HomeWork3

Ivan Ojiambo

October 5, 2017

Exe1

```
sample_n <- function(N, probs) {
   count <- 0
   for(i in 1:100){
      sample <- table(sample(c(1,2,3,4), N, replace = T, prob = probs))
      sample_prob <- prop.table(sample)
      error <- as.vector(abs(probs - sample_prob))
      if(all(error < 0.005)){
        count <- count+1
      }
   }

 return(count);
}
```

a

I first tried out with sample size N = 100 and I got a percentage range of 0 to 1

```
prob <-  c(0.1, 0.2, 0.3,0.4)
sample_n(100, prob)
```

```
## [1] 0
```

I then tried out with sample size N = 500 and the ouput also resulted in a percentage range of 0 to 2

```
sample_n(500, prob)
```

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

I now increased the sample size N to 4000 and I got a percentage range of 12 to 25

```
sample_n(4000, prob)
```

```
## [1] 12
```

After three atemptsof with a small sample size it was clear to me that the sample size was very large and so i tried out N=30000. The results showed a percentage range of 80 to 92%.

```
sample_n(30000, prob)
```

```
## [1] 83
```

I finally tried out with sample size N = 51000 and it resulted in percentage range of 95 to 97%.

decided to take the sample size as 51000 as my final sample size since it yielded consistent results of over 95%

```
sample_n(51000, prob)
```

```
## [1] 94
```

b

with sample size N =4000, it resulted in a percentage range of 5 to 20

```
prob2 <-  c(0.25, 0.25, 0.25, 0.25)
sample_n(4000, prob2)
```

```
## [1] 17
```

It so happend that with this distribution, with sample size N=49000 ,the percetage was was over 95%
Therefore the sample size in the second distribution is 49000

```
sample_n(50000, prob2)
```

```
## [1] 98
```

In the first distrubtion ie (0.1, 0.2, 0.3, 0.4), the smaple size is high compared to the second distribution (0.25, 0.25, 0.25,0.25)

Exe2

```
sample_n2 <- function(N, prob){

  count <- 0
  for(i in 1:100){
    error <-  (rnorm(4, mean = 0, sd = sqrt(prob*N)))/N
    if(all(error < 0.005 & error >-0.005)){
        count <- count+1
      }

  }

return(count)
}
```

a (0.1, 0.2, 0.3, 0.4)

It appears that with a sample size of N= 45000, it generetes a percentage with the range of 80- 90%

```
prob <- c(0.1, 0.2, 0.3, 0.4)
sample_n2(45000, prob)
```

```
## [1] 82
```

I decided to increase to the samplesize to N= 69000. it generates the percentage range of over 95%.

```
sample_n2(69000, prob)
```

```
## [1] 96
```

The sample size is 69000 and its clear that using the statistical approach the sample size will be higher than that with computational approach.

b

with the second distribution the the percentage is above 95% when N=64000

```
prob <- c(0.25, 0.25, 0.25, 0.25)
sample_n2(64000, prob)
```

```
## [1] 98
```

with a same sample size as that in the computation method, the statistical approach takes less time to generate the percentages, infact less than a second compared to the computation approach.

Exe3

- I assumed column X was auto generated from the database and it did not have any meaning to the data set so I had to remove it
- I assumed that the if there exists empty values , I perform pairwise deletion.
- Am calculating correlation using the spearman's correlation.
- I first calculated the corelation ceficient for the whole dataset
- I then selected the upper triangle for the coefficient matrix
- I calculated the min and max correlation

_I again calculated the min and max correlation and used them as threshold to determin those correlation that occured by chance.

```
data <- read.csv(file = "exe3.csv", header = TRUE, sep = ",")
data$X <- NULL
cor_matrix <- cor(data, use = "pairwise.complete.obs", method = "spearman");
cor_matrix[lower.tri(cor_matrix, diag=TRUE)] <- NA
min(cor_matrix , na.rm = TRUE)
```

```
## [1] -0.3967837
```

```
max(cor_matrix, na.rm = TRUE)
```

```
## [1] 0.9881308
```

- I then shuffled the dataframe and got the correlation coffecient

```
shuffled_data <- data
for(i in 1:100){
  shuffled_data[,i] <- shuffled_data[sample(nrow(shuffled_data)),i]

}
corr_shuffled_data <- cor(shuffled_data)
corr_shuffled_data[lower.tri(corr_shuffled_data, diag = TRUE)] <- NA
min_cor <- min(corr_shuffled_data, na.rm = TRUE)
max_cor <- max(corr_shuffled_data, na.rm = TRUE)
```

They were roughly 15 columns that were correlated

```
sum(cor_matrix >max_cor | cor_matrix <min_cor, na.rm = TRUE)
```

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

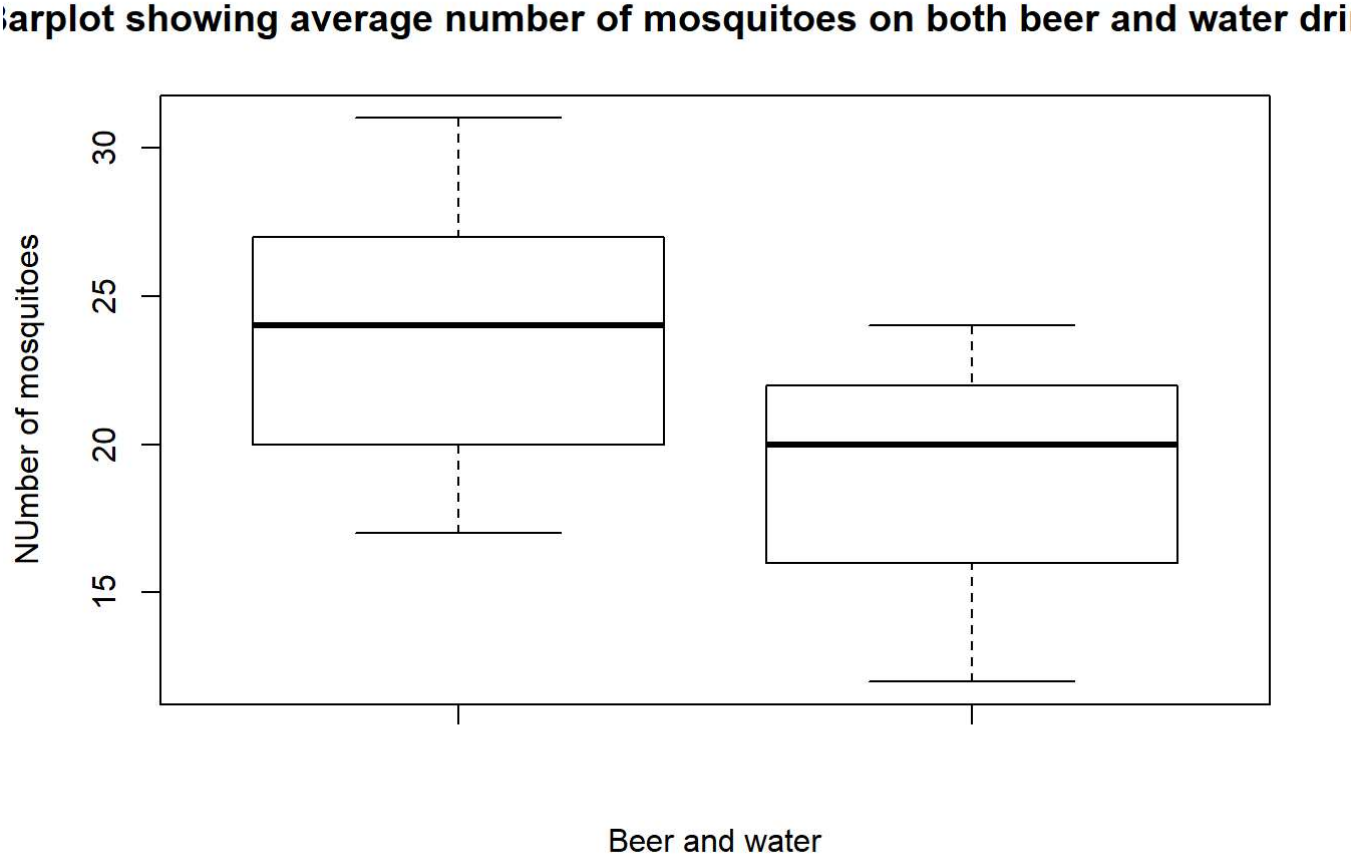
Exe4

t-test

```
beer <- c(27, 19, 20, 20, 23, 17, 21, 24, 31, 26, 28, 20, 27, 19, 25, 31, 24, 28, 24, 29, 21,
 21, 18, 27, 20)
water <- c(21, 19, 13, 22, 15, 22, 15, 22, 20, 12, 24, 24, 21, 19, 18, 16, 23, 20)
mean_beer <- mean(beer)
mean_water <- mean(water)
mean_diffs <- mean_beer - mean_water
mean_diffs
```

```
## [1] 4.377778
```

```
boxplot(beer,water, main="Barplot showing average number of mosquitoes on both beer and water
 drinkers", xlab="Beer and water", ylab="NUmber of mosquitoes")
```



Null hypothesis

The difference between the average number of mosquitoes on beer drinkers and average number of mosquitoes on people who drunk water is not equal to 4.377

```
ie H0:mean != 4.37
```

Alternative Hyp:

The difference between the average number of mosquitoes on beer drinkers and average number of mosquitoes on people who drunk water is equal to 4.377

```
ie H1:mean difference = 4.37
```

```
t.test(beer, water)
```

```
##
##  Welch Two Sample t-test
##
## data:  beer and water
## t = 3.6582, df = 39.113, p-value = 0.0007474
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.957472 6.798084
## sample estimates:
## mean of x mean of y
##  23.60000  19.22222
```

From the t-test value, p=0.00074 which is less than 0.005 therefore we neget the null hypothesis and conclude that the difference in the average number of mosquitoes for both beer and water drinkers is significantly equal to 4.37

using permutation test

Ho: differennce in mean of mosquito on beer and waters drinkers is not equal to 4.37

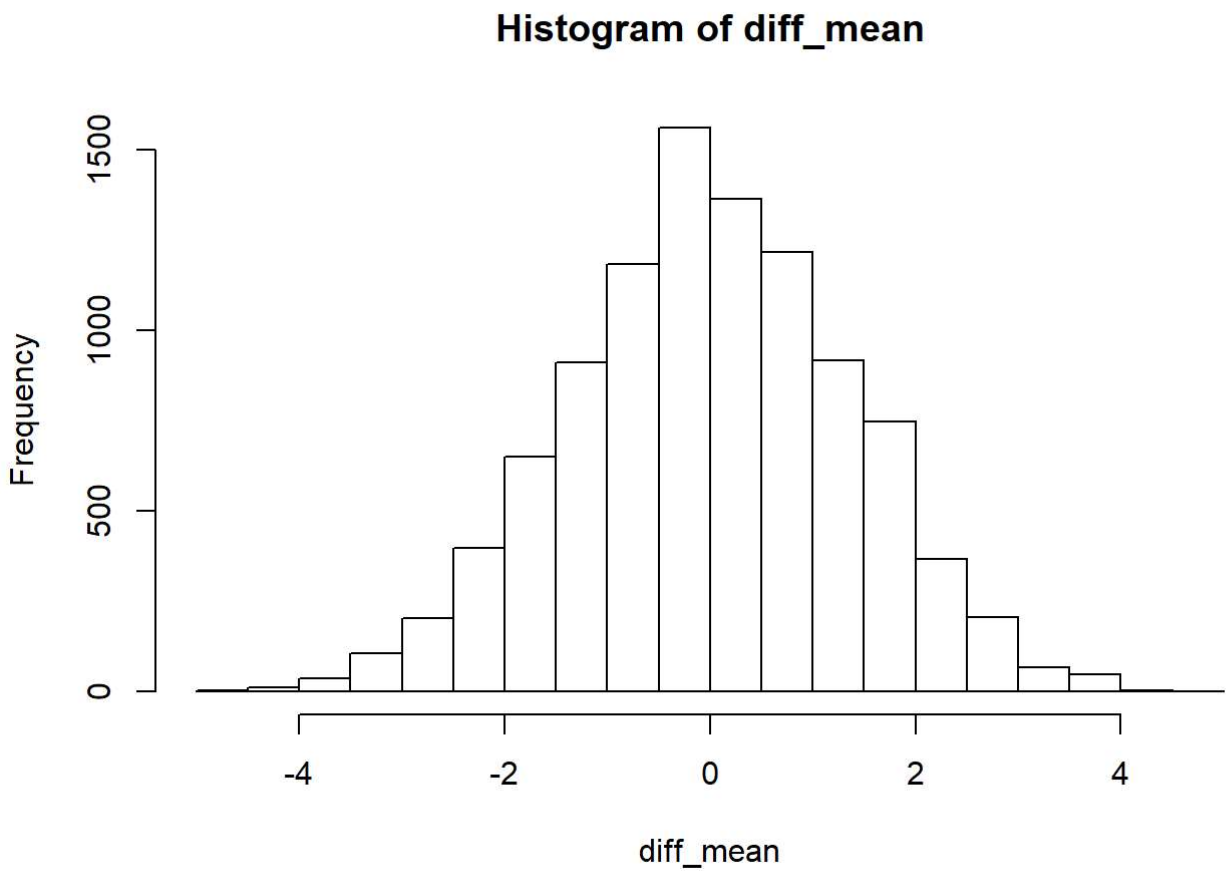
H1:The differennce in mean of beer drinkers and waters =4.37

```
combin_beer_water <- c(beer,water)

nsimulation <- 10000
diff_mean <- rep(NA, nsimulation)

for(i  in 1:nsimulation){
 shuffled_data <-  sample(combin_beer_water)
 mean_beer <- mean(shuffled_data[1:length(beer)])
 mean_water <- mean(shuffled_data[(length(beer)+1):length(shuffled_data)])
 diff_mean[i] <- mean_beer - mean_water

}
hist.default(diff_mean)
```




```
p_value <- sum(diff_mean == 4.377)/nsimulation
p_value
```

```
## [1] 0
```

From the histogram and p= 0, its clear that mean of 4.37 is less likely to occur by chance and therefore we negate the null hypothesis and conclude that the mean difference is signficantly equal to 4.37