rsingav1_4

2023-09-21

Firstly installing and loading lpSolveAPI library

```
library(lpSolveAPI)
```

#Decision Variables: L1, M1, S1: Number of large, medium, and small units produced at Plant 1. L2, M2, S2: Number of large, medium, and small units produced at Plant 2. L3, M3, S3: Number of large, medium, and small units produced at Plant 3. There are 9 in number:

```
# Create a model with 9 decision variables
lprec_weigelt = make.lp(0, 9)
```

Objective Function: Maximize P = 420(L1 + L2 + L3) + 360(M1 + M2 + M3) + 300*(S1 + S2 + S3)

Define the objective function coefficients (net unit profit). In this case, the net unit profit for each product: Large, Medium and small

```
set.objfn(lprec_weigelt, c(420, 420, 420, 360, 360, 360, 300, 300, 300))
```

Define the constraint matrix. This includes the production capacities of 3 plants, storage space for each plant, Sales forecast, Same percentage of their excess capacity

More details on Constraints:

Capacity Constraints for each plant:

L1 + M1 + S1 <= 750 L2 + M2 + S2 <= 900 L3 + M3 + S3 <= 450

In-process Storage Space Constraints for each plant:

20L1 + 15M1 + 12S1 <= 13000 20L2 + 15M2 + 12S2 <= 12000 20L3 + 15M3 + 12*S3 <= 5000

Sales Forecast Constraints for each size:

L1 + L2 + L3 <= 900 M1 + M2 + M3 <= 1200 S1 + S2 + S3 <= 750

Non-negativity Constraints: L1, M1, S1, L2, M2, S2, L3, M3, S3 >= 0

One last constraint to avoid layoffs is to introduce a condition that the ratio of utilized capacity to total capacity should be the same for all plants. This can be represented as:

(L1 + M1 + S1) / 750 = (L2 + M2 + S2) / 900 = (L3 + M3 + S3) / 450

This constraint is a bit tricky because it's a ratio, which is nonlinear and cannot be solved by lpSolve. However, we can transform it into a linear form by cross multiplication:

900 * (L1 + M1 + S1) - 750 * (L2 + M2 + S2) =0 450 * (L1 + M1 + S1) - 750 * (L3 + M3 + S3) =0 450 * (L2 + M2 + S2) - 900 * (L3 + M3 + S3) =0

Technically, we can use just two of these three equations as the third equation is just a combination of the first two and the solution would be the same. But I have used all the 3 here just for the sake of completeness.

```
# Adding constraints and defining the constraint directions and right hand side of the constraints as p
add.constraint(lprec_weigelt, c(1, 0, 0, 1, 0, 0, 1, 0, 0), "<=", 750) # Plant 1 capacity
add.constraint(lprec_weigelt, c(0, 1, 0, 0, 1, 0, 0, 1, 0), "<=", 900) # Plant 2 capacity
add.constraint(lprec_weigelt, c(0, 0, 1, 0, 0, 1, 0, 0, 1), "<=", 450) # Plant 3 capacity
add.constraint(lprec_weigelt, c(20, 0, 0, 15, 0, 0, 12, 0, 0), "<=", 13000) # Plant 1 storage
add.constraint(lprec_weigelt, c(0, 20, 0, 0, 15, 0, 0, 12, 0), "<=", 12000) # Plant 2 storage
add.constraint(lprec_weigelt, c(0, 0, 20, 0, 0, 15, 0, 0, 12), "<=", 5000) # Plant 3 storage
add.constraint(lprec_weigelt, c(1, 1, 1, 0, 0, 0, 0, 0, 0), "<=", 900)# Sales forecast large
add.constraint(lprec_weigelt, c(0, 0, 0, 1, 1, 1, 0, 0, 0), "<=", 1200)# Sales forecast medium
add.constraint(lprec_weigelt, c(0, 0, 0, 0, 0, 0, 1, 1, 1), "<=", 750)# Sales forecast small
add.constraint(lprec_weigelt, c(900, -750, 0, 900, -750, 0, 900, -750, 0), "=", 0) # Same percentage of
add.constraint(lprec_weigelt, c(-450, 0, 750, -450, 0, 750, -450, 0, 750), "=", 0)# Same percentage of
add.constraint(lprec_weigelt, c(0, 450, -900, 0, 450, -900, 0, 450, -900), "=", 0)# Same percentage of
add.constraint(lprec_weigelt, c(1, 0, 0, 0, 0, 0, 0, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 1, 0, 0, 0, 0, 0, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 1, 0, 0, 0, 0, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 0, 1, 0, 0, 0, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 0, 0, 1, 0, 0, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 0, 0, 0, 1, 0, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 0, 0, 0, 0, 1, 0, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 0, 0, 0, 0, 0, 1, 0), ">=", 0) # NonNegative constraint
add.constraint(lprec_weigelt, c(0, 0, 0, 0, 0, 0, 0, 0, 1), ">=", 0) # NonNegative constraint
```

```r
# Set all variables to be integer as we do not want float values for the unit sizes
for (i in 1:9) {
  set.type(lprec_weigelt, i, "integer")
}
```

Solve the linear programming problem by maximizing the profit while meeting the constraints

```r
# Set the objective function to maximize
lp.control(lprec_weigelt, 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"
```

```
# Solve the linear programming model
solve(lprec_weigelt)
```

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

This will give you the optimal production amounts for each size at each plant to maximize profit, subject to the constraints.

L1 = 530 L2 = 0 L3 = 1 M1 = 160 M2 = 688 M3 = 8 S1 = 0 S2 = 140 S3 = 405

```
# Get the solution
all_unit_sizes_for_all_plants = get.variables(lprec_weigelt)
print(all_unit_sizes_for_all_plants)
```

```
## [1] 530   0   1 160 688   8   0 140 405
```

Also the optimal profit or maximum profit is: 694680$

```
# Get the optimal profit
maximum_profit = get.objective(lprec_weigelt)
print(maximum_profit)
```

```
## [1] 694680
```

Writing the above lp model to a file:

```
write.lp(lprec_weigelt, filename = "weigelt_lp.lp")
```