment is done several times with the resulting sequences as shown below (H = Heads, T = Tails) Write a program that will take as input the data that is collected from this experiment and estimate the probability of heads for each coin.

My Solution:

I did one change before I load the data, I deleted the blank lines in the original dataset, I will attach my data.txt with the e-mail. It also can be found in my github repository in this link: https://github.com/Costaspap/Workable_Coin_Toss_Puzzle

Important: Make sure that the data.txt file is in R's current working directory, found with the getwd() command.

So in order to get started, we first load the data. We don't have headers and we don't want strings to be stored as factors, so:

```
data = read.table("data.txt", header = F, strin = F)
names(data) = "Tosses"
```

Lets make sure every row has 50 characters.

```
lapply(data,nchar)
```

```
## $Tosses
##  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [24] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [47] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
## [70] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
```

We only want to keep the head occurences.

```
heads = sapply(data,function(x) gsub("T","",x))
```

Then we find the number of heads.

```
nheads = apply(heads,2,nchar)
```

Find the head propability for each experiment.

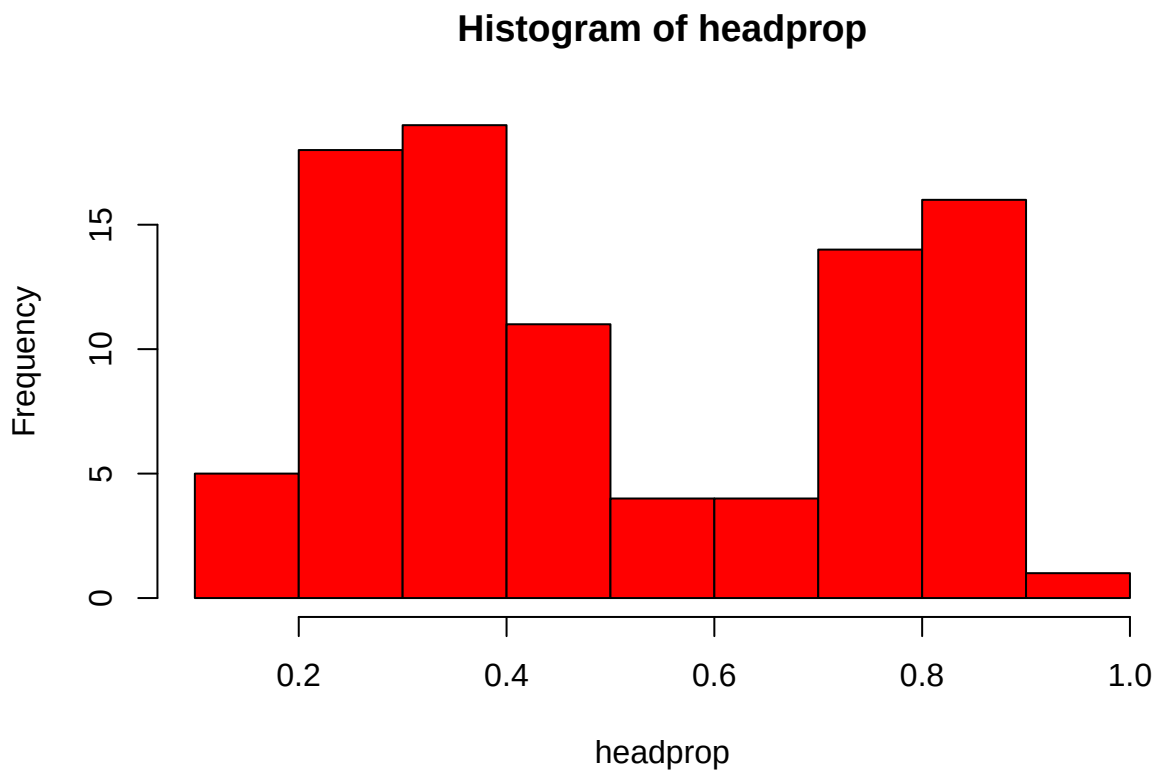
```
headprop = nheads/50
```

Let's make them look nice

```
headprop = sort(headprop)
```

Now let's take a look at our data:

```
hist(headprop,col="red")
```



I am going to use k-means clustering. I chose the initial centers based on the histogram.

```
k=kmeans(headprop,c(0.23,0.35,0.48,0.8),iter.max=10)
k
```

```
## K-means clustering with 4 clusters of sizes 18, 25, 15, 34
##
## Cluster means:
##        [,1]
## 1 0.2388889
## 2 0.3472000
## 3 0.5026667
## 4 0.8058824
##
## Clustering vector:
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [36] 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4
## [71] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
##
```

```
## Within cluster sum of squares by cluster:
## [1] 0.02997778 0.02950400 0.03189333 0.12922353
##  (between_SS / total_SS =  95.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

==========Results==========

So my best estimates for each coin's head propability are:

Coin 1: 0.2388889 Coin 2: 0.3472000 Coin 3: 0.5026667 Coin 4: 0.8058824

After experimenting with many different centers, those gave me the highest reduction in sum of squares, 95,7%