Homework3

Denizalp KAPISIZ

Exercise 1

We can specify a range for N as $N > 60000$.

Here is the sampling function to try the distributions and sample size.

```
sample_n1 <- function(N, probs) {

  trials = c(rep(F,100))
  for(index in 1:100){
  mySample = sample(c(1,2,3,4), N, replace = T, prob = probs)
  distribution = table(mySample) / N
  errors = abs(distribution-probs)
  trials[index] = (length(which(errors<0.005)) == 4)
  }
  numberOfTrue = length(which(trials==TRUE))
  return(numberOfTrue)

}
```

Case (10%, 20%, 30% and 40%)

```
sample_n1(1000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 2
```

```
sample_n1(10000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 36
```

```
sample_n1(20000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 74
```

```
sample_n1(30000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 85
```

```
sample_n1(40000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 95
```

```
sample_n1(50000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 97
```

```
sample_n1(60000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 99
```

Case (25%, 25%, 25% and 25%)

```
sample_n1(1000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 2
```

```
sample_n1(10000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 42
```

```
sample_n1(20000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 67
```

```
sample_n1(30000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 80
```

```
sample_n1(40000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 90
```

```
sample_n1(50000,c(0.25,0.25,0.25,0.25))
```

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

```
sample_n1(60000,c(0.25,0.25,0.25,0.25))
```

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

There is no difference between the two different true distributions.

Exercise 2

We can specify a range for N as N $>$ 80000.

Here is the sampling function to try the distributions and sample size.

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

  trials = c(rep(F,100))
  for(index in 1:100){
    sDev = sqrt(probs*N)
    errors = abs(rnorm(4, mean = 0, sd = sDev) / N)
    trials[index] = (length(which(errors<0.005)) == 4)
  }
  numberOfTrue = length(which(trials==TRUE))
  return(numberOfTrue)

}
```

Case (10%, 20%, 30% and 40%)

```
sample_n2(1000,c(0.1,0.2,0.3,0.4))
```

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

```
sample_n2(10000,c(0.1,0.2,0.3,0.4))
```

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

```
sample_n2(20000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 52
```

```
sample_n2(30000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 70
```

```
sample_n2(40000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 76
```

```
sample_n2(50000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 88
```

```
sample_n2(60000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 87
```

```
sample_n2(70000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 97
```

```
sample_n2(80000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 95
```

Case (25%, 25%, 25% and 25%)

```
sample_n2(1000,c(0.25,0.25,0.25,0.25))
```

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

```
sample_n2(10000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 20
```

```
sample_n2(20000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 55
```

```
sample_n2(30000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 67
```

```
sample_n2(40000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 86
```

```
sample_n2(50000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 95
```

```
sample_n2(60000,c(0.25,0.25,0.25,0.25))
```

```
## [1] 92
```

```
sample_n2(70000,c(0.25,0.25,0.25,0.25))
```

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

```
sample_n2(80000,c(0.25,0.25,0.25,0.25))
```

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

There is no difference between two different true distributions. For the sample size, we can claim that EX1 requires lower sample size for confidence than EX2. For the speed, we can use tictoc library of R to calculate the times for the functions in EX1 and EX2.

```
tic("EX1 Sample:10000")
sample_n1(10000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 34
```

```
toc()
```

```
## EX1 Sample:10000: 0.788 sec elapsed
```

```
tic("EX1 Sample:30000")
sample_n1(30000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 88
```

```
toc()
```

```
## EX1 Sample:30000: 2.117 sec elapsed
```

```
tic("EX2 Sample:10000")
sample_n2(10000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 20
```

```
toc()
```

```
## EX2 Sample:10000: 0.003 sec elapsed
```

```
tic("EX2 Sample:30000")
sample_n2(30000,c(0.1,0.2,0.3,0.4))
```

```
## [1] 64
```

```
toc()
```

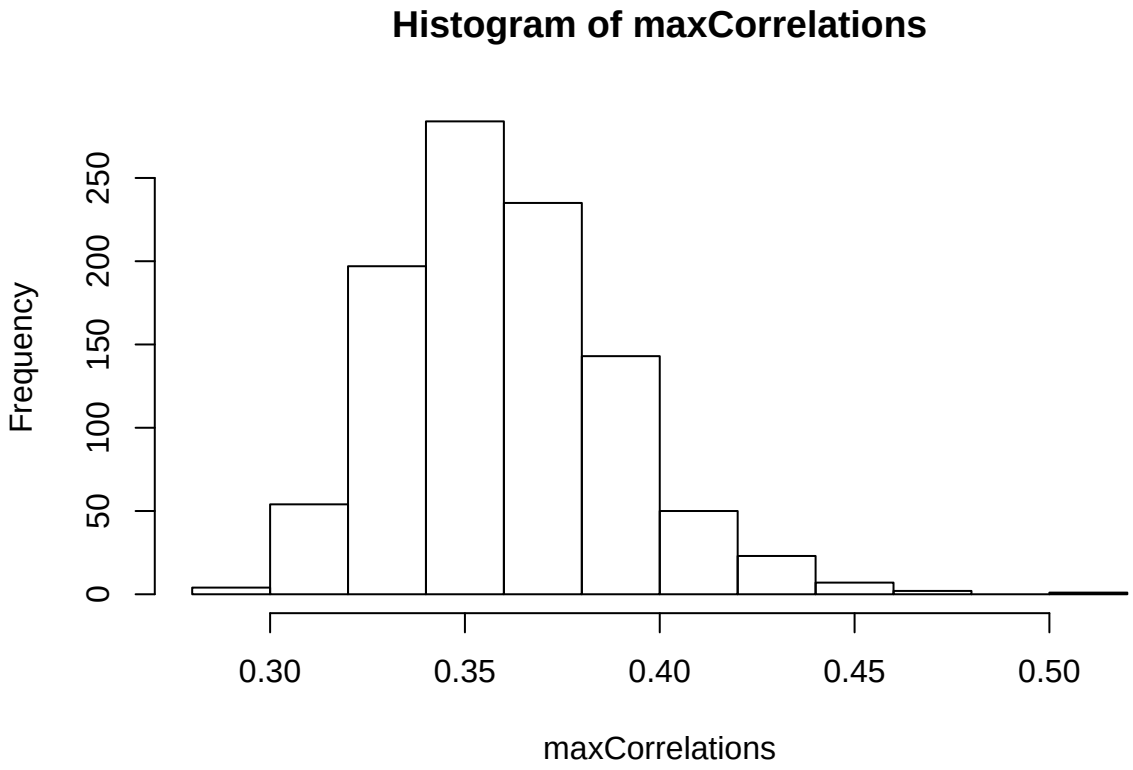
```
## EX2 Sample:30000: 0.003 sec elapsed
```

Here, we can argue that EX2 works faster than EX1.

Exercise 3

First of all, we can follow the null hypothesis that correlations in the data are random. In order to test this, we keep the maximum and minimum correlation in the original data as the test points. Then, we shuffle each column independently to observe maximum and minimum correlations. Distribution of these extremes will provide an evidence about how likely the correlations in the original data.

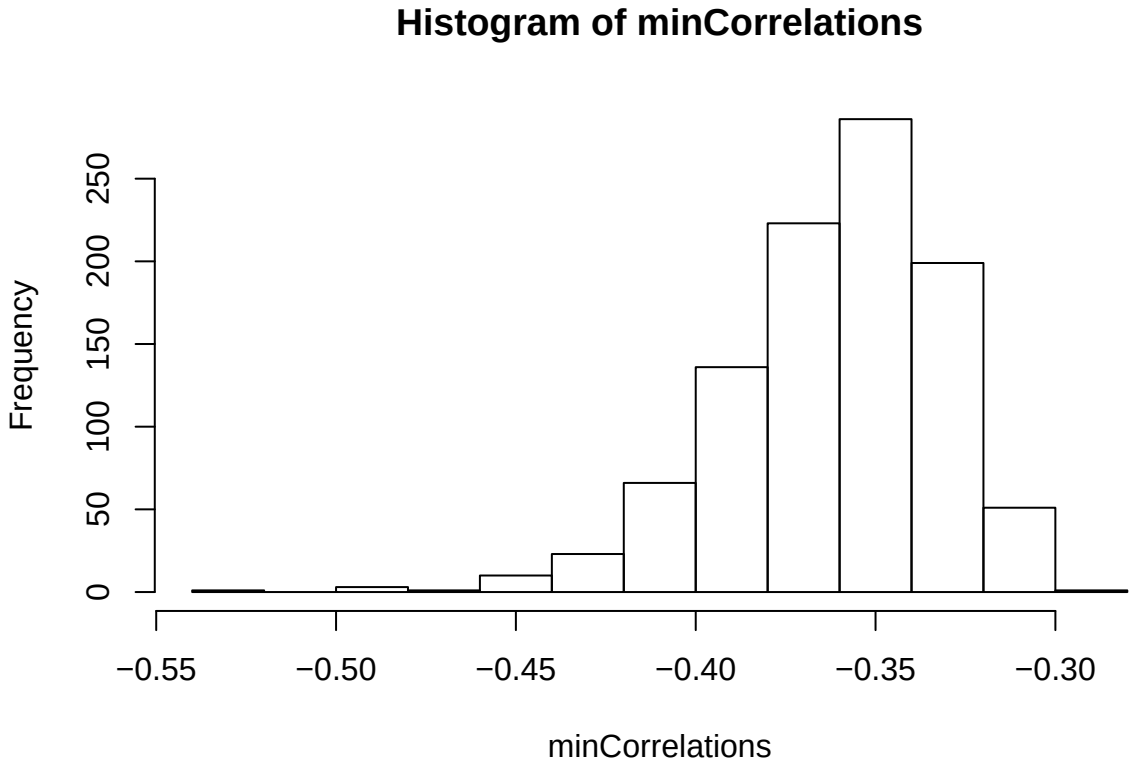
```
hist(maxCorrelations)
```




```
testMax
```

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

```
hist(minCorrelations)
```



```
testMin
```

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

Since $0.117 \geq 0.05$, we cannot reject that negative correlations are random. On the other hand, $0 < 0.05$ impiles that we can reject that positive correlations are random; therefore, we claim that pairs having correlations greater than the maximum of positive random correlations are correlated. There are 10 correlated pairs in the original data.

Exercise 4

T-test approach

```
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
```

Since p-value < 0.05, we can claim that there is a statistically significant difference between means of beer and water.

Permutation test

Here, we assumed that the difference between means of beer and water has no significant point. Therefore, mixing these values would not yield significant result. In order to apply this, we picked 10 random numbers from 1 to 18, then we replaced the values from beer to water and vice versa. We repeated this by 50000 times.

```
testMeanDiff = mean(beer) - mean(water)
meanDiffs = c(rep(0,50000))
for(r in 1:50000){
  place = sample(1:18,10)
  fromBeer = beer[place]
  fromWater = water[place]
  beer[place] = fromWater
  water[place] = fromBeer
  meanDiffs[r] = mean(beer) - mean(water)
}
pvalue = length(which(meanDiffs >= testMeanDiff)) / 50000
pvalue
```

```
## [1] 0.00168
```

By our p-value, we can understand that our assumption is unlikely; thus, we reject that the difference between means of beer and water has no significant point.

Talk

T-test is the same as that in the talk. The approach of mixing values is different because we select randomly the places before; on the other hand, in the talk, he randomly chooses from both sides then distributes those randomly. However, the histogram is similar to that in the talk.

```
hist(meanDiffs)
```