QMM Assignment 4 YASH BHANUSHALI

Yash Bhanushali

9/23/2022

assignment question as given

Consider the problem from a previous assignment. 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, 1,200, 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. "Solve the problem using lpsolve, or any other equivalent library in R."

```
library(lpSolveAPI)

## Warning: package 'lpSolveAPI' was built under R version 4.1.3

lprec <- make.lp(0,9)
set.objfn(lprec, c(420,360,300,420,360,300,420,360,300))
lp.control(lprec,sense='max')
```

```
## $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] "maximize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 0
##
## $verbose
## [1] "neutral"
```

```
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)
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)
add.constraint(lprec, c(1,0,0,1,0,0,1,0,0), "<=", 900)
add.constraint(lprec, c(0,1,0,0,1,0,0,1,0), "<=", 1200)
add.constraint(lprec, c(0,0,1,0,0,1,0,0,1), "<=", 750)
add.constraint(lprec, c(900,900,900,-750,-750,-750,0,0,0), "=", 0)
add.constraint(lprec, c(450,450,450,0,0,0,-750,-750,-750), "=", 0)
Namesofrows <- c("Capacity1", "Capacity2", "Capaticy3",
```

```
"Square_footage1", "Square_footage2", "Square_footage3", "Sales1",
"Sales2", "Sales3", "Same_perc_of_Cpacity1", "Same_perc_of_Cpacity2")
Namesofcolumn <- c("Large1", "Medium1", "Small1","Large2", "Medium2",
"Small2","Large3", "Medium3", "Small3")
dimnames(lprec) <- list(Namesofrows, Namesofcolumn)
lprec

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

write.lp(lprec,filename = "Assignment4.lp", type = "lp")
solve(lprec)

## [1] 0

get.objective(lprec)

## [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]  6.944444e+02  8.333333e+02  4.166667e+02  1.300000e+04
1.200000e+04
##  [6]  5.000000e+03  5.166667e+02  8.444444e+02  5.833333e+02
-2.037268e-10
## [11]  0.000000e+00
```

Explanation

As suggested in the question to either use the Lpsolve or any other equivalent library in R I have used the library Lpsolveapi for solving this assignment.

The code has created a total of 9 columns after the lpsolveapi library was used.

In the following line of code the model has been set to as a maximization

We had to add the constraints in our model in order for it to produce an efficient output the next few lines have added the constraints with two non-negavtive constraints at the bottom.

In the following line of code an object named as “Namesofrow” is created under which 11 labels are stored which have followed the pattern same as of constraints

Again an object known as Namesofcolumn is created under which 9 labels are stored there in the same order as the objective function

In the following line we have merged the Namesofrow and Namesofcolumn objects into the lp model as the respective headers of rows and columns

The next line is just the lp model, providing a summary of said model in the output.

Next, an lp file is created known as “Assignment4.lp”

last lines solves the lp model, then retreive the objective, variables, and constraints of the model which are displayed at the bottom of the outputs.