STAT 206 Lab 2_Lihua Xu

Due Monday, October 16, 5:00 PM

General instructions for labs: You are encouraged to work in pairs to complete the lab. Labs must be completed as an R Markdown file. Be sure to include your lab partner (if you have one) and your own name in the file. Give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Each answer must be supported by written statements as well as any code used.

Agenda: Manipulating data frames; practicing iteration; practicing re-writing code; checking how reliable random methods are.

Part I – Data Frames

R includes a number of pre-specified data objects as part of its default installation. We will load and manipulate one of these, a data frame of 93 cars with model year 1993. Begin by ensuring that you can load this data with the commands

```
library(MASS)
data(Cars93)
```

Begin by examining the data frame with the command `View(Cars93)` to understand the underlying object. You will need to use functions and other commands to extract elements for this assignment.

1. Obtain a `summary()` of the full data structure. Can you tell from this how many rows are in the data? If so, say how; if not, use another method to obtain the number of rows.

```
summary(Cars93)
```

```
##     Manufacturer     Model         Type      Min.Price         Price
##  Chevrolet: 8    100    : 1   Compact:16   Min.   : 6.70   Min.   : 7.40
##  Ford     : 8    190E   : 1   Large  :11   1st Qu.:10.80   1st Qu.:12.20
##  Dodge    : 6    240    : 1   Midsize:22   Median :14.70   Median :17.70
##  Mazda    : 5    300E   : 1   Small  :21   Mean   :17.13   Mean   :19.51
##  Pontiac  : 5    323    : 1   Sporty :14   3rd Qu.:20.30   3rd Qu.:23.30
##  Buick    : 4    535i   : 1   Van    : 9   Max.   :45.40   Max.   :61.90
##  (Other)  :57    (Other):87
##    Max.Price       MPG.city      MPG.highway                  AirBags
##  Min.   : 7.9   Min.   :15.00   Min.   :20.00   Driver & Passenger:16
##  1st Qu.:14.7   1st Qu.:18.00   1st Qu.:26.00   Driver only       :43
##  Median :19.6   Median :21.00   Median :28.00   None              :34
##  Mean   :21.9   Mean   :22.37   Mean   :29.09
##  3rd Qu.:25.3   3rd Qu.:25.00   3rd Qu.:31.00
##  Max.   :80.0   Max.   :46.00   Max.   :50.00
##
##  DriveTrain  Cylinders    EngineSize      Horsepower         RPM
##  4WD  :10   3     : 3   Min.   :1.000   Min.   : 55.0   Min.   :3800
##  Front:67   4     :49   1st Qu.:1.800   1st Qu.:103.0   1st Qu.:4800
##  Rear :16   5     : 2   Median :2.400   Median :140.0   Median :5200
##             6     :31   Mean   :2.668   Mean   :143.8   Mean   :5281
##             8     : 7   3rd Qu.:3.300   3rd Qu.:170.0   3rd Qu.:5750
##             rotary: 1   Max.   :5.700   Max.   :300.0   Max.   :6500
##
```

```
##   Rev.per.mile  Man.trans.avail Fuel.tank.capacity   Passengers
##  Min.   :1320   No :32          Min.   : 9.20      Min.   :2.000
##  1st Qu.:1985   Yes:61          1st Qu.:14.50      1st Qu.:4.000
##  Median :2340                   Median :16.40      Median :5.000
##  Mean   :2332                   Mean   :16.66      Mean   :5.086
##  3rd Qu.:2565                   3rd Qu.:18.80      3rd Qu.:6.000
##  Max.   :3755                   Max.   :27.00      Max.   :8.000
##
##      Length        Wheelbase         Width        Turn.circle
##  Min.   :141.0   Min.   : 90.0   Min.   :60.00   Min.   :32.00
##  1st Qu.:174.0   1st Qu.: 98.0   1st Qu.:67.00   1st Qu.:37.00
##  Median :183.0   Median :103.0   Median :69.00   Median :39.00
##  Mean   :183.2   Mean   :103.9   Mean   :69.38   Mean   :38.96
##  3rd Qu.:192.0   3rd Qu.:110.0   3rd Qu.:72.00   3rd Qu.:41.00
##  Max.   :219.0   Max.   :119.0   Max.   :78.00   Max.   :45.00
##
##  Rear.seat.room   Luggage.room       Weight         Origin
##  Min.   :19.00   Min.   : 6.00   Min.   :1695   USA    :48
##  1st Qu.:26.00   1st Qu.:12.00   1st Qu.:2620   non-USA:45
##  Median :27.50   Median :14.00   Median :3040
##  Mean   :27.83   Mean   :13.89   Mean   :3073
##  3rd Qu.:30.00   3rd Qu.:15.00   3rd Qu.:3525
##  Max.   :36.00   Max.   :22.00   Max.   :4105
##  NA's   :2       NA's   :11
##             Make
##  Acura Integra: 1
##  Acura Legend : 1
##  Audi 100     : 1
##  Audi 90      : 1
##  BMW 535i     : 1
##  Buick Century: 1
##  (Other)      :87
```

```
# I cannot tell the number of rows and coluns in the data.
dim(Cars93)
```

```
## [1] 93 27
```

```
# Using the dimension command, the data has 93 rows and 27 columns.
```

2. What is the mean price of a car with a rear-wheel drive train?

```
data_price <- Cars93[Cars93$DriveTrain=="Rear",]$Price
data_price
```

```
##  [1] 30.0 23.7 15.1 18.8 38.0 15.9 20.9 47.9 35.2 36.1 32.5 31.9 61.9 14.9
## [15] 17.7 22.7
```

```
mean_price <- mean(data_price)
mean_price
```

```
## [1] 28.95
```

```
#The mean price of a car with a rear-wheel drive train is 28.95.
```

3. What is the minimum horsepower of all cars with capacity for 7 passengers? With a capacity of at least 6 passengers?

```
#horsepower of all cars with capacity for 7 passengers
Hosepower_7 <- Cars93[Cars93$Passengers==7,]$Horsepower
Hosepower_7
```

```
## [1] 170 142 145 155 151 170 138 109
```

```
Minimum_horsepower_7 <- min(Hosepower_7)
Minimum_horsepower_7
```

```
## [1] 109
```

```
#The minimum horsepower of all cars with capacity for 7 passengers is 109.

#horsepower of all cars with at least capacity for 7 passengers
Hosepower_6 <- Cars93[Cars93$Passengers>=6,]$Horsepower
Hosepower_6
```

```
##  [1] 172 110 170 180 200 110 170 165 170 153 141 147 100 142 100 214 145
## [18] 190 160 210 155 151 170 170 170 138 109
```

```
Minimum_horsepower_6 <- min(Hosepower_6)
Minimum_horsepower_6
```

```
## [1] 100
```

```
#The minimum horsepower of all cars with a capacity of at least 6 passengers is 100.
```

4. Assuming that these cars are exactly as fuel efficient as this table indicates, find the cars that have the maximum, minimum and median distance travellable for highway driving. You will need at least two columns to work this out; why those two?

```
#For calculating the distance, we need two columns,
#one is the "MPG.highway",
#which is related to the miles per gallon of gas
#when the car is running on the highway.
#Another column we need is the "Fuel.tank.capacity,
#which informs us about the size of the tank for filling the gas.
#(The unit for the tank should be gallon.)
distance_highway <- Cars93$MPG.highway * Cars93$Fuel.tank.capacity
distance_highway
```

```
##  [1] 409.2 450.0 439.4 548.6 633.0 508.4 504.0 575.0 507.6 450.0 500.0
## [12] 547.2 530.4 434.0 478.5 460.0 540.0 598.0 500.0 504.0 448.0 416.0
## [23] 435.6 406.0 432.0 420.0 432.0 475.2 435.6 504.0 330.0 396.0 429.3
## [34] 446.6 465.0 420.0 480.0 520.0 530.0 446.4 492.9 547.4 527.0 392.7
## [45] 397.3 404.6 464.4 495.0 444.0 473.8 478.4 520.0 488.4 522.0 527.0
## [56] 470.4 500.0 420.5 462.5 288.6 468.0 435.6 456.0 435.6 477.0 460.0
## [67] 481.0 471.2 511.5 460.0 504.0 477.0 541.2 471.2 434.0 445.5 504.0
## [78] 468.0 486.4 340.4 477.0 477.0 455.8 440.3 508.8 536.5 435.6 409.2
## [89] 443.1 555.0 462.5 442.4 540.4
```

```
max_distance <- max(distance_highway)
max_distance
```

```
## [1] 633
```

```
min_distance <- min(distance_highway)
min_distance
```

```
## [1] 288.6
```

```
median_distance <- median(distance_highway)
median_distance
```

```
## [1] 470.4
```

```
#The maximum, minimum and median distance travellable
#for highway driving are 633, 288.6 and 470.4 respectively.
```

Part II – Reproducibility and Functions

Some of the lectures have included examples of planning production for a factory that turns steel and labor into cars and trucks. Below is a piece of code that optimizes the factory's output (roughly) given the available resources, using a `repeat` loop. It's embedded in a function to make it easier for you to run.

```
factory.function <- function (cars.output=1, trucks.output=1) {
  factory <- matrix(c(40,1,60,3),nrow=2,
    dimnames=list(c("labor","steel"),c("cars","trucks")))
  available <- c(1600,70); names(available) <- rownames(factory)
  slack <- c(8,1); names(slack) <- rownames(factory)
  output <- c(cars.output, trucks.output); names(output) <- colnames(factory)

  passes <- 0 # How many times have we  been around the loop?
  repeat {
     passes <- passes + 1
     needed <- factory %*% output # What do we need for that output level?
     # If we're not using too much, and are within the slack, we're done
     if (all(needed <= available) &&
         all((available - needed) <= slack)) {
       break()
     }
     # If we're using too much of everything, cut back by 10%
     if (all(needed > available)) {
       output <- output * 0.9
       next()
     }
     # If we're using too little of everything, increase by 10%
     if (all(needed < available)) {
       output <- output * 1.1
       next()
     }
     # If we're using too much of some resources but not others, randomly
     # tweak the plan by up to 10%
      # runif == Random number, UNIFormly distributed, not "run if"
     output <- output * (1+runif(length(output),min=-0.1,max=0.1))
  }

  return(output)
}
```

5. Run the function above with the command

```
factory.function()
```

```
##     cars   trucks
## 10.22242 19.74999
```

to obtain a default output value, starting from a very low initial planned output. What is the final output capacity obtained?

```
factory.function()
```

```
##     cars   trucks
## 10.25208 19.75304
```

```
#The final output capacity are as above values.
```

6. Repeat this four more times to obtain new output values. Do these answers differ from each other? If so why? If not, why not?

```
factory.function()
```

```
##      cars    trucks
##  9.905872 19.941354
```

```
factory.function()
```

```
##      cars    trucks
##  9.666259 20.102574
```

```
factory.function()
```

```
##     cars   trucks
## 10.39423 19.73254
```

```
factory.function()
```

```
##     cars   trucks
## 10.40352 19.68544
```

```
#Because there are a lot of randomness inside the function.
```

7. Right now, the number of `passes` is a value held within the function itself and not shared. Change the code so that the number of `passes` will be returned at the end of the function, as well as the final `output`.

```
factory.function <- function (cars.output=1, trucks.output=1) {
  factory <- matrix(c(40,1,60,3),nrow=2,
    dimnames=list(c("labor","steel"),c("cars","trucks")))
  available <- c(1600,70); names(available) <- rownames(factory)
  slack <- c(8,1); names(slack) <- rownames(factory)
  output <- c(cars.output, trucks.output); names(output) <- colnames(factory)

  passes <- 0 # How many times have we  been around the loop?
  repeat {
     passes <- passes + 1
     needed <- factory %*% output # What do we need for that output level?
     # If we're not using too much, and are within the slack, we're done
     if (all(needed <= available) &&
         all((available - needed) <= slack)) {
       break()
     }
     # If we're using too much of everything, cut back by 10%
     if (all(needed > available)) {
```

```
      output <- output * 0.9
      next()
    }
    # If we're using too little of everything, increase by 10%
    if (all(needed < available)) {
      output <- output * 1.1
      next()
    }
    # If we're using too much of some resources but not others, randomly
    # tweak the plan by up to 10%
     # runif == Random number, UNIFormly distributed, not "run if"
    output <- output * (1+runif(length(output),min=-0.1,max=0.1))
  }

  multiple_values=c(output,Passes_num=passes)
  return(multiple_values)
}
```

```
factory.function()
```

```
##       cars     trucks Passes_num
##   10.11474   19.83383  394.00000
```

8. Now, set the initial output levels to 30 cars and 20 trucks and run the code. What is the final output plan (`output`)? What is the final demand for resources (`needed`)? Is the plan within budget and within the slack? How many iterations did it take to converge (`passes`)? For all but `output` you will need to either print this message out deliberately, or return an object that contains all the quantities you want.

```
factory.function <- function (cars.output=30, trucks.output=20) {
  factory <- matrix(c(40,1,60,3),nrow=2,
    dimnames=list(c("labor","steel"),c("cars","trucks")))
  available <- c(1600,70); names(available) <- rownames(factory)
  slack <- c(8,1); names(slack) <- rownames(factory)
  output <- c(cars.output, trucks.output); names(output) <- colnames(factory)

  passes <- 0 # How many times have we  been around the loop?
  repeat {
     passes <- passes + 1
     needed <- factory %*% output # What do we need for that output level?
     # If we're not using too much, and are within the slack, we're done
     if (all(needed <= available) &&
         all((available - needed) <= slack)) {
       break()
     }
     # If we're using too much of everything, cut back by 10%
     if (all(needed > available)) {
       output <- output * 0.9
       next()
     }
     # If we're using too little of everything, increase by 10%
     if (all(needed < available)) {
       output <- output * 1.1
       next()
     }
     # If we're using too much of some resources but not others, randomly
```

```
    # tweak the plan by up to 10%
     # runif == Random number, UNIFormly distributed, not "run if"
    output <- output * (1+runif(length(output),min=-0.1,max=0.1))
  }


  multiple_values=c(output,demand_resources=needed,Passes_num=passes)
  return(multiple_values)
}
```

```
factory.function()
```

```
##              cars            trucks demand_resources1 demand_resources2
##          10.25035          19.71206        1592.73762          69.38654
##        Passes_num
##         533.00000
```

```
#The result are as the above values.
#The values under "cars"" and "trucks" are regarding to the final output plan.
#The values under "demand_resources1" and "demand_resources2" are
#regarding to the final demand for resources cars and trucks respectively.
#The values under "Passes_num" is regarding to the number of iterations .
#The plan is within budget and within the slack.
```