Лабораторная работа №1 «Первичный графический анализ статистических данных»
Тема: Динамика изменения оценок по теории вероятности и математической статистике за два года.
Выполнил: Саратовцев Артем 18Пи-1
Цель работы: Создание статистического ряда и изучение графических методов первичного анализа статистических данных с использованием встроенных в базовую версию пакета R функций.
ТЕОРИЯ:
Одномерное непрерывное равномерное распределение - распределение случайной вещественной величины, принимающей значения, принадлежащие некоторому промежутку конечной длины, характеризующееся тем, что плотность вероятности на этом промежутке почти всюду постоянна.
Нормальное распределение (распределение Гаусса) - распределение вероятностей, которое в одномерном случае задаётся функцией плотности вероятности, совпадающей с функцией Гаусса:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

где параметр μ — математическое ожидание (среднее значение), медиана и мода распределения, а параметр σ — среднеквадратическое отклонение (σ^2 — дисперсия) распределения.
Математическое ожидание - среднее (взвешенное по вероятностям возможных значений) значение случайной величины. Для непрерывных случайных величин находится по формуле:

$$M[X] = \int_{-\infty}^{\infty} x f_X(x)\, dx$$

Дисперсия - мера разброса значений случайной величины относительно её математического ожидания. Находится по формуле: $D[X] = M[X^2] - (M[X])^2.$

Среднеквадратическое отклонение - показатель рассеивания значений случайной величины относительно её математического ожидания. Находится по формуле: $\sigma = \sqrt{D[X]}$

Гистограмма - наглядное представление функции плотности вероятности некоторой случайной величины, построенное по выборке.

Коробка с усами (диаграмма размаха) - график, использующийся в описательной статистике, компактно изображающий одномерное распределение вероятностей. Границами ящика служат первый и третий квартили (25-й и 75-й процентили соответственно), линия в середине ящика — медиана (50-й процентиль). Концы усов — края статистически значимой выборки (без выбросов), минимальное и максимальное наблюдаемые значения данных по выборке (в этом случае выбросы отсутствуют).

Квантиль - значение, которое заданная случайная величина не превышает с фиксированной вероятностью. Если вероятность задана в процентах, то квантиль называется процентилем или перцентилем.

- 0,25 - квантиль называется первым или нижним квартилем;
- 0,50 - квантиль называется медианой или вторым квартилем;
- 0,75 - квантиль называется третьим или верхним квартилем.

Процентиль - это процентная доля элементов из выборки стандартизации, первичный результат которых ниже данного первичного показателя.

ХОД РАБОТЫ:

Задание 1:

1. Создать два массива данных оценок по стобальной системе, объемом 51 и 62 элемента. Для этого использовать случайный набор чисел из нормального распределения с математическим ожиданием 70 и 65 и среднеквадратичным отклонением 30 и 40, соответственно. Начальный отсчет – Ваш номер в списке группы. Все числа >100 и все отрицательные числа заменить на специальную переменную NA.

Для этого я воспользовался функцией R rnorm() и собственной написанной функцией insetrNA().

Функция rnorm(n, mean= , sd=):

значения:

n - кол-во элементов выборки

mean - математическое ожидание

sd - среднеквадратическое отклонение

нужна для создания выборки, используя нормальное распределение.

Функция insertNA(arr):

arr — выборка

нужна для замены всех отрицательных значений и значений, превышающих 100 на NA.

```r
insertNA <- function(arr) {
# replaces elements, which less than 0 or more than 100, with NA
  for (i in 1:length(arr)) {
    # arr[i] <- round(arr[i]) # if you want to round the elements of the array
    if ((arr[i]<0 || 100<arr[i])&&!is.na(arr[i])) {
      arr[i] <- NA
    }
  }
  return(arr)
}
```

Код:

```r
N1 <- 51 # number of elements
math_expect1 <- 70 # mathematical expectation
st_dev1 <- 30 # standard deviation
marks1 <-rnorm(N1, mean = math_expect1, sd=st_dev1); marks1
marks1 <- insertNA(marks1); marks1
```

```r
N2 <- 62 # number of elements
math_expect2 <- 65 # mathematical expectation
st_dev2 <- 40 # standard deviation
marks2 <- rnorm(N2, mean = math_expect2, sd=st_dev2); marks2
marks2 <- insertNA(marks2); marks2
```

Результат выполнения (на следующей стр.):

```
> N1 <- 51 # number of elements
> math_expect1 <- 70 # mathematical expectation
> st_dev1 <- 30 # standard deviation
> marks1 <-rnorm(N1, mean = math_expect1, sd=st_dev1); marks1
 [1]  56.141129 127.790218  73.661644 102.761517 -19.167724 103.854672 109.360004  55.521761
 [9]  63.254230 120.032959  71.236556 101.145245  87.629288  65.251013  61.237438  27.403031
[17]   5.850459  69.324997  78.093129  54.910918  88.924222  84.236956  59.780228  71.488047
[25] 108.857896  79.438726  94.047175  84.581767 121.766832 147.575032  40.304888  56.186764
[33]  77.129845  61.918082  64.913378  89.946069  46.733473  12.422166 130.579976  83.056315
[41] 101.505065  60.636555  72.808529   5.404367  80.739792  84.035944  42.799335  54.119503
[49] 110.307302  84.083022  62.406847
> marks1 <- insertNA(marks1); marks1
 [1] 56.141129        NA 73.661644        NA        NA        NA        NA 55.521761 63.254230
[10]        NA 71.236556        NA 87.629288 65.251013 61.237438 27.403031  5.850459 69.324997
[19] 78.093129 54.910918 88.924222 84.236956 59.780228 71.488047        NA 79.438726 94.047175
[28] 84.581767        NA        NA 40.304888 56.186764 77.129845 61.918082 64.913378 89.946069
[37] 46.733473 12.422166        NA 83.056315        NA 60.636555 72.808529  5.404367 80.739792
[46] 84.035944 42.799335 54.119503        NA 84.083022 62.406847

> N2 <- 62 # number of elements
> math_expect2 <- 65 # mathematical expectation
> st_dev2 <- 40 # standard deviation
> marks2 <- rnorm(N2, mean = math_expect2, sd=st_dev2); marks2
 [1]  79.764149  11.131712 111.877417 -11.349454  24.824368  92.067501 111.092870 -49.712280
 [9]  89.052354  78.666013  63.588760  39.953399  30.761152  71.071114 113.254908   5.456885
[17]  22.985035 130.514965  16.706633 -40.046203  65.698531  78.767761  65.508793  30.061995
[25]  78.712011  57.904490 101.857330  77.037776  92.735470  78.018876  81.322041  95.393690
[33] -26.486141  85.713530  11.110550  79.526394 117.590103  47.067764  32.673421  61.549260
[41] 128.416971  69.042298  70.950488  43.142613  43.892977  59.405138   1.423543 114.364167
[49]  51.581806 -15.040100  64.301664  55.361324  -6.679576 146.987660  20.311296  10.858018
[57]  83.052092 106.329097  57.164572  74.092415  20.912010 103.211095
> marks2 <- insertNA(marks2); marks2
 [1] 79.764149 11.131712        NA        NA 24.824368 92.067501        NA        NA 89.052354
[10] 78.666013 63.588760 39.953399 30.761152 71.071114        NA  5.456885 22.985035        NA
[19] 16.706633        NA 65.698531 78.767761 65.508793 30.061995 78.712011 57.904490        NA
[28] 77.037776 92.735470 78.018876 81.322041 95.393690        NA 85.713530 11.110550 79.526394
[37]        NA 47.067764 32.673421 61.549260        NA 69.042298 70.950488 43.142613 43.892977
[46] 59.405138  1.423543        NA 51.581806        NA 64.301664 55.361324        NA        NA
[55] 20.311296 10.858018 83.052092        NA 57.164572 74.092415 20.912010        NA
```

Задание 2:

2. Разбить полученные данные на категории (использовать R-функции cut(), table()):

а) по пятибальной системе:

2 – баллы от 0 до 50;

3 – баллы от 51 до 68;

4 – баллы от 69 до 81;

5 – баллы от 86 до 100.

б) по европейской системе:

F - баллы от 0 до 30;

FX - баллы от 31 до 50;

E - баллы от 51 до 60;

D - баллы от 61 до 68;

C - баллы от 69 до 85;

B - баллы от 86 до95;

A - баллы от 96 до 100.

Для этого я использовал собственно написанные функции moveTo5Point() и moveToEUPoint() и функцию R table() для проверки корректности работы предыдущих функций.
Функция moveTo5Point(arr):

arr — выборка

переводит оценки из 100-бальной системы в 5-бальную посредством разбиения выборки с помощью фунции cut(), которая получает на вход вектор и делит их на равные или заранее заданные интервалы.

```
12 moveTo5Point <- function(arr) {
13 # exchange on 5-point system
14   arr <- cut(x=arr, breaks = c(0, 50, 68, 81, 100)); arr
15   arr.f <- factor(arr); arr.f
16   levels(arr.f) <- c("2", "3", "4", "5"); arr.f
17   arr.o <- ordered(arr.f, labels = c("2", "3", "4", "5")); arr.o
18   return(arr.o)
19 }
```

Функция moveToEUPoint(arr):

arr — выборка

переводит оценки из 100-бальной системы в 7-бальную посредством разбиения выборки с помощью фунции cut(), которая получает на вход вектор и делит их на равные или заранее заданные интервалы.

Функция table(x):

x — набор значений

возвращает таблицу с частотами встречаемости каждого значения x.

```
21 moveToEUPoint <- function(arr) {
22 # exchange on European(EU) system
23   arr <- cut(x=arr, breaks = c(0, 30, 50, 60, 68, 85, 95, 100)); arr
24   arr.f <- factor(arr); arr.f
25   levels(arr.f) <- c("F", "FX", "E", "D", "C", "B", "A"); arr.f
26   arr.o <- ordered(arr.f, labels = c("F", "FX", "E", "D", "C", "B", "A")); arr.o
27   return(arr.o)
28 }
```

Код:

```
80 marks1.5 <- moveTo5Point(arr = marks1); table(marks1.5)
81 marks1.EU <- moveToEUPoint(marks1); table(marks1.EU)
```

```
89 marks2.5 <- moveTo5Point(arr = marks2); table(marks2.5)
90 marks2.EU <- moveToEUPoint(marks2); table(marks2.EU)
```

Результат выполнения:

```
> marks1.5 <- moveTo5Point(arr = marks1); table(marks1.5)
marks1.5
 2  3  4  5
14  9 10 11
> marks1.EU <- moveToEUPoint(marks1); table(marks1.EU)
marks1.EU
 F FX  E  D  C  B  A
 6  8  6  3 13  6  2
> marks2.5 <- moveTo5Point(arr = marks2); table(marks2.5)
marks2.5
 2  3  4  5
19  9  4  6
> marks2.EU <- moveToEUPoint(marks2); table(marks2.EU)
marks2.EU
 F FX  E  D  C  B  A
 6 13  4  5  4  4  2
```

Задание 3:

3. Создать таблицу относительных частот в каждой категории.

Для этого я использовал собственно написанные функции makeTable() и makeDoubleTable().
Функция makeTable(arr):

arr — выборка

нужна для создания таблицы относительных частот в выборке arr.

```
30 makeTable <- function(arr) {
31 # makes table of probabiliry for one arr
32   a <- data.frame(row.names =levels(arr), table(arr)); a
33   res <- data.frame(row.names =levels(arr), probability  = prop.table( a[, -1]))
34   return(res)
35 }
```

Функция makeTable(arr1, arr2):

arr — выборка

нужна для создания таблицы относительных частот в выбороках arr1 и arr2 для удобного их сравнения.

```
37 makeDoubleTable <- function(arr1, arr2) {
38 # makes table of probabiliry for two arrs for better comparison
39   a <- data.frame(row.names =levels(arr1), table(arr1)); a
40   b <- data.frame(row.names =levels(arr2), table(arr2)); b
41   res <- data.frame(row.names =levels(arr1), probability1  = prop.table( a[, -1]), probability2  = prop.table( b[, -1]))
42   return(res)
43 }
```

Код:

```
 92 # making tables
 93 table1.5 <- makeTable(marks1.5); table1.5
 94 table1.EU <- makeTable(marks1.EU); table1.EU
 95
 96 table2.5 <- makeTable(marks2.5); table2.5
 97 table2.EU <- makeTable(marks2.EU); table2.EU
 98
 99 table.5 <- makeDoubleTable(marks1.5, marks2.5); table.5
100 table.EU <- makeDoubleTable(marks1.EU, marks2.EU); table.EU
```

Результат выполнения:

```
> table1.5 <- makeTable(marks1.5); table1.5
  probability
2   0.3181818
3   0.2045455
4   0.2272727
5   0.2500000
> table1.EU <- makeTable(marks1.EU); table1.EU
   probability
F   0.13636364
FX  0.18181818
E   0.13636364
D   0.06818182
C   0.29545455
B   0.13636364
A   0.04545455
```

table1.5

Filter

	probability
2	0.3181818
3	0.2045455
4	0.2272727
5	0.2500000

table1.EU

Filter

	probability
F	0.13636364
FX	0.18181818
E	0.13636364
D	0.06818182
C	0.29545455
B	0.13636364
A	0.04545455

```
> table2.5 <- makeTable(marks2.5); table2.5
  probability
2   0.5000000
3   0.2368421
4   0.1052632
5   0.1578947
> table2.EU <- makeTable(marks2.EU); table2.EU
   probability
F   0.15789474
FX  0.34210526
E   0.10526316
D   0.13157895
C   0.10526316
B   0.10526316
A   0.05263158
```

table2.5

	probability
2	0.5000000
3	0.2368421
4	0.1052632
5	0.1578947

table2.EU

	probability
F	0.15789474
FX	0.34210526
E	0.10526316
D	0.13157895
C	0.10526316
B	0.10526316
A	0.05263158

```
> table.5 <- makeDoubleTable(marks1.5, marks2.5); table.5
  probability1 probability2
2    0.3181818    0.5000000
3    0.2045455    0.2368421
4    0.2272727    0.1052632
5    0.2500000    0.1578947
> table.EU <- makeDoubleTable(marks1.EU, marks2.EU); table.EU
   probability1 probability2
F    0.13636364   0.15789474
FX   0.18181818   0.34210526
E    0.13636364   0.10526316
D    0.06818182   0.13157895
C    0.29545455   0.10526316
B    0.13636364   0.10526316
A    0.04545455   0.05263158
```

table.5

	probability1	probability2
2	0.3181818	0.5000000
3	0.2045455	0.2368421
4	0.2272727	0.1052632
5	0.2500000	0.1578947

table.EU

	probability1	probability2
F	0.13636364	0.15789474
FX	0.18181818	0.34210526
E	0.13636364	0.10526316
D	0.06818182	0.13157895
C	0.29545455	0.10526316
B	0.13636364	0.10526316
A	0.04545455	0.05263158

Задание 4

4. Построить гистограммы и график функции плотности на одном рисунке для каждой выборки, взяв соответствующие интервалы для построения гистограмм.

Для этого я использовал собственно написанные функции makePlot5() и makePlotEU(), в которых также использовались такие функции, как par(), hist(), lines(), density().
Функция par():

нужна для того, чтобы разделить пространство, на котором будут графики, на некое определенное количество маленьких пространств, для того, чтобы нарисовать сразу несколько графиков на одном экране.

Функция hist(x, breaks =):

x — переменная

breaks — кол-во столбцов

нужна для создания гистограмм частот значений переменной x; аргумент breaks = можно использовать, чтобы изменить принятое по умолчанию количество столбцов.

Функция lines(x, col = , lwb =):

x — значения

col — цвет

lwd — толщина линии

нужна для создания линий на графиках.

Функция density():

нужна для нахождения ядерных плотностей вероятностей (ядерная плотность вероятности — оценка случайной величины).

Функция makePlot5(arr1, arr2):

arr1, arr2 -выборки

нужна для создания гистограмм двух выборок в 5-балльной системе счисления.

```r
makePlot5 <- function(arr1, arr2) {
# function for 5-point system
# makes two histogramms and graphs the density functions of arr1 and arr2
#
  par(mfcol=c(1, 2))

  marks1 <- as.numeric(as.character(na.omit(arr1))); marks1
  hist(marks1, freq = FALSE)
  lines(density(marks1, na.rm = TRUE), col = "red", lwd = 2)

  marks2 <- as.numeric(as.character(na.omit(arr2))); marks2
  hist(marks2, freq = FALSE)
  lines(density(marks2, na.rm = TRUE), col = "red", lwd = 2)
}
```

Функция makePloEU(arr1, arr2):

arr1, arr2 -выборки

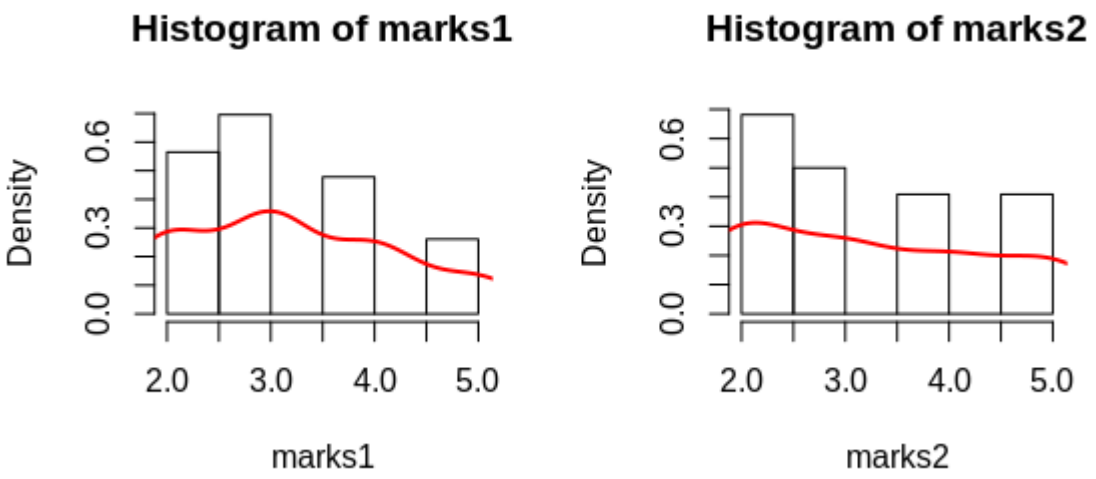
нужна для создания гистограмм двух выборок в 7-балльной системе счисления.

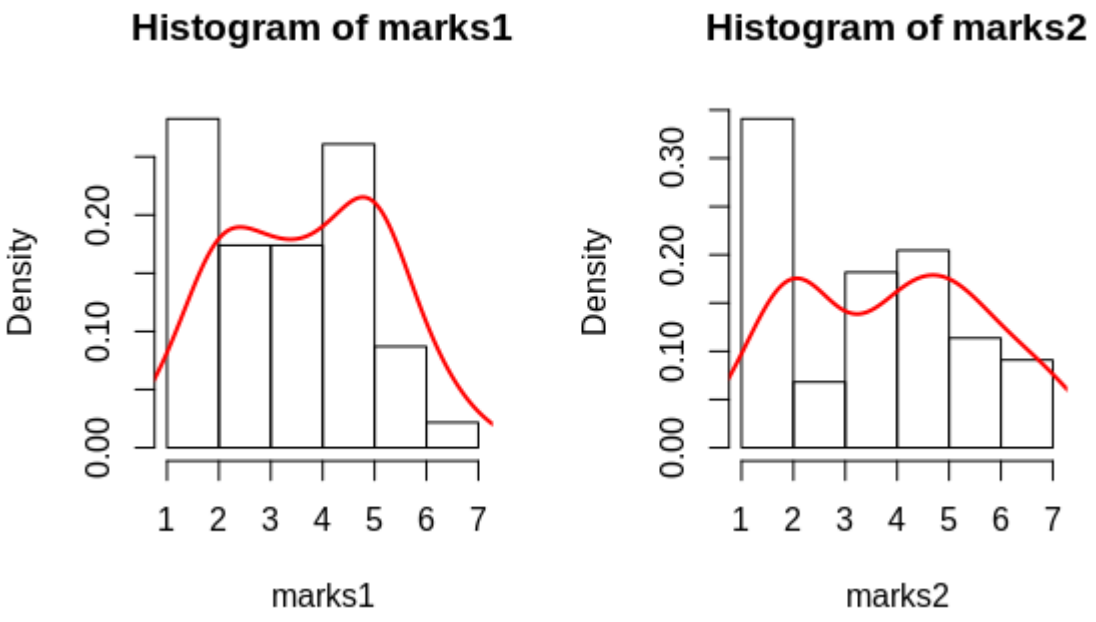
```r
makePlotEU <- function(arr1, arr2) {
# function for EU-point system
# makes two histogramms and graphs the density functions of arr1 and arr2
  par(mfcol=c(1, 2))

  marks1 <- as.numeric(na.omit(arr1)); marks1
  hist(marks1, freq = FALSE)
  lines(density(marks1, na.rm = TRUE), col = "red", lwd = 2)

  marks2 <- as.numeric(na.omit(arr2)); marks2
  hist(marks2, freq = FALSE)
  lines(density(marks2, na.rm = TRUE), col = "red", lwd = 2)
}
```

Код:

```
102  # making histograms
103  makePlot5(marks1.5, marks2.5)
104  makePlotEU(marks1.EU, marks2.EU)
```

Результат выполнения:





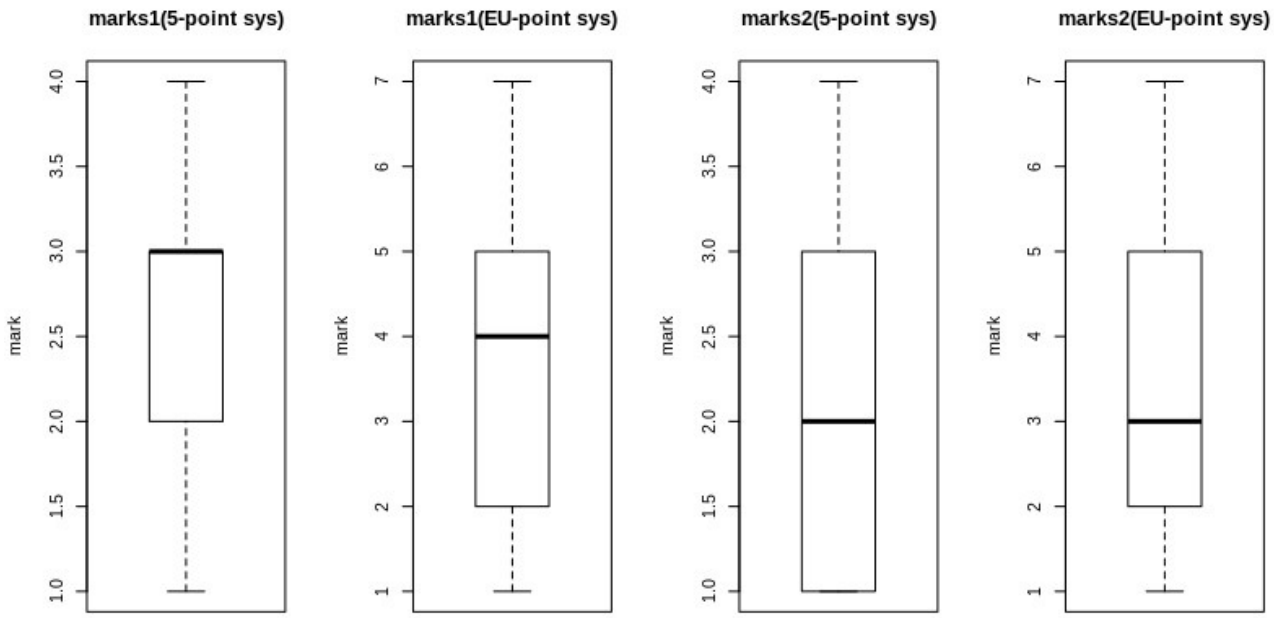
Задание 5:

5. На одном рисунке построить «ящики с усами» для всех выборок.

Для этого использовал функцию boxplot() и dev.off().
Функция dev.off():
нужна для очистки окна вывода.
Функция boxplot(x, main = , ylab =):
x — выборка
main — название графика
ylab — название оси OY
нужна для построения диаграмм размахов («коробок с усами»).
Код:

```
106  # making boxes
107  dev.off()
108  par(mfcol=c(1, 4))
109
110  boxplot(marks1.5, main="marks1(5-point sys)", ylab="mark")
111  boxplot(marks1.EU, main="marks1(EU-point sys)", ylab="mark")
112  boxplot(marks2.5, main="marks2(5-point sys)", ylab="mark")
113  boxplot(marks2.EU, main="marks2(EU-point sys)", ylab="mark")
```

Результат выполнения:



Вывод:

Проведя данную лабораторную работу, я укрепил свои знания в теоретической части, а также получил опыт использования встроенных в R и написания собственных функций в Rstudio.

Конец.