# Assignment 1

Josh Ellis

2/16/2022

## 1. Based on your reading assignments answer the following questions:

### a) What is Data Science?

Data Science is a multidisciplinary field with influences from Statistics & Mathematics, Computer Science, and Hacking. Data Science also requires a strong knowledge about the specific domain in which the methods of data science is being applied in. The primary goal that data science achieves is that of extracting meaningful insights from vast amounts of heterogenous data.

Data Science often involves preparing data for analysis and processing, performing advanced data analysis, and presenting the results of to reveal patterns and enable stakeholder to draw informed conclusions.

### b) Explain with an example what you mean by data product

Data products make use of data-related technologies, i.e., databases, cloud architects, algorithms, and software to deliver a virtual product to a consumer in which brings value in the form of information or entertainment to the user. The goal of a data product is to leverage data to deliver only the results of that data to the consumer pf the data product. Therefore, Data products are the results of data, not data itself. Netflix and Spotify, for example, are two of the world's most popular data products. These products leverage the use of data technologies to deliver video and music entertainment, respectively, to the users of its product, all while disguising the actual data that is in the background.

Spotify uses data that is automatically collected from each user to generate new playlists for users by way of a recommendation engine, specifically BART. This algorithm finds patterns in a user's listening behavior and uses that history to continue playing and/or recommending similar songs that the user may also like. In addition, Spotify will continue to play more and more music directly to the user without the need to ever have to open the app to select a new song. By knowing the kind of music the user enjoys, Spotify can continue playing songs for the user, often playing new music that the user never knew that they would enjoy.

### c) Carefully read Cleveland's paper shown in lecture 2 and discuss what he suggested about the field of statistics and data science?

Cleveland's plan suggests how technical resources should be allocated in a university setting to enhance traditional field statistics into a data science curriculum. His plan is focused on the analyst, and how the analyst can benefit from the merging of field statistics, computer science, and mathematical theory. This emergence of fields, topped with the continual development of tools, provides new and efficient ways that analysts can analyze data.

I find it interesting how these ideas in 2001, of how data science can emerge from statistics, is much of a reality in today's world of data science. As we learned in our lecture, data science is the intersection of math & statistics, hacking, and domain expertise. All of these were individually recognized in Cleveland's paper, over 20 years ago. Today, we are continuing to see STEM-related courses in all levels of academy, as we are seeing the field of data science and other technical field grow exponentially.

**d) Explain in a short paragraph how data science differs from computer science**

Data science is more focused on the theoretical applications of data analysis and modeling. Data science is what data scientist do, and data scientists attempt to find value in data. These tasks in data science may involve data visualization, pattern recognition, and statistical modeling. Computer science on the other hand is more focused on development of computational systems and networks. Computer scientists are responsible for inventing new methods of computing. Computer science plays a huge role in what a data scientist needs to analyze data through methods of computing. In order to compute data, data scientists must use the tools, technologies, and architecture developed by computer scientists.

**e) What is data literacy? Is it important to be data literate in this modern world? Explain why or why not.**

In its simplest form, data literacy is a person's ability to translate data into information. It is crucial that the information derived from data can be effectively communicated to stakeholder, who are not as interested in the data, but are interested in the information. It is extremely important to have some sense of data literacy in today's world. Data are everywhere, and data are required to for any form of decision making. Without being data literate, one would find it difficult to make reliable decisions.

**f) In his article, Donoho talked about the Common Task Framework. Explain what it is and why he mentioned it.**

The common task framework, CTF, is type of research paradigm where independent research teams/individuals compete to best solve a common task. The common task is typically to best train a prediction model with the provided data, where the scoring is determined by a referee. It is called the common task framework because each competitor in the research has the same common tasks – to minimize the prediction errors and thus increasing the performance of predictive models.

There are several reasons why the author, Donoho, mentions the Common Task Framework in his publishing. Donoho claims that the advances and accumulation in knowledge that we see in machine learning today, can be either directly or indirectly credited to the CTF research paradigm. Today, there is such a large community in around machine learning and modeling, and statistical researchers should be taking advantage of the CTF paradigm to continue developing better predictive models. Donoho feels as if CTF is not as well recognized as it should be, considering the influence and impact it has made on predictive models as we know them today.

**g) According to Donoho, what are the activities of greater data science?**

Donoho explains that there are six activities of greater data science: Data Exploration and preparation, data representation and transformation, computing with data, data modeling, data visualization and presentation, and science about data science. Donoho feels that traditional academic studies of Data Science only cover a fraction of what is required to full embrace all of greater data science. Academia needs to shift focus from only teaching data modeling, and instead start distributing resources into the other 5 activities.

## 2. What are the very first few steps one should take once data is loaded into R? Demonstrate them by loading tips data from http://www.ggobi.org/book/data/tips.csv

The goal of doing these steps is to understand what the data looks like before working with it. This allows us to have a better idea of what needs to be done to the data, and what can be done with the data in terms of analysis.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
df <- read.csv('http://www.ggobi.org/book/data/tips.csv')

# View the head of the dataframe
head(df, 5)
```

```
##   obs totbill  tip sex smoker day  time size
## 1   1   16.99 1.01   F     No Sun Night    2
## 2   2   10.34 1.66   M     No Sun Night    3
## 3   3   21.01 3.50   M     No Sun Night    3
## 4   4   23.68 3.31   M     No Sun Night    2
## 5   5   24.59 3.61   F     No Sun Night    4
```

```
# View the tail of the dataframe
tail(df, 5)
```

```
##     obs totbill  tip sex smoker day  time size
## 240 240   29.03 5.92   M     No Sat Night    3
## 241 241   27.18 2.00   F    Yes Sat Night    2
## 242 242   22.67 2.00   M    Yes Sat Night    2
## 243 243   17.82 1.75   M     No Sat Night    2
## 244 244   18.78 3.00   F     No Thu Night    2
```

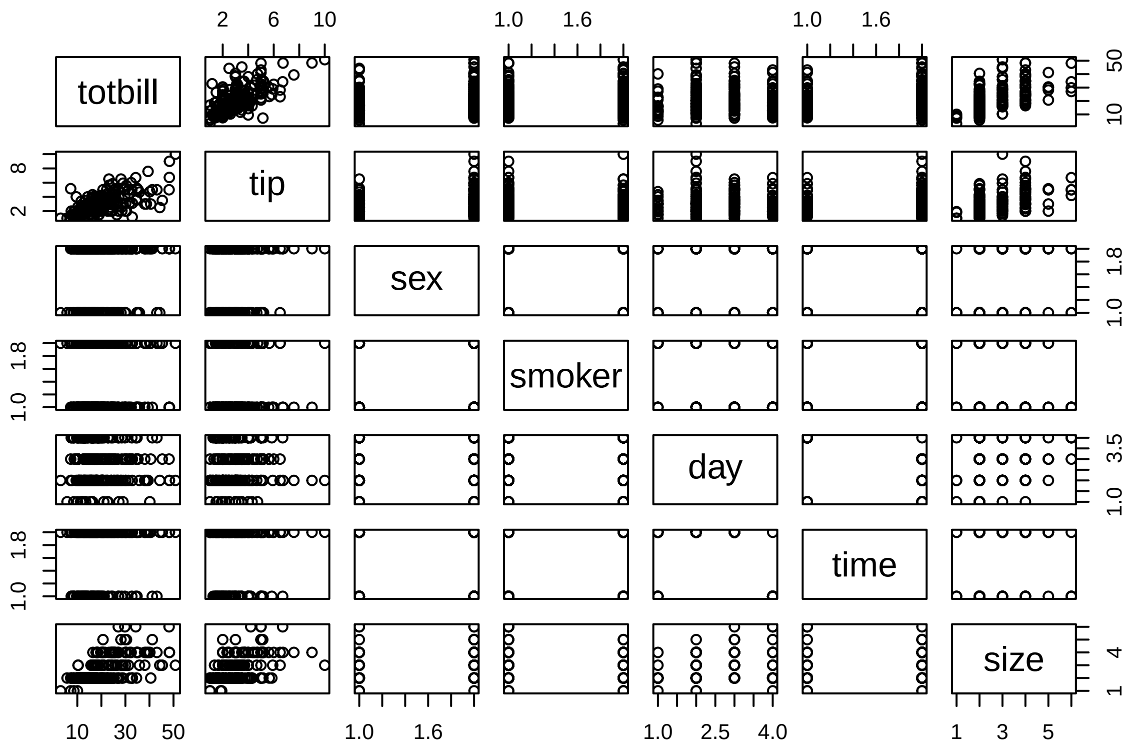
```
# view the numerical summary
summary(df %>% select(-c('obs')))
```

```
##     totbill           tip              sex               smoker
##  Min.   : 3.07   Min.   : 1.000   Length:244         Length:244
##  1st Qu.:13.35   1st Qu.: 2.000   Class :character   Class :character
##  Median :17.80   Median : 2.900   Mode  :character   Mode  :character
##  Mean   :19.79   Mean   : 2.998
##  3rd Qu.:24.13   3rd Qu.: 3.562
##  Max.   :50.81   Max.   :10.000
##      day                time                size
##  Length:244         Length:244         Min.   :1.00
##  Class :character   Class :character   1st Qu.:2.00
##  Mode  :character   Mode  :character   Median :2.00
##                                        Mean   :2.57
##                                        3rd Qu.:3.00
##                                        Max.   :6.00
```

```
# View the internal structure of the dataframe
str(df)
```

```
## 'data.frame':    244 obs. of  8 variables:
##  $ obs    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ totbill: num  17 10.3 21 23.7 24.6 ...
##  $ tip    : num  1.01 1.66 3.5 3.31 3.61 4.71 2 3.12 1.96 3.23 ...
##  $ sex    : chr  "F" "M" "M" "M" ...
##  $ smoker : chr  "No" "No" "No" "No" ...
##  $ day    : chr  "Sun" "Sun" "Sun" "Sun" ...
##  $ time   : chr  "Night" "Night" "Night" "Night" ...
##  $ size   : int  2 3 3 2 4 4 2 4 2 2 ...
```

```
# Pairwise plot of numerical fields
plot(df %>% select(-c('obs')))
```



**3. In our `R` class, we learned about recursive functions that produce a sequence of numbers up to a given number, say n, as demonstrated with the following code:**

```
foo = function(x){
  print(x)
  if(x>1){
    foo(x-1)
  }
}

moo = function(x){
  if(x>1)
    {moo(x-1)}
  print(x)
}

# Which Return:
x=3
foo(x)
```

```
## [1] 3
## [1] 2
## [1] 1
```

```
moo(x)
```

```
## [1] 1
## [1] 2
## [1] 3
```

**Explain why moo() prints 1 through 3 while foo() prints 3 to 1**

**Function moo() explained:** In the function, `moo()`, The initial input value of the function is created, but the function will continue call itself again and again with an input value of 1 less the previous function input before the value of the input is printed. The conditional test of `if(x>1)` must return `FALSE` before the function can continue to the `print(x)` function step. This is the reduction step of recursion. The sequence of the input values must converge (reduce) to the base case, which in this function is 1. The input of moo decreases by 1 until recursion can no longer continue, which is at the base value of 1. Once the input reduces the base value, 1, the base case is printed, then followed by all preceding inputs of moo.

We can trace this computation as follows:

```
moo(3)
..moo(2)
....moo(1)
....print(1)
..print(2)
print(3)
```

This tracing makes it clear to see why the values are returned as 1, 2, 3

**Function foo() Explained:** In the function, `foo()`, the initial input value of the function is created and then printed immediately. If x is greater than 1, then the function will be called again, this time 1 less the previous parameter value. The function is recursive, so it will continue to call itself with a value of 1 less the previous input if and only if the result of the conditional statement returns `TRUE`. The function will continue to call itself until the conditional returns a `FALSE`, thus, not requiring itself to be called again.

The function can be traced as follows:

```
foo(3)
print(3)
...foo(2)
...print(2)
......foo(1)
......print(1)
```

The input of the function is printed before recursion occurs, so the values will be returned as 3, 2, 1 since the input is being reduced by 1 each time the function is re-called.

## 4. The function `sqrt()` provides the square root of a non-negative number. Note what happens when you try `sqrt(-1)`. We want to create a our own function that either finds the square root of a non-negative number or provides a custom message if we pass it a negative number

### a) Create a new `R` function `getRootNotVectorized()` that will return the square root of any non-negative number and 'not possible' for a negative number. Further, `getRootNotVectorized()` should only successfully return 'Not Possible' if the negative value is the first element that you pass to the function. Otherwise, your function should return `NaN` for negative values.

```r
getRootNotVectorized <- function(x){
  if(x >= 0)
    return(sqrt(x))
  else
    return ('Not Possible')
}

getRootNotVectorized(4)
```

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

```r
getRootNotVectorized(-4)
```

```
## [1] "Not Possible"
```

```r
getRootNotVectorized(c(-1, -4))
```

```
## Warning in if (x >= 0) return(sqrt(x)) else return("Not Possible"): the
## condition has length > 1 and only the first element will be used
```

```
## [1] "Not Possible"
```

```
getRootNotVectorized(c(0, 1, -1, 4, -4))
```

```
## Warning in if (x >= 0) return(sqrt(x)) else return("Not Possible"): the
## condition has length > 1 and only the first element will be used

## Warning in sqrt(x): NaNs produced

## [1]   0   1 NaN   2 NaN
```

**b) Now create a second function getRootVectorized() that will return the square root of any non-negative number and 'not possible' for a negative number regardless of the number's position in a numeric vector of arbitrary length.**

```
getRootVectorized <- function(x) {
  ifelse(
    x >= 0,
    sqrt(x),
    'Not Possible'
  )
}

getRootVectorized(4)
```

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

```
getRootVectorized(-4)
```

```
## [1] "Not Possible"
```

```
getRootVectorized(c(-1, -4))
```

```
## [1] "Not Possible" "Not Possible"
```

```
getRootVectorized(c(0, 1, -1, 4, -4))
```

```
## Warning in sqrt(x): NaNs produced

## [1] "0"            "1"            "Not Possible" "2"            "Not Possible"
```

**c) Describe the difference in your code between `getRootNotVectorized()` and `getRootVectorized()` that allowed you to get the desired message output for any negative element of a vector in the latter function but not the former.**

The difference is how the inputs of the function are evaluated in the condition statement. The first function, `getRootNotVectorized()`, makes use of an **if()** function, whereas the second function, `getRootVectorized()`, makes use of an **ifelse()** function. In getRootNotVectorized(), the **if()** function can only evaluate a logical vector of length 1. When passing a vector with multiple elements into the **if()** function, `R` will only evaluate the first element. The **if()** function can only evaluate one element in a vector at one time, and we are attempting to check every element in the vector at once. In getRootVectorized(), the ifelse() function will evaluate each element in a vector at one time.

### d) Why do you see a difference between the output of the following lines of code?

```
is.numeric(getRootVectorized(c(0, 1, 4)))
```

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

```
is.numeric(getRootVectorized(c(0, 1, -4)))
```

```
## Warning in sqrt(x): NaNs produced
```

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

The function, `is.numeric()` will evaluate if every element in a given vector is of type numeric. If all positions in a vector are numeric, then the output of `is.numeric()` will return `TRUE`, otherwise it will return `FALSE`. Since the output of the second line of code contains a string, "Not Possible", in position 3, the output of this vector will return `FALSE`. The input elements in the first line of code, contains all positive integers, therefore each output of this line will result in a numeric value, which is the square root of the input elements.

## 5. This problem will give you some practice with creating and manipulating vectors

### a) Using seq(), create a vector consisting of an arithmetic sequence of integers from 5 to 50 with a common difference of 5 stored in a variable called mySeq. Report mySeq.

```
mySeq = seq(5,50,5)
mySeq
```

```
##  [1]  5 10 15 20 25 30 35 40 45 50
```

### b) Describe how the different arguments in each of the three following commands changes the output of rep(): rep(mySeq, 5), rep(mySeq, each = 5), and rep(mySeq, mySeq).

```
rep(mySeq, 5)
```

```
##  [1]  5 10 15 20 25 30 35 40 45 50  5 10 15 20 25 30 35 40 45 50  5 10 15 20 25
## [26] 30 35 40 45 50  5 10 15 20 25 30 35 40 45 50  5 10 15 20 25 30 35 40 45 50
```

This repeat pattern simply repeats the sequence 5 times in a row

```
rep(mySeq, each=5)
```

```
##  [1]  5  5  5  5  5 10 10 10 10 10 15 15 15 15 15 20 20 20 20 20 25 25 25 25 25
## [26] 30 30 30 30 30 35 35 35 35 35 40 40 40 40 40 45 45 45 45 45 50 50 50 50 50
```

This repeat pattern repeats each element in the sequence 5 times

```r
rep(mySeq, each=5)
```

```
##  [1]  5  5  5  5  5 10 10 10 10 10 15 15 15 15 15 20 20 20 20 20 25 25 25 25 25
## [26] 30 30 30 30 30 35 35 35 35 35 40 40 40 40 40 45 45 45 45 45 50 50 50 50 50
```

This repeat pattern repeats each value by the value of itself. If an element is 15, then the element of 15 will get repeated 15 times.

**c) Concatenate the sequence 1:14 to the end of the vector described by rep(mySeq, mySeq) and store the resulting vector in the same mySeq variable. Report the length of mySeq.**

```r
mySeq = c(1:14, rep(mySeq, mySeq))
mySeqLength = length(mySeq)
mySeqLength
```

```
## [1] 289
```

**d) Create a square matrix populated row-wise from your mySeq vector and store it in a variable called sqMtrx. Report the vector of values described by the column sums of sqMtrx**

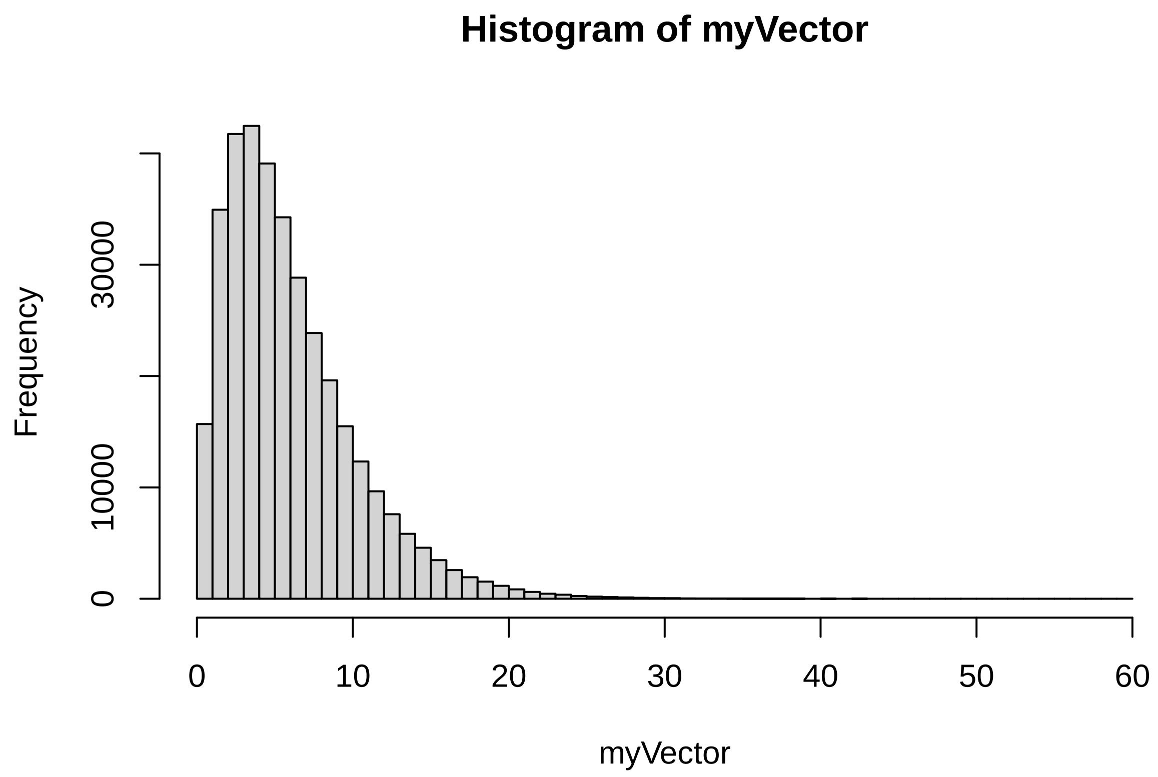
```r
matrix_dimension_size = sqrt(mySeqLength)
sqMtrx = matrix(mySeq, nrow = matrix_dimension_size, ncol = matrix_dimension_size)
colSums(sqMtrx)
```

```
##  [1] 120 185 290 360 425 490 510 595 595 675 680 730 765 765 845 850 850
```

## 6. Write a program that will do the following. Include your codes and necessary outputs to demonstrate your work

**a) Generate 350,000 random numbers from a gamma distribution with shape = 2 and scale = 3 and store these numbers in a vector called myVector. Report a histogram of the numbers you just generated.**

```r
myVector = rgamma(350000, shape=2, scale=3)
hist(myVector, breaks=50)
```



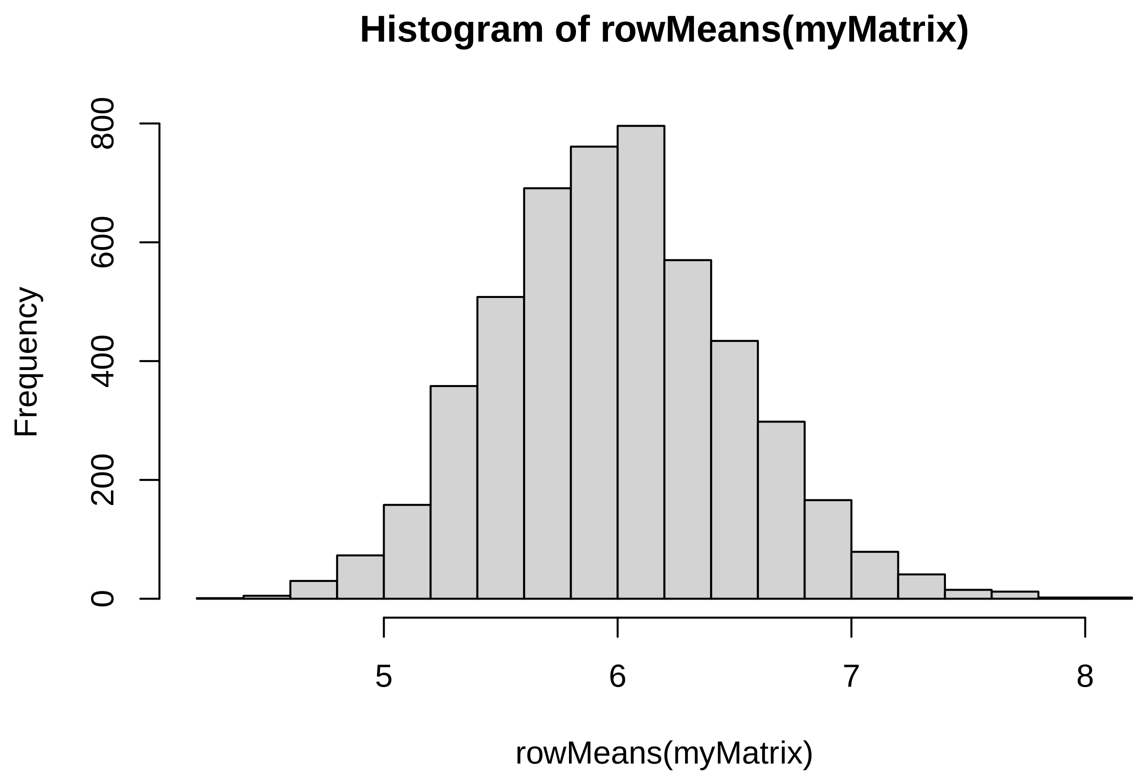
**b) Convert myVector into a matrix with 5,000 rows and assign it to an object called myMatrix. Report the dimensions of myMatrix.**

```
myMatrix = matrix(myVector, nrow=5000)
dim(myMatrix)
```

```
## [1] 5000   70
```

**c) compute the row means of myMatrix and report a histogram of those row means**

```
hist(rowMeans(myMatrix), breaks=15)
```



**d) Explain why the two histograms produced in 6a and 6c have different shapes**

Although the gamma distribution is right skewed, the average of myVector in 6a is 5.99158. This tells me that the majority of data in myVector is going to be distributed around that point. Therefore, once the vector is created into an 5000, 70 dimensional matrix, the rows and column of that matrix would also tend to be distributed around 5.99158. This is why when plotting the row means of that matrix, we see that the average of row tends to be normally distributed around 5.99158

## 7. Perform the following reproducable procedure

```
set.seed(2019)

x = rnorm(1000, mean=100, sd=20)
summary(x)
```

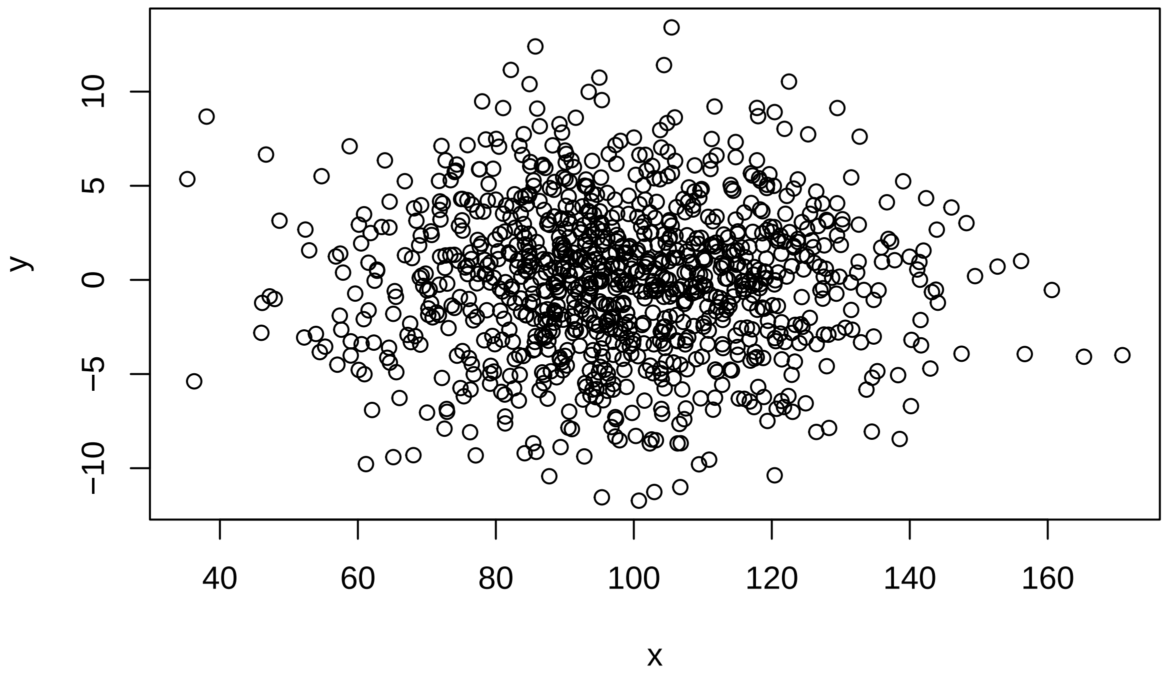
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   35.28   85.45   97.36   98.44  111.97  170.83
```

```
y = rnorm(1000, mean =0, sd=4)
summary(y)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -11.72496  -2.76170   0.07867  -0.02481   2.55656  13.40945
```

```r
df = cbind(x, y)

plot(x, y)
```



```r
knitr::kable(tail(df))
```

| | x | y |
|---|---:|---:|
| [995,] | 94.00851 | -5.823106 |
| [996,] | 87.08630 | -3.073515 |
| [997,] | 81.18893 | -2.066092 |
| [998,] | 121.24816 | -2.850405 |
| [999,] | 84.73008 | 1.825883 |
| [1000,] | 119.20879 | 1.137595 |

## 8. Based on our lecture notes, answer the following questions.

**a) We have a vector of values x = c(2,4,5,“3.5”). What would be the mode of the vector x?**

```r
x = c(2,4,5,'3.5')
mode(x)
```

```
## [1] "character"
```

**b) How do you load a package into `R`? Show that by loading `ggplot2` package.**

library(ggplot2)

**c) How can you check if there is an missing values in a vector?**

```r
y = c(3, 5, 8, NA, 6)
any(is.na(y))
```

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