Homework_3

Group 16

9 February 2019

The Linear Program

Part 1.

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	4.9%	2.3%	2.4%	3.9%	4.4%	4.6%	2.6%	1.9%	3.7%	2.6%
Variables	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}

Solutions & RHS in equations would be in millions (**M**)

Constraints:

$$r_1 + r_2 + r_3 + r_4 + r_5 + r_6 + r_7 + r_8 + r_9 + r_{10} = 10$$

$$-r_1 - r_2 + r_5 + r_{10} >= 0$$

$$2r_3 + 2r_4 - r_5 - r_6 - r_7 - r_8 - r_9 <= 0$$

$r_i >= 3$ where $i = 1, 2, 3 ... 10$

Objective:

$0.049r_1 + 0.023r_2 + 0.024r_3 + 0.039r_4 + 0.044r_5 + 0.046r_6 + 0.026r_7 + 0.019r_8 + 0.037r_9 + 0.026r_{10}$

Part 2.

```
library(lpSolve)
```

```
## Warning: package 'lpSolve' was built under R version 3.4.4
```

```
c=c(0.031,0.049,0.024,0.039,0.016,0.024,0.046,0.026,0.033,0.044)
A=matrix(0,13,10)
A[1,]=1
A[2,c(5,10)]=-1
A[2,c(1,2)]=1
A[3,c(3,4)]=2
A[3,seq(from=5,to=9,by=1)]=-1
A[4:13,]=diag(10)
b=c(10,0,0,rep(3,10))
dir=c('=',rep("<=",12))
alc1=lp("max",c,A,dir,b, compute.sens = 1)
print("The solution and maximised value are given below in order: (alc1)")
```

```
## [1] "The solution and maximised value are given below in order: (alc1)"
```

```
alc1$solution
```

```
##  [1] 0 3 0 1 0 0 3 0 0 3
```

```
alc1$objval
```

```
## [1] 0.456
```

Part 3.

Defined the function in separate R script

Test

```
source("allocation_g16.R")
ROI_vec=c(0.031,0.049,0.024,0.039,0.016,0.024,0.046,0.026,0.033,0.044)
upper_bound=3
budget=10
result = allocation(ROI_vec, upper_bound, budget)
result$objval
```

```
## [1] 0.456
```

```
result$sol
```

```
##  [1] 0 3 0 1 0 0 3 0 0 3
```

Part 4

```
source("allocation_g16.R")
ROI_vec=c(0.031,0.049,0.024,0.039,0.016,0.024,0.046,0.026,0.033,0.044)
alc2 = allocation(ROI_vec,budget=10)
alc2$objval
```

```
## [1] 0.465
```

```
alc2$sol
```

```
##  [1] 0 5 0 0 0 0 0 0 0 5
```

alc1 = [0 3 0 1 0 0 3 0 0 3]
alc2 = [0 5 0 0 0 0 0 0 0 5]

The Optimizer’s curse

Part 1

```
ROI_vec=c(0.049,0.023,0.024,0.039,0.044,0.046,0.026,0.019,0.037,0.026)
upper_bound=3
alc3 = allocation(ROI_vec,upper_bound,budget=10)
alc3$objval
```

```
## [1] 0.456
```

```
alc3$sol
```

```
##  [1] 3 0 0 1 3 3 0 0 0 0
```

alc1 = [0 3 0 1 0 0 3 0 0 3]
v1 = 0.456

alc2 = [0 5 0 0 0 0 0 0 0 5]
v2 = 0.465

alc3 = [3 0 0 1 3 3 0 0 0 0]
v3 = 0.456

We see that the value of objective function remains the same as with the other ROI vector but the value of optimal investments change.

Part 2

Commenting on advice of CMO

Disappointment in allocation1 vs allocation 2 is **0.009**

```
cat("Sensitivity values for alc1: ", alc1$duals)
```

```
## Sensitivity values for alc1:  0.039 0 0 0 0.01 0 0 0 0 0.007 0 0 0.005 -0.008 0 -0.015 0 -0.02
3 -0.015 0 -0.013 -0.006 0
```

According to the Optimal solution(with $3M constraint), we are putting in the money for channels: TV, Adwords, Instagram, Email. Now, when we carried out sensitivity analysis, we concluded following from the value of duals obtained for those 4 channels:
a) there’s indeed no gain in return if we increase the constraint from 3M in the case of Adwords. Here, CMO was spot on.
b) The value of objective increases (i.e. non-zero duals for constraints that put a cap on Email, Instagram, TV) if more budget is allowed to be allocated in these three channels.

Hence, we can conclude that more profit can be made if we allocate more budget to TV, Email and Instagram. So, the CMO’s advice didn’t really help the company make the best decision.

Part 3

We first find the maximum of average returns from alc1, alc2 and alc3.

```
dt = data.frame(upper_bound =0,result = 1)

ROI_vec1=c(0.031,0.049,0.024,0.039,0.016,0.024,0.046,0.026,0.033,0.044)
ROI_vec2=c(0.049,0.023,0.024,0.039,0.044,0.046,0.026,0.019,0.037,0.026)

value_to_compare1 = (alc1$objval + sum(alc1$sol*ROI_vec2))/2
value_to_compare2 = (alc2$objval + sum(alc2$sol*ROI_vec2))/2
value_to_compare3 = (alc3$objval + sum(alc3$sol*ROI_vec1))/2
value_to_compare = max(value_to_compare1,value_to_compare2,value_to_compare3)
```

So, we need to find an allocation which will give us average returns more than 0.36
For this we will alter the value of upper ound from 1 to 5 in steps of 0.1 and check the average value of returns from ROI vector 1 and 2.

```
final_allocation = c()
for (up_bound in seq(1, 5, by=0.1)){
    alc_a = allocation(ROI_vec1,up_bound,budget=10)
    x = alc_a$objval
    y = sum(alc_a$sol*ROI_vec2)
    result = (x+y)/2
    if (result > value_to_compare){final_allocation = alc_a$sol; break}
    else
    {
      alc_b = allocation(ROI_vec2,up_bound,budget=10)
      y = alc_b$objval
      x = sum(alc_b$sol*ROI_vec1)
      result = (x+y)/2
      if (result > value_to_compare){final_allocation = alc_b$sol; break}
    }
    dt = rbind(dt, c(up_bound, result))
}

final_allocation
```

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

```
result
```

```
## [1] 0.362
```

Multi Period Allocation

Part 1

```
#install.packages("miceadds")
#library(miceadds)
load('Project1.Rdata')
allocation_monthly_table = matrix(0,nrow = 12, ncol = 10)
retrns = 0.0
tot_rtns = 0.0
budg = 10.0
ROI_mat = ROI_mat/100
for (i in 1:12){
  cat("i is : ",i,'\n')
  roi_vec = ROI_mat[i, ]
  upper_bnd = 3
  alc = allocation(roi_vec,upper_bnd,budget=budg)
  cat("objective val is ", alc$objval,'\n')
  allocation_monthly_table[i, ] = alc[[2]]
  cat("alloc is: ", alc$sol,'\n')
  retrns = (alc[[1]]/2)
  budg = budg + retrns
  if(i < 12)
  {
    tot_rtns = tot_rtns + retrns
  }
  else
  {
    tot_rtns = tot_rtns + retrns*2
  }
}
```

```
## i is :  1
## objective val is  0.373
## alloc is:  3 0 0 1.333333 0 0 2.666667 0 0 3
## i is :  2
## objective val is  0.406296
## alloc is:  3 0 0 2.3955 3 0 0 0 1.791 0
## i is :  3
## objective val is  0.414417
## alloc is:  0 0 0 3 0 3 1.389648 0 3 0
## i is :  4
## objective val is  0.4144868
## alloc is:  0 0 0 3 0 3 3 0 1.596856 0
## i is :  5
## objective val is  0.4321435
## alloc is:  1.8041 0 0 0 0 0 3 0 3 3
## i is :  6
## objective val is  0.4547665
## alloc is:  3 0 0 0 0 0 3 0 2.020172 3
## i is :  7
## objective val is  0.4686546
## alloc is:  1.123777 0 0 3 1.123777 0 3 0 3 0
## i is :  8
## objective val is  0.4879661
## alloc is:  3 0 0 1.827294 0 0.6545882 0 0 3 3
## i is :  9
## objective val is  0.4592199
## alloc is:  1.362933 0 0 3 0 3 0 0 3 1.362933
## i is :  10
## objective val is  0.4275752
## alloc is:  0 0 0 3 0 3 3 0 0 2.955475
## i is :  11
## objective val is  0.5173756
## alloc is:  3 0 0 2.056421 0 1.112842 3 0 0 3
## i is :  12
## objective val is  0.5168342
## alloc is:  3 3 0 0.4279507 3 0 0 0 0 3
```

```
print(tot_rtns)
```

```
## [1] 2.944785
```

```
print(allocation_monthly_table)
```

```
##           [,1] [,2] [,3]      [,4]     [,5]      [,6]     [,7] [,8]
##  [1,] 3.000000    0    0 1.3333333 0.000000 0.0000000 2.666667    0
##  [2,] 3.000000    0    0 2.3955000 3.000000 0.0000000 0.000000    0
##  [3,] 0.000000    0    0 3.0000000 0.000000 3.0000000 1.389648    0
##  [4,] 0.000000    0    0 3.0000000 0.000000 3.0000000 3.000000    0
##  [5,] 1.804100    0    0 0.0000000 0.000000 0.0000000 3.000000    0
##  [6,] 3.000000    0    0 0.0000000 0.000000 0.0000000 3.000000    0
##  [7,] 1.123777    0    0 3.0000000 1.123777 0.0000000 3.000000    0
##  [8,] 3.000000    0    0 1.8272941 0.000000 0.6545882 0.000000    0
##  [9,] 1.362933    0    0 3.0000000 0.000000 3.0000000 0.000000    0
## [10,] 0.000000    0    0 3.0000000 0.000000 3.0000000 3.000000    0
## [11,] 3.000000    0    0 2.0564210 0.000000 1.1128419 3.000000    0
## [12,] 3.000000    3    0 0.4279507 3.000000 0.0000000 0.000000    0
##           [,9]    [,10]
##  [1,] 0.000000 3.000000
##  [2,] 1.791000 0.000000
##  [3,] 3.000000 0.000000
##  [4,] 1.596856 0.000000
##  [5,] 3.000000 3.000000
##  [6,] 2.020172 3.000000
##  [7,] 3.000000 0.000000
##  [8,] 3.000000 3.000000
##  [9,] 3.000000 1.362933
## [10,] 0.000000 2.955475
## [11,] 0.000000 3.000000
## [12,] 0.000000 3.000000
```

Part 2

We were able to solve this multi period allocation using the same function as used for single period allocation since returns from all the medias were positive and allocation in a month was independent of the allocation of other months. This implied that maximing the returns for whole year was same as maximising the returns for each month individually.

Part 3

If the allocations in a month depend on allocations in the previous month, the problem can no longer be solved indivually for each month. In this case, it would be necessary to define a matrix A for contraints, c as the objective function, and multiple decision variables.
Decision variables
There will be a total of (120 + 12) DVs. 120 for investment in 10 medias over a period of 12 months and remaining 12 are the returns from each month.
Contraints
We will have the three contraints from part 1 for each month. This will give us 3*12 = 36 contraints. We will have 12 more equalities for defining the returns from each month. And 11 stability constraints on each media implying 11*12 = 132 more contraints. Summing these up gives a total of 180 constraints.

This means that our A matrix will be of dimension (180*132).