INTRODUCTION TO THE PRACTICE OF STATISTICS

Natalia Bercz

05/05/2020

Report 3

Assignment 1

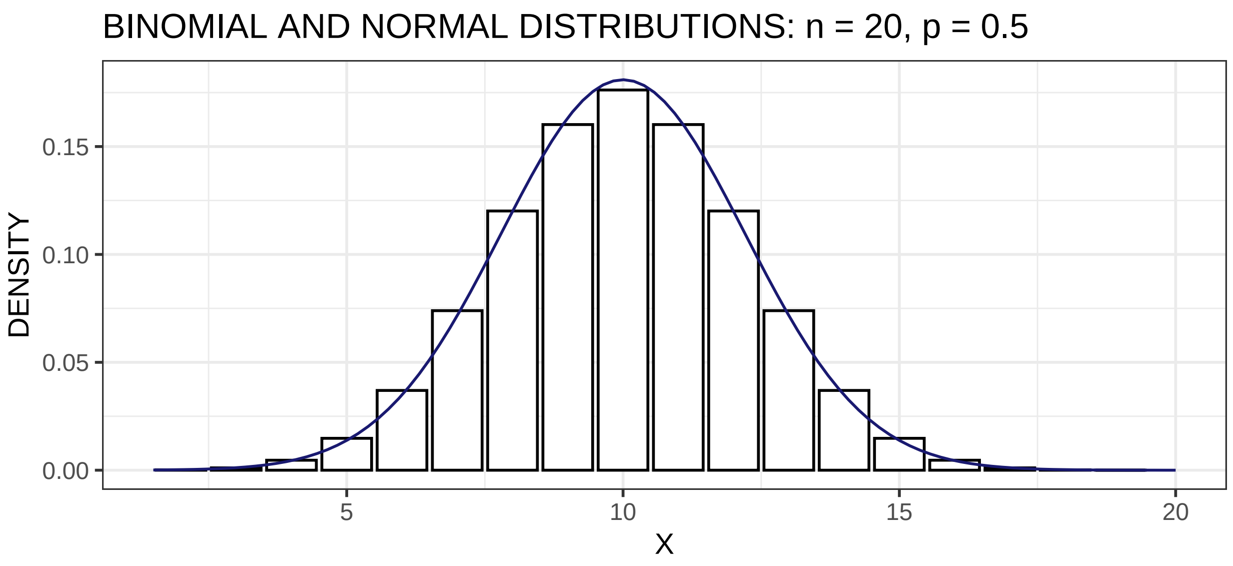
```
x20  = 1:20
x100 = 1:100
```

a)

```
y20_p05 = dbinom(x20, 20, 0.5)
dbin20_p05 = data.frame(y20_p05)
mean = mean(y20_p05)
sd = sd(y20_p05)
```

A histogram of the binomial distribution and the graph of the density of the normal distribution, p = 0.5 i n = 20.

```
ggplot(dbin20_p05, aes(x20, y20_p05))+
  geom_histogram(stat = 'identity', color = 'black', fill = 'white')+
  stat_function(fun = dnorm, args = list(mean, sd), color = 'midnightblue')+
  labs(title = 'BINOMIAL AND NORMAL DISTRIBUTIONS: n = 20, p = 0.5')+
  xlim(1.5, 20)+xlab('X')+ylab('DENSITY')+theme_bw()
```

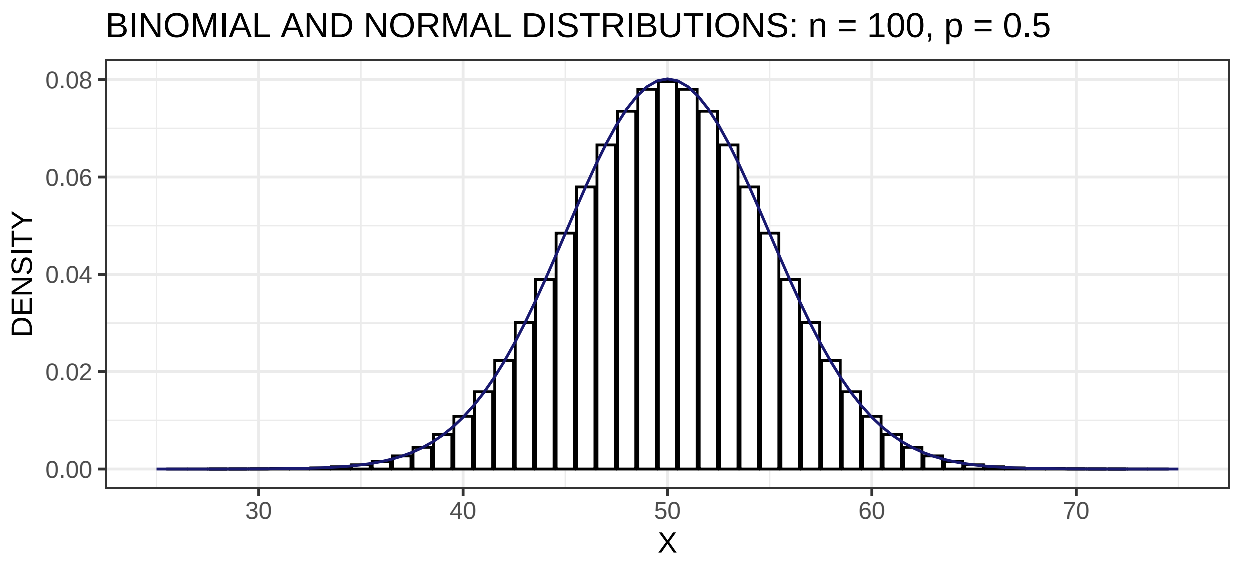


b)

```
y100_p05 = dbinom(x100, 100, 0.5)
dbin100_p05 = data.frame(y100_p05)
mean = mean(y100_p05)
sd = sd(y100_p05)
```

A histogram of the binomial distribution and the graph of the density of the normal distribution, p = 0.5 i n = 100.

```
ggplot(dbin100_p05, aes(x100, y100_p05))+
  geom_histogram(stat = 'identity', color = 'black', fill = 'white')+
  stat_function(fun = dnorm, args = list(mean, sd), color = 'midnightblue')+
  labs(title = 'BINOMIAL AND NORMAL DISTRIBUTIONS: n = 100, p = 0.5')+
  xlim(25, 75)+xlab('X')+ylab('DENSITY')+theme_bw()
```

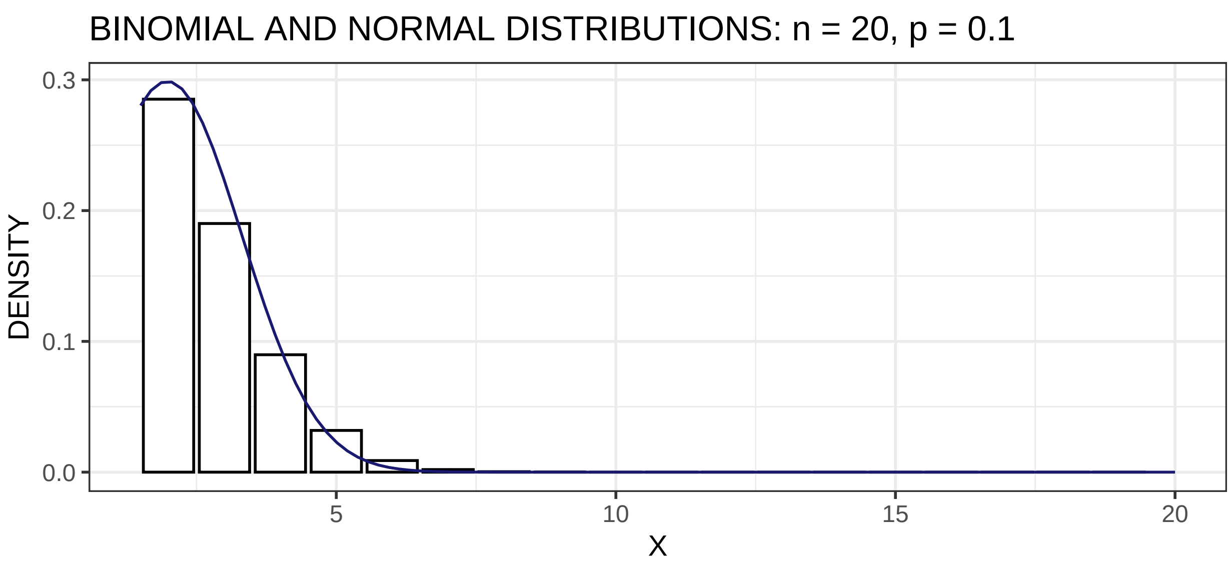


c)

```
y20_p01 = dbinom(x20, 20, 0.1)
dbin20_p01 = data.frame(y20_p01)
mean = mean(y20_p01)
sd = sd(y20_p01)
```

A histogram of the binomial distribution and the graph of the density of the normal distribution, p = 0.1 i n = 20.

```
ggplot(dbin20_p01, aes(x20, y20_p01))+
  geom_histogram(stat = 'identity', color = 'black', fill = 'white')+
  stat_function(fun = dnorm, args = list(mean, sd), color = 'midnightblue')+
  labs(title = 'BINOMIAL AND NORMAL DISTRIBUTIONS: n = 20, p = 0.1')+
  xlim(1.5, 20)+xlab('X')+ylab('DENSITY')+theme_bw()
```

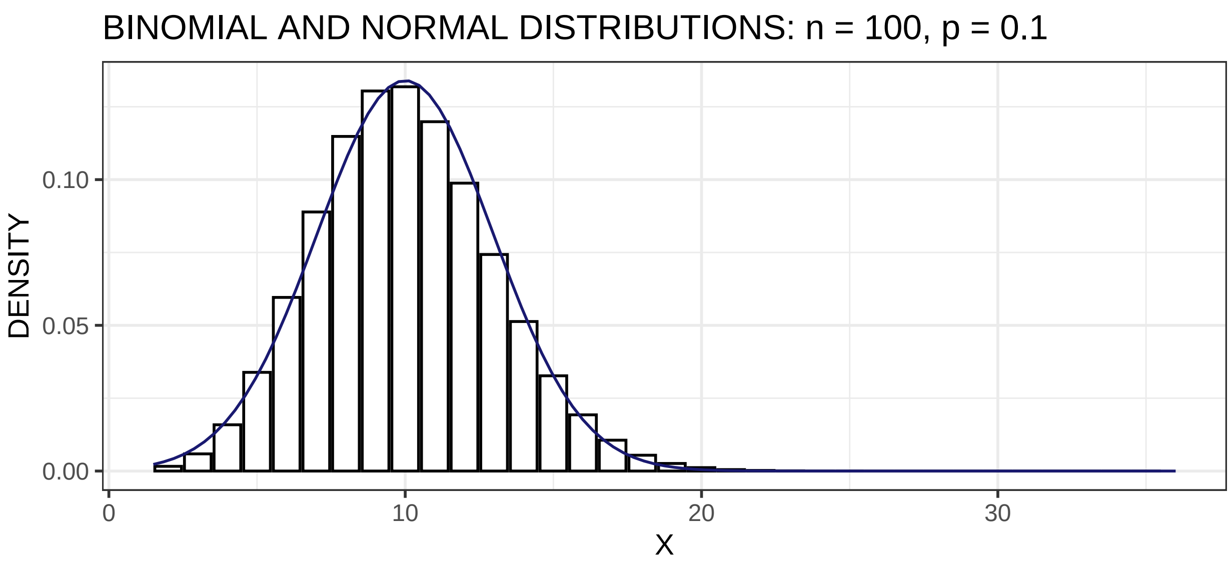


d)

```
y100_p01 = dbinom(x100, 100, 0.1)
dbin100_p01 = data.frame(y100_p01)
mean = mean(y100_p01)
sd = sd(y100_p01)
```

A histogram of the binomial distribution and the graph of the density of the normal distribution, p = 0.1 i n = 100.

```
ggplot(dbin100_p01, aes(x100, y100_p01))+
  geom_histogram(stat = 'identity', color = 'black', fill = 'white')+
  stat_function(fun = dnorm, args = list(mean, sd), color = 'midnightblue')+
  labs(title = 'BINOMIAL AND NORMAL DISTRIBUTIONS: n = 100, p = 0.1')+
  xlim(1.5, 36)+xlab('X')+ylab('DENSITY')+theme_bw()
```



Assignment 2

```
confidence_interval_t2 = function(n){
  sample = rnorm(n, 0, 1)
  mean = mean(sample)
  sigma = sd(sample)
  qse = qnorm(0.975)*(sigma/sqrt(n))
  left = mean - qse
  right = mean + qse
  if (0 >= left & 0 <= right){
    contain = 1
  } else {
    contain = 0}
  return(c(left, right, contain))}
```

a)

```
set.seed(1)
conf = confidence_interval_t2(100)
confid = round(conf, digits = 3)
cat('confidence interval:\n', '[', str_c(confid[1:2], c(', ', '')), ']', sep = '')
```

```
confidence interval:
[-0.067, 0.285]
```

I use the function and I generate 100-element random sample from a standard normal distribution.
Then I construct a confidence interval for the expected value at the 95% confidence level.

b)

```
vec = c()
width100 = c()
set.seed(10)
for (i in 1:1000){
  con100 = confidence_interval_t2(100)
  width100 = c(width100, (con100[2] - con100[1]))
  if (con100[3] == 1){
    vec = c(vec, 1)
  } else {
    vec = c(vec, 0)}}

len = length(vec[vec == 1])/1000
cat(str_c('probability: ', len*100, '%', '\n'))
```

```
probability: 95.5%
```

I repeat the previous experience 1000 times and I calculate how often the constructed confidence intervals contain the actual expected value.
The expected value is contained in 95.5% of cases so confidence level is actually 95%.

c)

```
vec = c()
width200 = c()
set.seed(1)
for (i in 1:1000){
  con200 = confidence_interval_t2(200)
  width200 = c(width200, (con200[2] - con200[1]))
  if (con200[3] == 1){
    vec = c(vec, 1)
  } else {
    vec = c(vec, 0)}}
prob = length(vec[vec == 1])/length(vec)
mean_width = mean(width100) > mean(width200)
cat(str_c('probability: ', prob*100, '%', '\nis it wider? ', mean_width, '\n'), sep = '')
```

```
  probability: 95.5%
  is it wider? TRUE
```

The probability of containing the actual expected value is 95.5%.
Average width of the confidence interval based on a 100-element sample is greater than average width based on a 200-element sample.
The estimation is better when a sample size is larger.

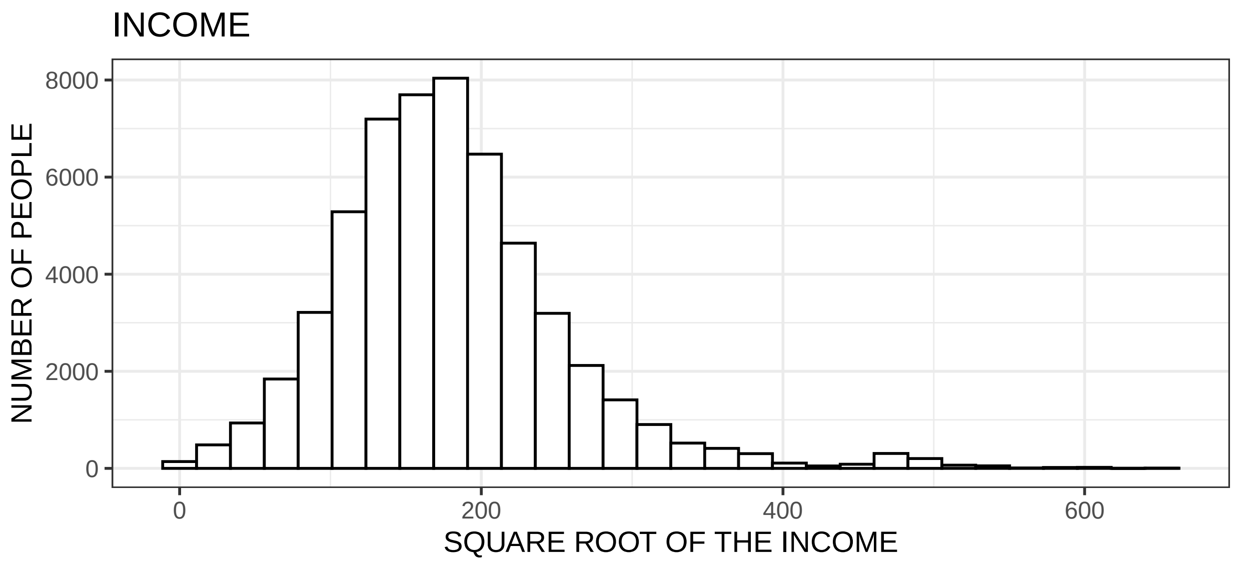
Assignment 3

```
income = read.table('income.dat.txt')
colnames(income) = (c('nb', 'age', 'edu', 'gender', 'inc', 'job_cl'))
income = filter(income, inc >= 0)
inc = income$inc
U = sqrt(inc)
```

I select just incomes which are greater than 0.

a)

```
ggplot()+
  geom_histogram(aes(U), color = 'black', fill = 'white')+
  xlab('SQUARE ROOT OF THE INCOME')+ylab('NUMBER OF PEOPLE')+ggtitle('INCOME')+
  theme_bw()
```



The picture above shows the histogram of square root of the income.

```
miU = round(mean(U), digits = 3)
miD = round(mean(inc), digits = 3)
cat(str_c('μU: ', miU, '\n'), str_c('μD: ', miD, '\n'), sep = '')
```

```
  μU: 179.409
  μD: 37993.628
```

The mean of square root for the whole data set is about 179.
The mean income is almost 37994.

b)

```
sampU = sample(U, 200)
est_miU = round(mean(sampU), digits = 3)
sampD = sample(inc, 200)
est_miD = round(mean(sampD), digits = 3)
cat('estimators:\n',
    str_c('est_µU: ', est_miU, '\n'),
    str_c('est_µD: ', est_miD, '\n'), sep = '')
```

```
estimators:
est_µU: 182.594
est_µD: 33321.15
```

I draw a 200-element random sample from the data set and I construct estimators for µU and µD.

```
confidence_interval_t3 = function(sample, mean, mi){
  sd = sd(sample)
  qse = qt(0.975, length(sample)-1)*(sd/sqrt(length(sample)))
  left = mean - qse
  right = mean + qse
  conf = c(left, right)
  if (mi >= left & mi <= right){
    contain = 1
    } else {
    contain = 0
  }
  return(c(conf, contain))}

ciU = confidence_interval_t3(sampU, est_miU, miU)
ciD = confidence_interval_t3(sampD, est_miD, miD)
ciU = round(ciU, digits = 3)
if (ciU[3] == 1){
  ciU[3] = 'interval contains actual value'
  } else {
    ciU[3] = 'interval doesn\'\t contain actual value'}
ciD = round(ciD, digits = 3)
if (ciD[3] == 1){
  ciD[3] = 'interval contains actual value'
  } else {
    ciD[3] = 'interval doesn\'\t contain actual value'}

cat('confidence intervals:\nU: [',
    str_c(ciU[1:2], c(', ', '')), ']\n   ', ciU[3], '\nD: [',
    str_c(ciD[1:2], c(', ', '')), ']\n   ', ciD[3], '\n', sep = '')
```

```
confidence intervals:
U: [173.378, 191.81]
   interval contains actual value
D: [28584.801, 38057.499]
   interval contains actual value
```

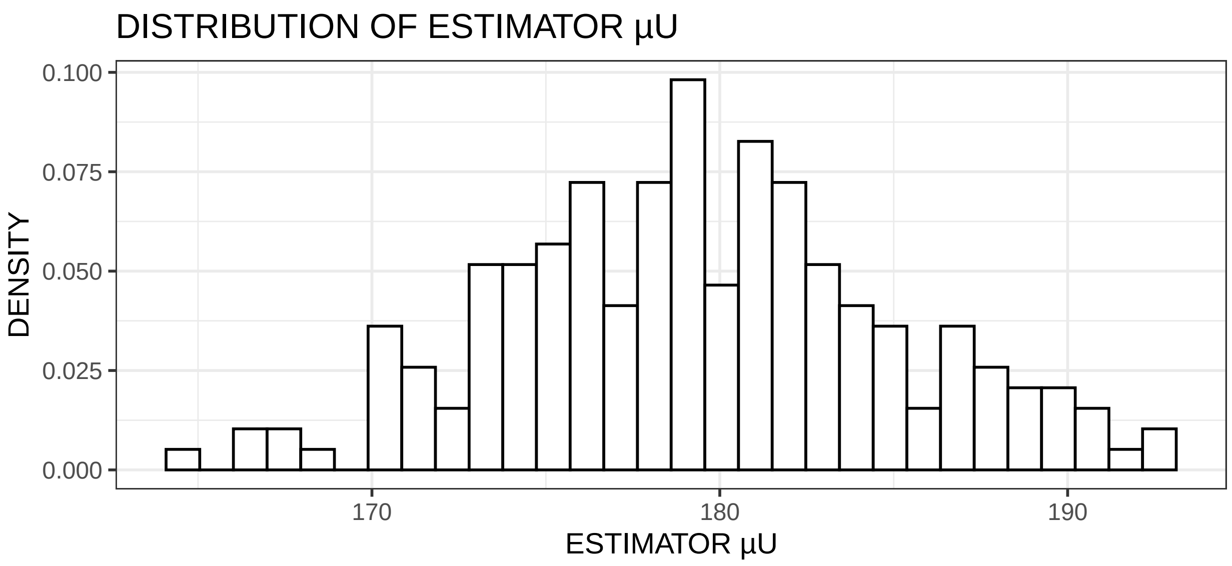
I use the function twice to construct 95% confidence intervals for µU and µD parameters.
The function return information that the both intervals contain the actual values of the parameters.

c)

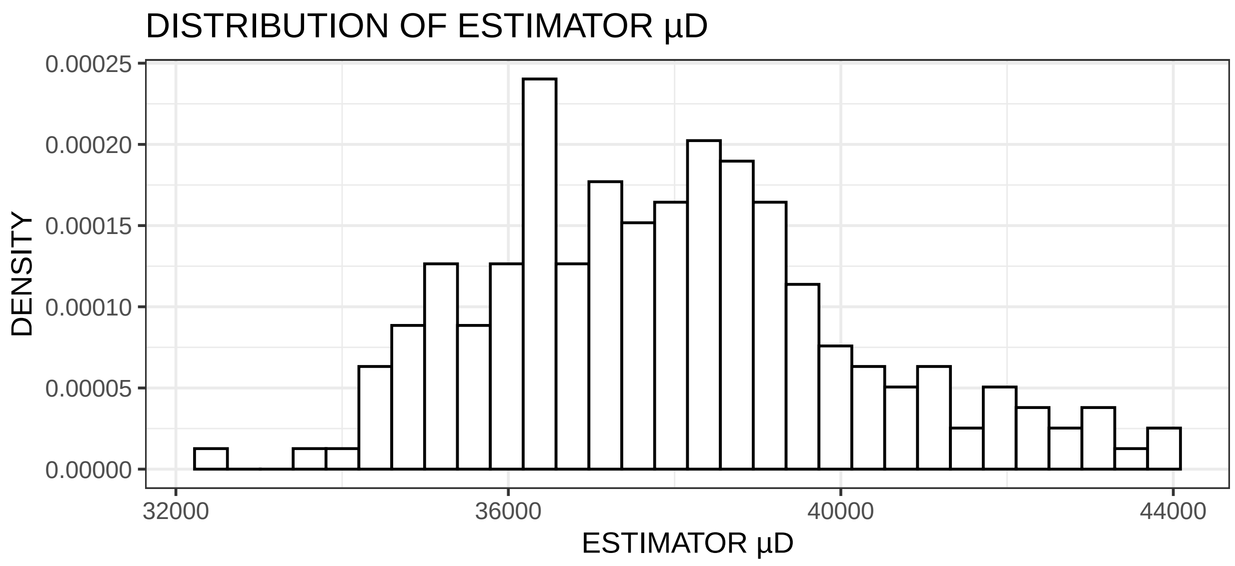
```
vec_m_U = c()
vec_m_D = c()
contU = c()
contD = c()
set.seed(90)
for (i in 1:200){
  sampU = sample(U, 200)
  est_miU = mean(sampU)
  sampD = sample(inc, 200)
  est_miD = mean(sampD)
  conU = confidence_interval_t3(sampU, est_miU, miU)
  conD = confidence_interval_t3(sampD, est_miD, miD)
  vec_m_U = c(vec_m_U, est_miU)
  vec_m_D = c(vec_m_D, est_miD)
  if (conU[3] == 1){
    contU = c(contU, 1)
  if (conD[3] == 1){
    contD = c(contD, 1)
  }
    }
}
```

I repeat the algorithm 200 times.

```
ggplot()+geom_histogram(aes(vec_m_U, ..density..), color = 'black', fill = 'white')+
  xlab('ESTIMATOR µU')+ylab('DENSITY')+labs(title = 'DISTRIBUTION OF ESTIMATOR µU')+
  theme_bw()
```



```
ggplot()+geom_histogram(aes(vec_m_D, ..density..), color = 'black', fill = 'white')+
  xlab('ESTIMATOR µD')+ylab('DENSITY')+labs(title = 'DISTRIBUTION OF ESTIMATOR µD')+
  theme_bw()
```



There are histograms of the distributions of the estimators above.

```
probU = round(100*length(contU)/200, digits = 3)
probD = round(100*length(contD)/200, digits = 3)
cat(str_c('probability µU: ', probU, '%\n'), str_c('probability µD: ', probD, '%\n'), sep = '')
```

```
  probability µU: 95%
  probability µD: 93.5%
```

The probability of containing the actual expected value for square root of the income variable is 95%
so the confidence level condition is fulfilled.
The approximation for the income is a little worse because the probability is 93.5%.

Assignment 4

```
grades = read.table('grades.txt')
colnames(grades) = c('nb', 'gpa', 'iq', 'gender', 'pt')
iq = grades$iq
pt = grades$pt

confidence_interval_t4 = function(vec){
  mean = mean(vec)
  sd = sd(vec)
  qse = qt(0.975, length(vec)-1)*(sd/sqrt(length(vec)))
  left = mean - qse
  right = mean + qse
  conf = c(left, right)
  names(conf) = c('left', 'right')
  return(conf)}

con_iq = round(confidence_interval_t4(iq), digits = 3)
con_pt = round(confidence_interval_t4(pt), digits = 3)
cat('confidence intervals:\nIQ: [',
    str_c(con_iq, c(', ', '')), ']\nPT: [',
    str_c(con_pt, c(', ', '')), ']\n', sep = '')
```

```
  confidence intervals:
  IQ: [105.953, 111.893]
  PT: [54.163, 59.76]
```

I construct confidence intervals for the average IQ and the average result of the psychological test using the function.
We don't know the sigma estimator so I use the t-distribution.