lab2

Лабораторная работа 2

Исходные данные

Установленная среда R включает в себя набор заранее сконфигурированных массивов данных. В данной лабораторной требуется использовать одну из них — датафрейм из 93 машин-новинок 1993 года. Для того, чтобы загрузить данный датафрейм, необходимо выполнить следующие команды:

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

Для того, чтобы ознакомиться со структурой и составом данных, можно выполнить команду `View(Cars93)` или нажать на соответствующую кнопку в RStudio.

```
View(Cars93)
head(Cars93)
```

```
##   Manufacturer   Model    Type Min.Price Price Max.Price MPG.city
## 1        Acura Integra   Small      12.9  15.9      18.8       25
## 2        Acura  Legend Midsize      29.2  33.9      38.7       18
## 3         Audi      90 Compact      25.9  29.1      32.3       20
## 4         Audi     100 Midsize      30.8  37.7      44.6       19
## 5          BMW    535i Midsize      23.7  30.0      36.2       22
## 6        Buick Century Midsize      14.2  15.7      17.3       22
##   MPG.highway            AirBags DriveTrain Cylinders EngineSize
## 1          31               None      Front         4        1.8
## 2          25 Driver & Passenger      Front         6        3.2
## 3          26        Driver only      Front         6        2.8
## 4          26 Driver & Passenger      Front         6        2.8
## 5          30        Driver only       Rear         4        3.5
## 6          31        Driver only      Front         4        2.2
##   Horsepower  RPM Rev.per.mile Man.trans.avail Fuel.tank.capacity
## 1        140 6300         2890             Yes               13.2
## 2        200 5500         2335             Yes               18.0
## 3        172 5500         2280             Yes               16.9
## 4        172 5500         2535             Yes               21.1
## 5        208 5700         2545             Yes               21.1
## 6        110 5200         2565              No               16.4
##   Passengers Length Wheelbase Width Turn.circle Rear.seat.room
## 1          5    177       102    68          37           26.5
## 2          5    195       115    71          38           30.0
## 3          5    180       102    67          37           28.0
## 4          6    193       106    70          37           31.0
## 5          4    186       109    69          39           27.0
## 6          6    189       105    69          41           28.0
##   Luggage.room Weight  Origin          Make
## 1           11   2705 non-USA Acura Integra
## 2           15   3560 non-USA  Acura Legend
## 3           14   3375 non-USA       Audi 90
## 4           17   3405 non-USA      Audi 100
## 5           13   3640 non-USA      BMW 535i
## 6           16   2880     USA Buick Century
```

Задание 1

1. Выполните команду `summary()` на полном наборе данных. Можно ли по результату выполнения сказать сколько строк в датафрейме? Если да, напишите сколько. Если нет, то приведите другой способ.

```
summary(Cars93) #summary is a generic function used to produce result summaries of the results o
f various model fitting functions
```

```
##     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
```

```
nrow(Cars93) #nrow and ncol return the number of rows or columns present in x
```

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

2. Найдите среднюю цену машин с задним приводом.

```
#DriveTrain: Front, Rear, 4WD
mean(subset(Cars93$Price, Cars93$DriveTrain=="Rear"))
```

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

3. Найдите минимальное число лошадиных сил автомобиля для 7 пассажиров. Для 6 пассажиров.

```
min(subset(Cars93$Horsepower,Cars93$Passengers=="7"))
```

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

```
min(subset(Cars93$Horsepower,Cars93$Passengers=="6"))
```

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

4. Найдите машины с максимальным, минимальным и средним(медианой) расстоянием, которая машина может проехать по трассе. Вам понадобятся 2 колонки, чтобы рассчитать расстояние. Какие?

MPG.highway - расход топлива Fuel.tank.capacity - объем бака

```
miles <- Cars93$MPG.highway * Cars93$Fuel.tank.capacity;

Cars93$Make[which(miles == max(miles))]
```

```
## [1] BMW 535i
## 93 Levels: Acura Integra Acura Legend Audi 100 Audi 90 ... Volvo 850
```

```
Cars93$Make[which(miles == min(miles))]
```

```
## [1] Mercury Capri
## 93 Levels: Acura Integra Acura Legend Audi 100 Audi 90 ... Volvo 850
```

```
Cars93$Make[which(miles == median(miles))]
```

```
## [1] Mazda MPV
## 93 Levels: Acura Integra Acura Legend Audi 100 Audi 90 ... Volvo 850
```

Задание 2

В самом начале занятий приводился пример с фабрикой и производством автомобилей. Ниже приведён пример кода, который старается оптимизировать выпуск продукции ориентируясь на доступные ресурсы.

```
factory.run <- function (o.cars=1, o.trucks=1) {
  factory <- matrix(c(40,1,60,3),nrow=2, dimnames=list(c("workday","steel"),c("car","truck")))
  warehouse <- c(1600,70) #Доступно материалов на складе
  names(warehouse) <- rownames(factory)
  reserve <- c(8,1)
  names(reserve) <- rownames(factory)
  output <- c(o.cars, o.trucks)
  names(output) <- colnames(factory)

  steps <- 0 # Счётчик числа шагов цикла
  repeat {
    steps <- steps + 1
    needed <- factory %*% output # Подсчитаем ресурсы, которые нам нужны для производства требуе
мого кол-ва машин
    #message(steps)
    #print(needed)

    # Если ресурсов достаточно и остаток меньше или равен резерву, то мы произвели максимум возм
ожного.
    # Нужно прекращать
    if (all(needed <= warehouse) && all((warehouse - needed) <= reserve)) {
      break()
    }
    # Если заявка слишком большая и ресурсов недостаточно, уменьшим её на 10%
    if (all(needed > warehouse)) {
      output <- output * 0.9
      next()
    }
    # Если всё наоброт, то увеличим на 10%
    if (all(needed < warehouse)) {
      output <- output * 1.1
      next()
    }
    # Если мы потребили одного ресурса слишком много, а другого недостаточно,
    # то увеличим план на случайную величину
    output <- output * (1+runif(length(output),min=-0.1,max=0.1))
  }

  return(output)
}
```

1. Выполните код и запустите эту функцию `factory.run()`.

```
factory.run()
```

```
##      car    truck
## 10.20661 19.78173
```

Закомментирован промежуточный вывод какждого шага

2. С каким входными значениями функция вызвана? Какой получился результат?

Функция вызвана с входными значениями по умолчанию o.cars=1, o.trucks=1 -> нужно изготовить одну машину и один грузовик Результат получился другим, так как эта функция согласует входные параметры с доступными ресурсами

3. Повторите вызов 4 раза. Полученные ответы отличаются от полученных ранее? Если да, почему? Если нет, почему?

```
factory.run()
```

```
##      car    truck
## 10.05301 19.93837
```

```
factory.run()
```

```
##       car     truck
##  9.878768 19.997114
```

```
factory.run()
```

```
##      car    truck
## 10.27357 19.75646
```

```
factory.run()
```

```
##      car    truck
## 10.08720 19.80934
```

Результаты отличаются, так как в коде используется `runif` *(#Если мы потребили одного ресурса слишком много, а другого недостаточно, то увеличим план на случайную величину)*

4. В приведённом коде, переменные *steps* и *output* находятся внутри алгоритма. Измените функцию так, чтобы она возвращала число шагов и произведённое количество машин.

```
factory.run2 <- function (o.cars=1, o.trucks=1) {
 factory <- matrix(c(40,1,60,3),nrow=2, dimnames=list(c("workday","steel"),c("car","truck")))
  warehouse <- c(1600,70) #Доступно материалов на складе
  names(warehouse) <- rownames(factory)
  reserve <- c(8,1)
  names(reserve) <- rownames(factory)
  output <- c(o.cars, o.trucks)
  names(output) <- colnames(factory)

  steps <- 0 # Счётчик числа шагов цикла
  repeat {
    steps <- steps + 1
    names(steps) <- "steps";
    needed <- factory %*% output # Подсчитаем ресурсы, которые нам нужны для производства требуе
мого кол-ва машин
    # Если ресурсов достаточно и остаток меньше или равен резерву, то мы произвели максимум возм
ожного.
    # Нужно прекращать
    if (all(needed <= warehouse) && all((warehouse - needed) <= reserve)) {
      break()
    }
    # Если заявка слишком большая и ресурсов недостаточно, уменьшим её на 10%
    if (all(needed > warehouse)) {
      output <- output * 0.9
      next()
    }
    # Если всё наоброт, то увеличим на 10%
    if (all(needed < warehouse)) {
      output <- output * 1.1
      next()
    }
    # Если мы потребили одного ресурса слишком много, а другого недостаточно,
    # то увеличим план на случайную величину
    output <- output * (1+runif(length(output),min=-0.1,max=0.1))
  }

  return(list(steps,needed,output)) #если с, то пропадает структура(заголовки трудодни + сталь)
}
factory.run2()
```

```
## [[1]]
## steps
##   651
##
## [[2]]
##               [,1]
## workday 1599.82972
## steel     69.70016
##
## [[3]]
##      car    truck
## 10.29133 19.80294
```

5. Установите план равный тридцати автомобилям и 20 грузовикам и выполните функцию.

```
factory.run2(30,20)
```

```
## [[1]]
## steps
##   325
##
## [[2]]
##               [,1]
## workday 1598.07070
## steel     69.21916
##
## [[3]]
##      car    truck
## 10.68438 19.51159
```

1. Какой получили результат?

Результат не изменился, так как не изменилось количество ресурсов

2. Каким получился итоговый запрос ресурсов (переменная *needed*)
3. Как много итераций пришлось пройти, чтобы получить ответ (переменная *steps*)?
4. Для подсчёта можно пользоваться функциями печати (`print` , `message`) или вернуть результат из функции.