Assignment_3

Abhishek_Sau

09/10/2021

R Markdown

:

Assignment 3 - The Weigelt Corporation has three branch plants with excess production capacity. Fortunately, the corporation has a new product ready to begin production, and all three plants have this capability, so some of the excess capacity can be used in this way. This product can be made in three sizes–large, medium, and small–that yield a net unit profit of $420, $360, and $300, respectively. Plants 1, 2, and 3 have the excess capacity to produce 750, 900, and 450 units per day of this product, respectively, regardless of the size or combination of sizes involved. The amount of available in-process storage space also imposes a limitation on the production rates of the new product. Plants 1, 2, and 3 have 13,000, 12,000, and 5,000 square feet, respectively, of in-process storage space available for a day's production of this product. Each unit of the large, medium, and small sizes produced per day requires 20, 15, and 12 square feet, respectively. Sales forecasts indicate that if available, 900, 1200, and 750 units of the large, medium, and small sizes, respectively, would be sold per day. At each plant, some employees will need to be laid off unless most of the plant's excess production capacity can be used to produce the new product. To avoid layoffs if possible, management has decided that the plants should use the same percentage of their excess capacity to produce the new product. Management wishes to know how much of each of the sizes should be produced by each of the plants to maximize profit. 1. Solve the problem using lpsolve, or any other equivalent library in R. 2. Identify the shadow prices, dual solution, and reduced costs 3. Further, identify the sensitivity of the above prices and costs. That is, specify the range of shadow prices and reduced cost within which the optimal solution will not change. 4. Formulate the dual of the above problem and solve it. Does the solution agree with what you observed for the primal problem

```
setwd("D:\\Study\\Assignments\\DMM\\Assignment 3")

library(lpSolveAPI)
```

##Formulation of the Weigelt Corporation

```
# make an lp object with 0 constraints and 9 decision variables
lprec <- make.lp(0, 9)
lprec

## Model name:
##   a linear program with 9 decision variables and 0 constraints
```

```r
# Create the objective function and since we need to maximize profit, change the sense to max.
set.objfn(lprec, c(420, 360, 300, 420, 360, 300, 420, 360, 300))
lp.control(lprec,sense='max')
```

```
## $anti.degen
## [1] "none"
##
## $basis.crash
## [1] "none"
##
## $bb.depthlimit
## [1] -50
##
## $bb.floorfirst
## [1] "automatic"
##
## $bb.rule
## [1] "pseudononint" "greedy"       "dynamic"      "rcostfixing"
##
## $break.at.first
## [1] FALSE
##
## $break.at.value
## [1] 1e+30
##
## $epsilon
##       epsb       epsd      epsel     epsint epsperturb   epspivot
##      1e-10      1e-09      1e-12      1e-07      1e-05      2e-07
##
## $improve
## [1] "dualfeas" "thetagap"
##
## $infinite
## [1] 1e+30
##
## $maxpivot
## [1] 250
##
## $mip.gap
## absolute relative
##    1e-11    1e-11
##
## $negrange
## [1] -1e+06
##
## $obj.in.basis
## [1] TRUE
##
## $pivoting
```

```
## [1] "devex"    "adaptive"
##
## $presolve
## [1] "none"
##
## $scalelimit
## [1] 5
##
## $scaling
## [1] "geometric"   "equilibrate" "integers"
##
## $sense
## [1] "maximize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 0
##
## $verbose
## [1] "neutral"
```

```
# Add the constraints

#Capacity Constraints
add.constraint(lprec, c(1, 1, 1, 0, 0, 0, 0, 0, 0), "<=", 750)
add.constraint(lprec, c(0, 0, 0, 1, 1, 1, 0, 0, 0), "<=", 900)
add.constraint(lprec, c(0, 0, 0, 0, 0, 0,1, 1, 1), "<=", 450)

#Storage Constraints
add.constraint(lprec, c(20, 15, 12, 0, 0, 0, 0, 0, 0), "<=", 13000)
add.constraint(lprec, c(0, 0, 0, 20, 15, 12, 0, 0, 0), "<=", 12000)
add.constraint(lprec, c(0, 0, 0, 0, 0, 0, 20, 15, 12), "<=", 5000)

#Sales forecast constraints
add.constraint(lprec, c(1, 1, 1, 0, 0, 0, 0, 0, 0), "<=", 900)
add.constraint(lprec, c(0, 0, 0, 1, 1, 1, 0, 0, 0), "<=", 1200)
add.constraint(lprec, c(0, 0, 0, 0, 0, 0, 1, 1, 1), "<=", 750)

#Same Capacity % constraints
add.constraint(lprec, c(6, 6, 6, -5, -5, -5, 0, 0, 0), "=", 0)
add.constraint(lprec, c( 3, 3, 3, 0, 0, 0, -5, -5, -5), "=", 0)

#set.bounds(lprec, lower = c(0, 0, 0, 0, 0, 0, 0, 0, 0), columns = c(1,
2,3,4,5,6,7,8,9))

# To identify the variables and constraints, we can set variable names and
name the constraints
```

```r
RowNames <- c("CapCon1", "CapCon2", "CapCon3", "StoCon1", "StoCon2",
"StoCon3", "SalCon1", "SalCon2", "SalCon3", "%C1", "%C2")
ColNames <- c("P1Large", "P1Medium", "P1Small", "P2Large", "P2Medium",
"P2Small", "P3Large", "P3Medium", "P3Small")
dimnames(lprec) <- list(RowNames, ColNames)

lprec

## Model name:
##   a linear program with 9 decision variables and 11 constraints
```

1. Solve the problem using lpsolve, or any other equivalent library in R.

```r
solve(lprec)

## [1] 0

get.objective(lprec) #Maximized profit

## [1] 696000

get.variables(lprec)

## [1] 516.6667 177.7778   0.0000   0.0000 666.6667 166.6667   0.0000
0.0000
## [9] 416.6667

get.constraints(lprec)

##  [1]   694.4444   833.3333   416.6667 13000.0000 12000.0000  5000.0000
##  [7]   694.4444   833.3333   416.6667     0.0000     0.0000
```

2. Identify the shadow prices, dual solution, and reduced costs 3.Further, identify the sensitivity of the above prices and costs. That is, specify the range of shadow prices and reduced cost within which the optimal solution will not change.

```r
get.sensitivity.obj(lprec)

## $objfrom
## [1]  3.60e+02  3.45e+02 -1.00e+30 -1.00e+30  3.45e+02  2.52e+02 -1.00e+30
## [8] -1.00e+30  2.04e+02
##
## $objtill
## [1] 4.60e+02 4.20e+02 3.24e+02 4.60e+02 4.20e+02 3.24e+02 7.80e+02
4.80e+02
## [9] 1.00e+30

get.sensitivity.objex(lprec) #Reduced costs with Range

## $objfrom
## [1]  3.60e+02  3.45e+02 -1.00e+30 -1.00e+30  3.45e+02  2.52e+02 -1.00e+30
## [8] -1.00e+30  2.04e+02
##
## $objtill
```

```
## [1] 4.60e+02 4.20e+02 3.24e+02 4.60e+02 4.20e+02 3.24e+02 7.80e+02
4.80e+02
## [9] 1.00e+30
##
## $objfromvalue
## [1] -1.000000e+30 -1.000000e+30  1.111111e+02  2.500000e+02 -1.000000e+30
## [6] -1.000000e+30  2.500000e+01  6.666667e+01 -1.000000e+30
##
## $objtillvalue
## [1] NA NA NA NA NA NA NA NA NA

#$objfrom and objtill shows the upper and the lower limit variation

options(scipen = 0)
get.sensitivity.rhs(lprec) #Shadow prices with Range

## $duals
##  [1]    0    0    0   12   20   60    0    0    0  -12   84    0    0  -24
-40
## [16]    0    0 -360 -120    0
##
## $dualsfrom
##  [1] -1.000000e+30 -1.000000e+30 -1.000000e+30  1.041667e+04  1.000000e+04
##  [6]  4.800000e+03 -1.000000e+30 -1.000000e+30 -1.000000e+30 -3.333333e+02
## [11] -8.333333e+01 -1.000000e+30 -1.000000e+30 -8.611111e+02 -1.000000e+02
## [16] -1.000000e+30 -1.000000e+30 -5.000000e+01 -1.333333e+02 -1.000000e+30
##
## $dualstill
##  [1] 1.000000e+30 1.000000e+30 1.000000e+30 1.388889e+04 1.250000e+04
##  [6] 5.400000e+03 1.000000e+30 1.000000e+30 1.000000e+30 1.666667e+02
## [11] 1.666667e+02 1.000000e+30 1.000000e+30 1.111111e+02 2.500000e+02
## [16] 1.000000e+30 1.000000e+30 2.500000e+01 6.666667e+01 1.000000e+30

#$dualsfom and $dualstill shows the upper and the lower limits

get.dual.solution(lprec) #dual solutions

##  [1]    1    0    0    0   12   20   60    0    0    0  -12   84    0    0
-24
## [16]  -40    0    0 -360 -120    0
```

4. Formulate the dual of the above problem and solve it. Does the solution agree with what you observed for the primal problem?

#Dual problem:

MIN Z(y) = 750 y1 + 900 y2 + 450 y3 + 13000 y4 + 12000 y5 + 5000 y6 + 900 y7 + 1200 y8 + 750 y9
subject to

y1 + 20y4 + y7 + 6y10 + 3y11 ≥ 420

y1 + 15y4 + y7 + 6y10 + 3y11 ≥ 360

y1 + 12y4 + y7 + 6y10 + 3y11 ≥ 300

y2 + 20y5 + y8 - 5y10 ≥ 420

y2 + 15y5 + y8 - 5y10 ≥ 360

y2 + 12y5 + y8 - 5y10 ≥ 300

y3 + 20y6 + y9 - 5y11 ≥ 420

y3 + 15y6 + y9 - 5y11 ≥ 360

y3 + 12y6 + y9 - 5y11 ≥ 300

y1, y2, y3, y4, y5, y6, y7, y8, y9 ≥0 and y10, y11 unrestricted in sign

#Solving the dual problem

```r
# make an lp object with 0 constraints and 11 decision variables
dual <- make.lp(0, 11)


# Create the objective function and since we need to minimize z, change the
sense to min
set.objfn(dual, c(750,900,450,13000,12000,5000,900,1200,750,0,0))
lp.control(dual, sense='min')

## $anti.degen
## [1] "fixedvars" "stalling"
##
## $basis.crash
## [1] "none"
##
## $bb.depthlimit
## [1] -50
##
## $bb.floorfirst
## [1] "automatic"
##
## $bb.rule
## [1] "pseudononint" "greedy"       "dynamic"      "rcostfixing"
##
```

```
## $break.at.first
## [1] FALSE
##
## $break.at.value
## [1] -1e+30
##
## $epsilon
##       epsb       epsd      epsel     epsint epsperturb   epspivot
##      1e-10      1e-09      1e-12      1e-07      1e-05      2e-07
##
## $improve
## [1] "dualfeas" "thetagap"
##
## $infinite
## [1] 1e+30
##
## $maxpivot
## [1] 250
##
## $mip.gap
## absolute relative
##    1e-11    1e-11
##
## $negrange
## [1] -1e+06
##
## $obj.in.basis
## [1] TRUE
##
## $pivoting
## [1] "devex"    "adaptive"
##
## $presolve
## [1] "none"
##
## $scalelimit
## [1] 5
##
## $scaling
## [1] "geometric"   "equilibrate" "integers"
##
## $sense
## [1] "minimize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 0
##
```

```
## $verbose
## [1] "neutral"

# Add the constraints

add.constraint(dual, c(1, 0, 0, 20, 0, 0, 1, 0, 0, 6, 3), ">=", 420)
add.constraint(dual, c(1, 0, 0, 15, 0, 0, 1, 0, 0, 6, 3), ">=", 360)
add.constraint(dual, c(1, 0, 0, 12, 0, 0, 1, 0, 0, 6, 3), ">=", 300)

add.constraint(dual, c(0, 1, 0, 0, 20, 0, 0, 1, 0, -5, 0), ">=", 420)
add.constraint(dual, c(0, 1, 0, 0, 15, 0, 0, 1, 0, -5, 0), ">=", 360)
add.constraint(dual, c(0, 1, 0, 0, 12, 0, 0, 1, 0, -5, 0), ">=", 300)

add.constraint(dual, c(0, 0, 1, 0, 0, 20, 0, 0, 1, 0, -5), ">=", 420)
add.constraint(dual, c(0, 0, 1, 0, 0, 15, 0, 0, 1, 0, -5), ">=", 360)
add.constraint(dual, c(0, 0, 1, 0, 0, 12, 0, 0, 1, 0, -5), ">=", 300)



dual

## Model name:
##   a linear program with 11 decision variables and 9 constraints

options(scipen = 100)
solve(dual)

## [1] 0

get.objective(dual)

## [1] 700000

get.variables(dual)

##  [1]  0 60  0 12 20 50  0  0  0  0 60

get.constraints(dual)

## [1] 420 360 324 460 360 300 700 450 300
```