Stat 292 - Recitation 1

R-Review

Orçun Oltulu

17 / 3 / 2021

Exercise 1:

Create following sequences.

- (100 110 120 130 140 150 160 170 180 190 200)
- (25 20 15 10 5 0 -5 -10 -15)
- (-2.0 -1.5 -1.0 -0.5 0.0 0.5 1.0 1.5 2.0 -2.0 -1.5 -1.0 -0.5 0.0 0.5 1.0 1.5 2.0)
- (-5 -5 4 4 -3 -3 2 2 -1 -1 0 0)
- (“a” “c” “e” “g” “i” “k” “m” “o” “q” “s” “u” “w” “y”)

```
seq(100, 200, 10)
```

```
##  [1] 100 110 120 130 140 150 160 170 180 190 200
```

```
seq(25, -15, length.out = 9)
```

```
## [1]  25  20  15  10   5   0  -5 -10 -15
```

```
rep(seq(-2,2,0.5),2)
```

```
##  [1] -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0 -2.0 -1.5 -1.0 -0.5  0.0  0.5
## [16]  1.0  1.5  2.0
```

```
rep(seq(5,0,-1) * c(-1,1),each=2)
```

```
##  [1] -5 -5  4  4 -3 -3  2  2 -1 -1  0  0
```

```
letters[seq(1,length(letters),2)]
```

```
##  [1] "a" "c" "e" "g" "i" "k" "m" "o" "q" "s" "u" "w" "y"
```

Exercise 2:

Part A:

Create following matrices.

$$X = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}, \quad Y = \begin{bmatrix} 1 & -2 & -2 \\ 0 & 0 & 1 \\ -1 & 5 & 2 \\ 2 & 4 & 6 \end{bmatrix}, \quad Z = \begin{bmatrix} 2 & 1 & 3 \\ 3 & -1 & 5 \\ 4 & -2 & 6 \end{bmatrix}, \quad P = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 8 \end{bmatrix}, \quad K = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

```
X <- matrix(rep(c(1,0,1),3), nrow = 3, byrow = TRUE)
Y <- matrix(c(1,-2,-2,0,0,1,-1,5,2,2,4,6), nrow = 4, byrow = TRUE)
Z <- matrix(c(2,3,4,1,-1,-2,3,5,6), nrow = 3)
P <- diag(5:8)
K <- matrix(1:6, nrow = 2, byrow = T)
```

Part B:

Calculate following;

- X + Z
- $Z^{-1}X$
- YK^T
- PY
- max{|X|,|Z|,|P|}

```
X + Z
```

```
##      [,1] [,2] [,3]
## [1,]    3    1    4
## [2,]    4   -1    6
## [3,]    5   -2    7
```

```
solve(Z) %*% X
```

```
##      [,1] [,2] [,3]
## [1,] 0.00    0 0.00
## [2,] 0.25    0 0.25
## [3,] 0.25    0 0.25
```

```
Y %*% t(K)
```

```
##      [,1] [,2]
## [1,]   -9  -18
## [2,]    3    6
## [3,]   15   33
```

```
## [4,]   28   64
```

```
P %*% Y
```

```
##      [,1] [,2] [,3]
## [1,]    5  -10  -10
## [2,]    0    0    6
## [3,]   -7   35   14
## [4,]   16   32   48
```

```
max(c(det(X),det(Z),det(P)))
```

```
## [1] 1680
```

Part C:

Using those matrices;

- Replace [1,1] and [1,2] elements for each matrix with -100 and 100 respectively.
- Create a new matrix called A, by binding X and K properly, check its dimension.
- Create a new matrix called B, by binding Y and P properly, assign row and column names.
- Combine Y, P and K in a list. Add a vector of your 3 favorite TV Shows in your list.
- Create an array whose layers are X and Z.

```
X[1,1:2] <- Y[1,1:2] <- Z[1,1:2] <- P[1,1:2] <- K[1,1:2] <- c(-100,100)

A <- rbind(X,K); dim(A)
```

```
## [1] 5 3
```

```
B <- cbind(Y,P)
rownames(B) <- paste("row.",1:nrow(B),sep="")
colnames(B) <- paste("col.",1:ncol(B),sep="")

mylist <- list(Y, P, K)
mylist$TvShows <- c("Cennet Mahallesi","Acun Firarda","Çiçek Taksi")

myarray <- array(c(X,Z), dim = c(nrow(X),ncol(X),2))
```

Exercise 3:

- Read height.txt data into R. Then, write an if statement to group observations based on the following rule. Create another column on your dataframe called 'group'.

$$\text{Group} = \begin{cases} \text{Group1}, & Height \leq 165 \\ \text{Group2}, & Height > 165 \quad \& \quad Height < 180 \\ \text{Group3}, & Height \geq 180 \end{cases}$$

```
data <- read.table("height.txt",header = T)

data$Group <- ifelse(data$Height <= 165, "Group1",
                     ifelse(data$Height >= 180, "Group3","Group2"))

head(data)
```

```
##   ID Height  Group
## 1  1    153 Group1
## 2  2    188 Group3
## 3  3    150 Group1
## 4  4    183 Group3
## 5  5    172 Group2
## 6  6    163 Group1
```

- Print frequency and proportion of each group.

```
table(data$Group)
```

```
##
## Group1 Group2 Group3
##      7      6      7
```

```
table(data$Group) / nrow(data)
```

```
##
## Group1 Group2 Group3
##   0.35   0.30   0.35
```

Part B:

Write an if statement that prints;

- square of an integer if it is even and less than or equal to 20,
- cube of an integer if it is odd and less than or equal to 20,
- natural logarithm if it is more than 20.

Then, give different integers to see if your if statement works well.

```
myint <- 13
if(myint > 20){
  log(myint)
}else{
```

```
  if(myint %% 2 == 0){
    # meaning it is even
    myint^2
  }else{
    myint^3
  }
}
```

```
## [1] 2197
```

```
#ifelse(myint >20, log(myint),ifelse(myint %% 2 == 0, myint^2, myint^3))
```

Exercise 4:

Part A:

Write a while loop that prints out standard random normal numbers (use rnorm()) but stops (breaks) if you get a number bigger than 1.

```
while(TRUE){
  x <- rnorm(1)
  if(x > 1){
    break
  }
  print(x)
}
```

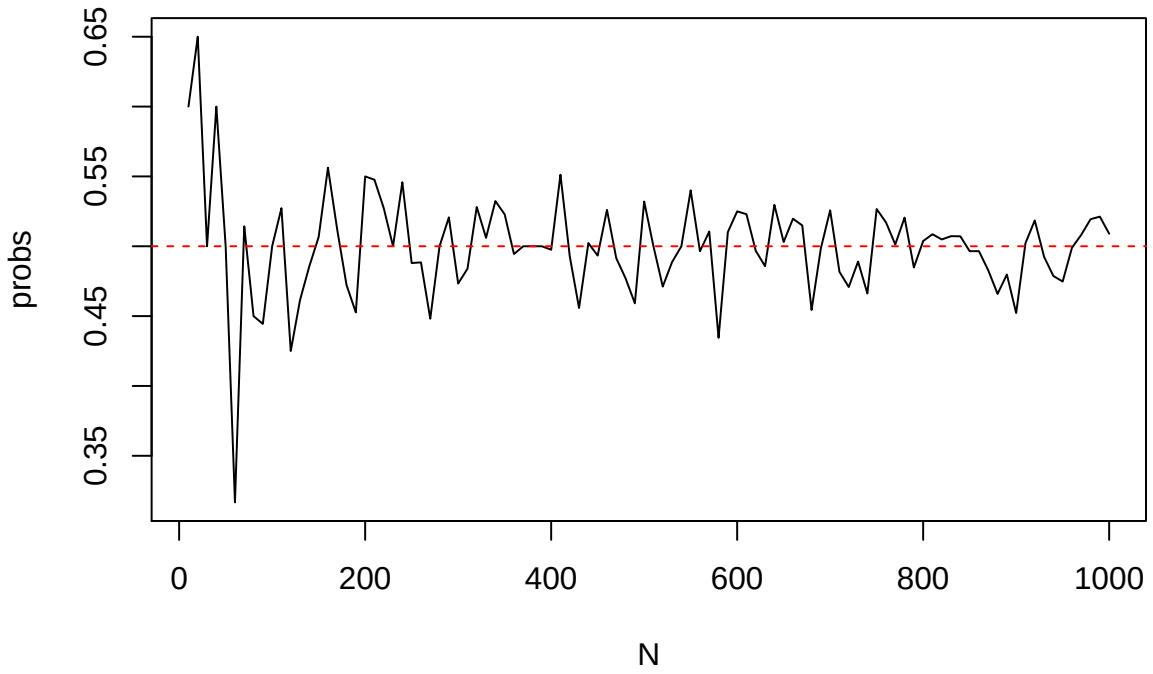
```
## [1] -0.4115108
## [1] 0.2522234
## [1] -0.8919211
## [1] 0.4356833
## [1] -1.237538
## [1] -0.2242679
## [1] 0.3773956
## [1] 0.1333364
## [1] 0.8041895
## [1] -0.05710677
## [1] 0.503608
```

Part B:

Consider a coin tossing experiment. Using 'sample' function simulate this experiment. First, 'flip your fair coin' 10 times then increase this number by 10 till you have 1000 flips (N = 10,20,30,. . . ,990,1000), and in each step store estimated probabilities in a vector.

After you simulate this experiment, draw a line graph to see if the estimated probability converges to the exact probability as N gets larger.

```
set.seed(12)
probs <- numeric()
N <- seq(10,1000,10)
for(i in 1:length(N)){
  probs[i] <- mean(sample(0:1, replace = T, size = N[i]))
}
plot(x = N, y = probs, type = "l")
abline(h = 0.5, col = "Red", lty = "dashed")
```



Part C:

Implement a multiplication game. A while loop that gives the user two random numbers from 2 to 12 and asks the user to multiply them. Only exit the loop after five correct answers. Use as.integer(readline()) to get the user's answers.

```
correct_number <- 0
while(correct_number < 5){
  x <- sample(2:12, size = 2, replace = T)
  mult_x <- x[1] * x[2]
```

```
  answer <- as.integer(readline(paste("What is the multiplication of",
                                      x[1], "and", x[2], "?",
                                      "Your answer:")))
  if(answer == mult_x){
    correct_number <- correct_number + 1
    cat("You are correct! You have",
        correct_number, "correct answers !")
  } else{
    cat("You are incorrect! Your answer is",
        answer, "The correct answer is", mult_x)
  }
}
```

Exercise 5:

Part A:

Write an R function which converts given amount of Turkish Lira to the USD, CAD, EURO and GBP.

```
currency_convertor <- function(TL){
  currency_vector <- c(TL = 1, USD = 0.13, CAD = 0.16,
                       EURO = 0.11, GBP = 0.095) # as of 13/3/21
  out_matrix <- matrix(currency_vector * TL,
                       dimnames = list(c(names(currency_vector)),"Amount"))
  return(out_matrix)
}

currency_convertor(75)
```

```
##      Amount
## TL   75.000
## USD   9.750
## CAD  12.000
## EURO  8.250
## GBP   7.125
```

Part B:

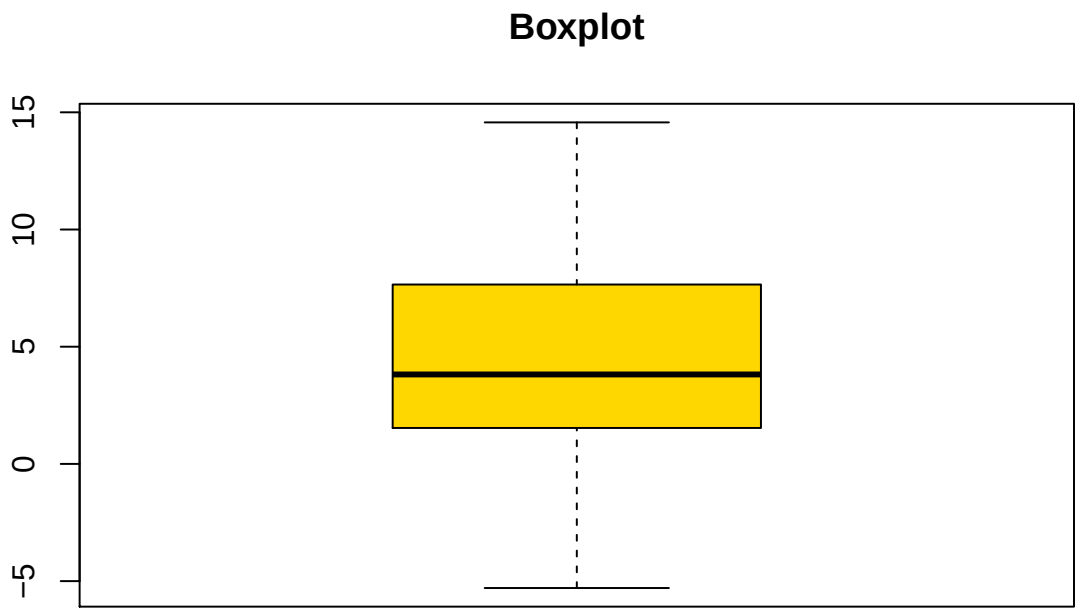
Write an R function which takes a continuous random sample (vector), calculates basic statistics (mean, median, variance, quartiles (1^{st} and 3^{rd}), skewness, kurtosis, range ...) and stores them in a list; finally returns that list. Also, your function must have an option for a boxplot, if TRUE it will plot a boxplot.

```r
all_stats <- function(vec, graph = F){

  out_list <- list(Mean = mean(vec),
                   Median = median(vec),
                   Variance = var(vec),
                   Quartiles = quantile(vec,c(0.25,0.75)),
                   Range = max(vec) - min(vec),
                   Skewness = moments::skewness(vec),
                   Kurtosis = moments::kurtosis(vec))

  if(graph){
    boxplot(vec, main = "Boxplot", col = "Gold")
  }
  return(out_list)
}

all_stats(vec = rnorm(100,5,4), graph = T)
```




```
## $Mean
## [1] 4.61188
##
```

```
## $Median
## [1] 3.816457
##
## $Variance
## [1] 19.67517
##
## $Quartiles
##      25%      75%
## 1.551665 7.633397
##
## $Range
## [1] 19.86436
##
## $Skewness
## [1] 0.1431411
##
## $Kurtosis
## [1] 2.509511
```

Part C:

Write an R function that given a vector and an integer will return how many times the integer appears inside the vector. (Don't use 'table' command)

```r
x <- sample(1:20, size = 20, replace = TRUE)

HowManyTimes <- function(vec, lookfor){
  # Do not use Table function !
  value <- sum(vec == lookfor)

  if(value == 0){

    stop(paste("Given vector does not include",
               lookfor, "! Try another one"))
    #paste("Given vector does not include", lookfor, "! Try another one")

  } else{
    paste("The value", lookfor, "appears",
          value, "times in the given vector")
  }

}

HowManyTimes(vec = x, lookfor = 15)
```

```
## [1] "The value 15 appears 1 times in the given vector"
```

Exercise 6:

Part A:

- Load 'Hitters' data set from 'ISLR' package.

```
library(ISLR)
data("Hitters")
```

- Check structure of the 'Hitters' data set.

```
str(Hitters)
```

```
## 'data.frame':    322 obs. of  20 variables:
##  $ AtBat    : int  293 315 479 496 321 594 185 298 323 401 ...
##  $ Hits     : int  66 81 130 141 87 169 37 73 81 92 ...
##  $ HmRun    : int  1 7 18 20 10 4 1 0 6 17 ...
##  $ Runs     : int  30 24 66 65 39 74 23 24 26 49 ...
##  $ RBI      : int  29 38 72 78 42 51 8 24 32 66 ...
##  $ Walks    : int  14 39 76 37 30 35 21 7 8 65 ...
##  $ Years    : int  1 14 3 11 2 11 2 3 2 13 ...
##  $ CAtBat   : int  293 3449 1624 5628 396 4408 214 509 341 5206 ...
##  $ CHits    : int  66 835 457 1575 101 1133 42 108 86 1332 ...
##  $ CHmRun   : int  1 69 63 225 12 19 1 0 6 253 ...
##  $ CRuns    : int  30 321 224 828 48 501 30 41 32 784 ...
##  $ CRBI     : int  29 414 266 838 46 336 9 37 34 890 ...
##  $ CWalks   : int  14 375 263 354 33 194 24 12 8 866 ...
##  $ League   : Factor w/ 2 levels "A","N": 1 2 1 2 2 1 2 1 2 1 ...
##  $ Division : Factor w/ 2 levels "E","W": 1 2 2 1 1 2 1 2 2 1 ...
##  $ PutOuts  : int  446 632 880 200 805 282 76 121 143 0 ...
##  $ Assists  : int  33 43 82 11 40 421 127 283 290 0 ...
##  $ Errors   : int  20 10 14 3 4 25 7 9 19 0 ...
##  $ Salary   : num  NA 475 480 500 91.5 750 70 100 75 1100 ...
##  $ NewLeague: Factor w/ 2 levels "A","N": 1 2 1 2 2 1 1 1 2 1 ...
```

- Print first 10 Players.

```
head(Hitters, 10)
```

```
##                   AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun
## -Andy Allanson      293   66     1   30  29    14     1    293    66      1
## -Alan Ashby         315   81     7   24  38    39    14   3449   835     69
## -Alvin Davis        479  130    18   66  72    76     3   1624   457     63
## -Andre Dawson       496  141    20   65  78    37    11   5628  1575    225
## -Andres Galarraga   321   87    10   39  42    30     2    396   101     12
```

```
## -Alfredo Griffin    594  169     4   74  51    35    11   4408  1133     19
## -Al Newman          185   37     1   23   8    21     2    214    42      1
## -Argenis Salazar    298   73     0   24  24     7     3    509   108      0
## -Andres Thomas      323   81     6   26  32     8     2    341    86      6
## -Andre Thornton     401   92    17   49  66    65    13   5206  1332    253
##                    CRuns CRBI CWalks League Division PutOuts Assists Errors
## -Andy Allanson        30   29     14      A        E     446      33     20
## -Alan Ashby          321  414    375      N        W     632      43     10
## -Alvin Davis         224  266    263      A        W     880      82     14
## -Andre Dawson        828  838    354      N        E     200      11      3
## -Andres Galarraga     48   46     33      N        E     805      40      4
## -Alfredo Griffin     501  336    194      A        W     282     421     25
## -Al Newman            30    9     24      N        E      76     127      7
## -Argenis Salazar      41   37     12      A        W     121     283      9
## -Andres Thomas        32   34      8      N        W     143     290     19
## -Andre Thornton      784  890    866      A        E       0       0      0
##                    Salary NewLeague
## -Andy Allanson         NA         A
## -Alan Ashby         475.0         N
## -Alvin Davis        480.0         A
## -Andre Dawson       500.0         N
## -Andres Galarraga    91.5         N
## -Alfredo Griffin    750.0         A
## -Al Newman           70.0         A
## -Argenis Salazar    100.0         A
## -Andres Thomas       75.0         N
## -Andre Thornton    1100.0         A
```

- Who has the highest and lowest salaries?

```
rownames(Hitters)[which.min(Hitters$Salary)]
```

```
## [1] "-BillyJo Robidoux"
```

```
rownames(Hitters)[which.max(Hitters$Salary)]
```

```
## [1] "-Eddie Murray"
```

- Check if there are any NA values in the data set. Given the fact that NA values appear only in the Salary column, find out the players whose Salaries are NA.

```
anyNA(Hitters)
```

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

- What is the mean 'Hits' for the ones with the highest 10% in Salary. (Remember, there are NA values in Salary, make sure you are dealing with them properly)

```
Hitters_new <- na.omit(Hitters)
Q90 <- quantile(Hitters_new$Salary,0.9)
mean(Hitters_new[Hitters_new$Salary > Q90,"Hits"])
```

```
## [1] 143.6667
```

- What is the correlation between HmRun and Hits?

```
cor(Hitters$Hits, Hitters$HmRun)
```

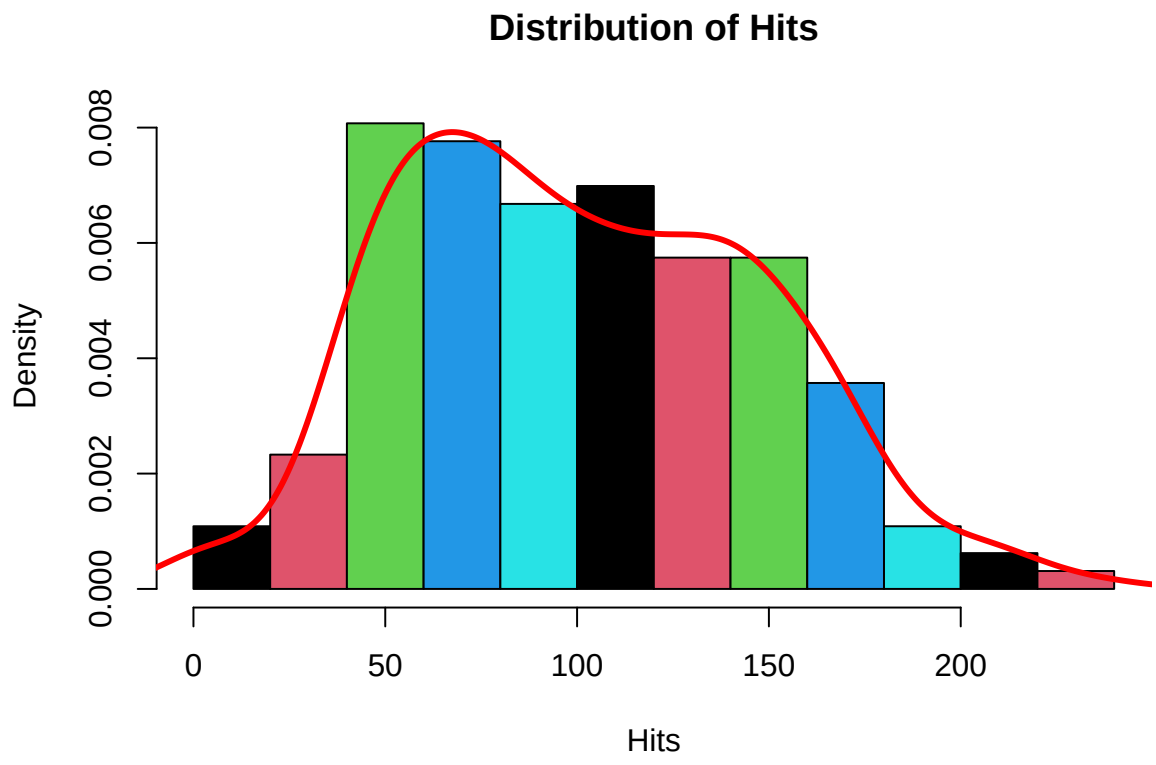
```
## [1] 0.5621579
```

Exercise 7:

Using the same Hitters data set;

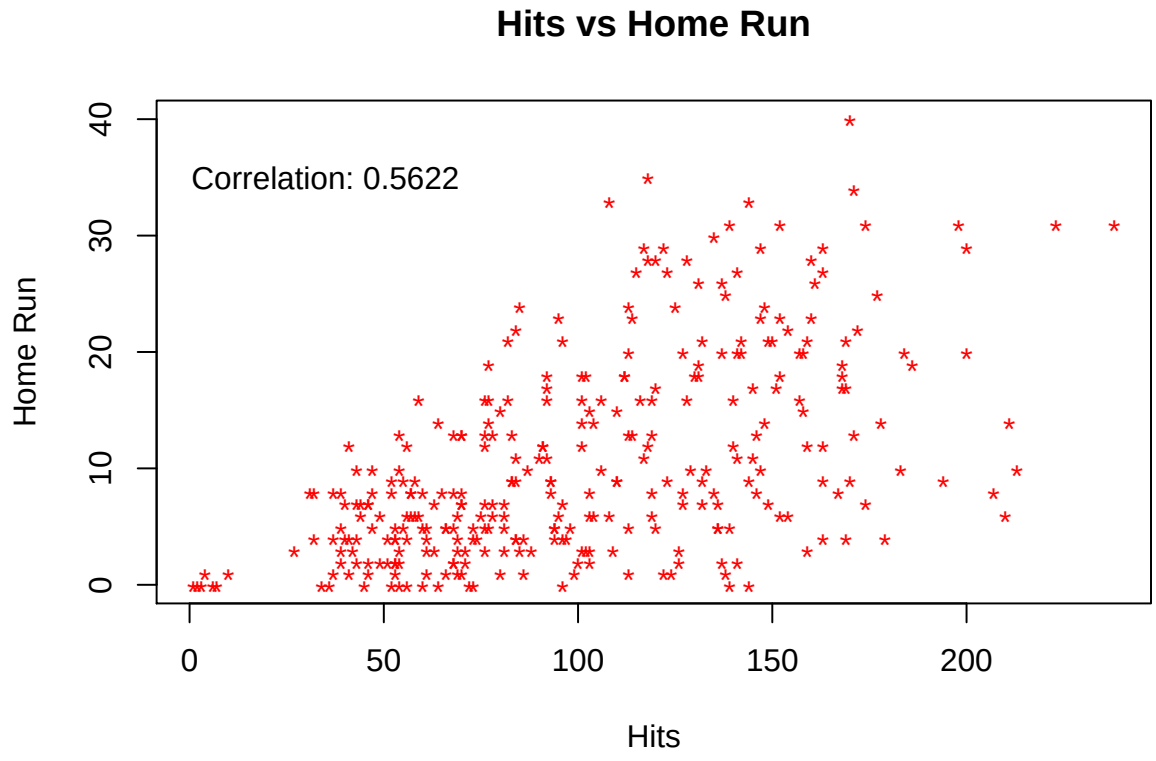
- Obtain a histogram of Hits. Give it a proper title and axis names. Also make sure your histogram has a fancy color.

```
hist(Hitters$Hits,
     main = "Distribution of Hits", xlab = "Hits",
     col = c(1,2,3,4,5), prob = T)
lines(density(Hitters$Hits), col = "Red", lwd = 3)
```



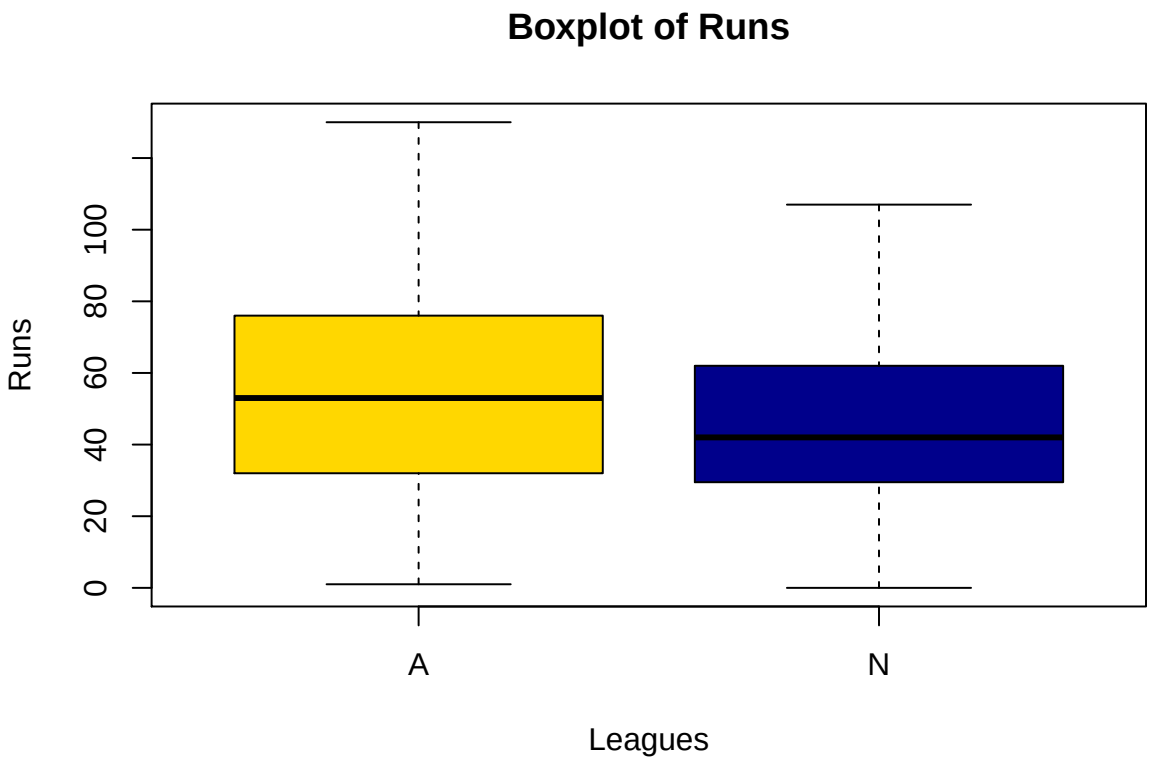
- Draw a Scatter Plot for Hits vs HmRun. Give it a proper title, and axis names. Add a text which states the correlation between those two variables.

```
plot(Hitters$Hits, Hitters$HmRun,
     main = "Hits vs Home Run", xlab = "Hits", ylab = "Home Run",
     col = "Red", pch = "*")
text(x = 35, y = 35,
     labels = paste("Correlation:",round(cor(Hitters$Hits, Hitters$HmRun),4)))
```



- Obtain Boxplots to see the distributions of Runs for each League. Give it a proper title and axis names. Also make sure your Boxplots have fancy colors.

```
boxplot(Hitters$Runs ~ Hitters$League,
        main = "Boxplot of Runs",
        xlab = "Leagues",
        ylab = "Runs",
        col = c("Gold","Dark Blue"))
```


Boxplot of Runs
Runs
0
20
40
60
80
100
A
N
Leagues