INTEGER PROGRAMMING

SANIA FATIMA

2023-11-19

Creating a table that provides an estimate of the number of workers needed each day of the week

```
#install.packages("kableExtra")

library(kableExtra)
data_1 <- data.frame(Day = c("Sunday","Monday","Tuesday","Wednesday","Thursday","Frid
ay","Saturday"),
                     Workers_Required = c(18,27,22,26,25,21,19))

data_1 %>%
 kable(align = "c", format = "html") %>%
  kable_styling(bootstrap_options = "bordered", full_width = TRUE) %>%
  row_spec(0, bold = TRUE)
```

Day	**Workers_Required**
Sunday	18
Monday	27
Tuesday	22
Wednesday	26
Thursday	25
Friday	21
Saturday	19

Package handlers at AP are guaranteed a five-day work week with two consecutive days off. The base wage for the handlers is $750 per week. Workers working on Saturday or Sunday receive an additional $25 per day. The possible shifts and salaries for package handlers are:

#Creating a table to show the possible shifts and salaries for package handlers

```
data_2 <- data.frame(Shift = c(1,2,3,4,5,6,7) ,
                     Days_Off = c("Sunday and Monday",
                                  "Monday and Tuesday",
                                  "Tuesday and Wednesday",
                                  "Wednesday and Thursday",
                                  "Thursday and Friday",
                                  "Friday and Saturday",
                                  "Saturday and Sunday"),
                     Wage = c(775,800,800,800,800,775,750))
data_2 %>%
 kable(align = "c", format = "html") %>%
  kable_styling(bootstrap_options = "bordered", full_width = TRUE) %>%
  row_spec(0, bold = TRUE)
```

Shift	Days_Off	Wage
1	Sunday and Monday	775
2	Monday and Tuesday	800
3	Tuesday and Wednesday	800
4	Wednesday and Thursday	800
5	Thursday and Friday	800
6	Friday and Saturday	775
7	Saturday and Sunday	750

Question: The manager wants to keep the total wage expenses as low as possible while ensuring that there are sufficient number of workers available each day. Formulate and solve the problem. What was the total cost? How many workers are available each day?

1.Formulate the problem.

Before getting into the formulation we first need to define the decision variables that are discussed as follows,

w1:Number of workers assigned to Shift1

w2:Number of workers assigned to Shift2

w3:Number of workers assigned to Shift3

w4:Number of workers assigned to Shift4

w5:Number of workers assigned to Shift5

w6:Number of workers assigned to Shift6

w7:Number of workers assigned to Shift7

Objective Function:

It is given that the manager wants to minimize the total wages as low as possible, by adjusting the decision variables we can control the labor cost. So, our objective function can be formulated as follows,

MinZ=775w1+800w2+800w3+800w4+800w5+775w6+750w7

Constraints:

Formulating the constraints for all the days in the week basing on the seven decision variables.

Sunday:$w2+w3+w4+w5+w6\geq18$

Monday:$w3+w4+w5+w6+w7\geq27$

Tuesday:$w1+w4+w5+w6+w7\geq22$

Wednesday:$w1+w2+w5+w6+w7\geq26$

Thursday:$w1+w2+w3+w6+w7\geq25$

Friday:$w1+w2+w3+w4+w7\geq21$

Saturday:$w1+w2+w3+w4+w5\geq19$

Non-negativity of the decision variables:

$w1\geq0,w2\geq0,w3\geq0,w4\geq0,w5\geq0,w6\geq0,w7\geq0$

Solve the problem in R markdown.

#Loading the required libraries.

```
library(lpSolveAPI)
```

#Creating an Linear Programming problem instance with 7 constraints and 7 decision variables.

```
lp <-make.lp(7,7)
```

#Setting the objective function coefficients for the 7 decision variables.

```
set.objfn(lp, c(775,800,800,800,800,775,750))
```

#Setting the Objective sense to ‘min’.

```
lp.control(lp, 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"
```

Since, this problem needs to be solved by integer programming, we need to use the type as integer, which says that all the variables should be of integers and not fractions #Adding the constraints to the LP problem.

```
set.row(lp,1,c(0,1,1,1,1,1,0))
set.row(lp,2,c(0,0,1,1,1,1,1))
set.row(lp,3,c(1,0,0,1,1,1,1))
set.row(lp,4,c(1,1,0,0,1,1,1))
set.row(lp,5,c(1,1,1,0,0,1,1))
set.row(lp,6,c(1,1,1,1,0,0,1))
set.row(lp,7,c(1,1,1,1,1,0,0))
```

#Creating a vector rhs containing the right-hand side values for the two constraints

```
rhs<-c(18,27,22,26,25,21,19)
```

#Setting the right-hand side values for the constraints in the LP problem.

```
set.rhs(lp,rhs)
```

#Setting the constraint types for the two constraints.

```
set.constr.type(lp,c(">=",">=",">=",">=",">=",">=",">="))
```

#Setting the lower bounds for the 7 decision variables. The lower bounds are set to zero in this case (non-negativity of the decision variables).

```
set.bounds(lp,lower = rep(0,7))
```

#Assigning names to the constraints.

```
lp.rownames<-c("Sunday","Monday","Tuesday", "Wednesday", "Thursday", "Friday", "Satur
day")
dimnames(lp) = list(lp.rownames,(1:7))
```

#Assigning names to the seven decision variables

```
lp.colnames <- c("w1","w2","w3","w4","w5","w6","w7")
set.type(lp, columns = 1:7, type = "integer")
lp
```

```
## Model name:
##                1    2    3    4    5    6    7
## Minimize     775  800  800  800  800  775  750
## Sunday         0    1    1    1    1    1    0  >=  18
## Monday         0    0    1    1    1    1    1  >=  27
## Tuesday        1    0    0    1    1    1    1  >=  22
## Wednesday      1    1    0    0    1    1    1  >=  26
## Thursday       1    1    1    0    0    1    1  >=  25
## Friday         1    1    1    1    0    0    1  >=  21
## Saturday       1    1    1    1    1    0    0  >=  19
## Kind         Std  Std  Std  Std  Std  Std  Std
## Type         Int  Int  Int  Int  Int  Int  Int
## Upper        Inf  Inf  Inf  Inf  Inf  Inf  Inf
## Lower          0    0    0    0    0    0    0
```

#Solving the LP problem using the specified constraints and objective function.

```
solve(lp)
```

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

#Retrieving the optimal objective value of the LP problem.

```
get.objective(lp)
```

```
## [1] 25675
```

#Retrieving the values of the decision variables at the optimal solution.

- This show how many workers are corresponding to each shift on respective day

```
round(get.variables(lp))
```

```
## [1]  2  4  5  0  8  1 13
```

#This is the total number of employees, along with their pay and shift pattern, which are described as follows:

```
#By using the variables from the lp model
Table = matrix(c(0,4,5,0,8,1,0,0,0,5,0,8,1,13,2,0,0,0,8,1,13,2,4,0,0,8,1,13,2,4,5,0,
0,1,13,2,3,4,0,0,0,13,2,4,5,0,8,0,0),ncol=7,byrow=TRUE)

colnames(Table) = c("Shift1", "Shift2", "Shift3", "Shift4", "Shift5", "Shift6", "Shif
t7")

row.names(Table) = c('Sunday', 'Monday', 'Tuesday','Wednesday','Thursday','Friday','S
aturday')

Table
```

```
##           Shift1 Shift2 Shift3 Shift4 Shift5 Shift6 Shift7
## Sunday         0      4      5      0      8      1      0
## Monday         0      0      5      0      8      1     13
## Tuesday        2      0      0      0      8      1     13
## Wednesday      2      4      0      0      8      1     13
## Thursday       2      4      5      0      0      1     13
## Friday         2      3      4      0      0      0     13
## Saturday       2      4      5      0      8      0      0
```

#The below table shows employees available each day based on the shift arrangement that reduces the overall wage cost.

```
rowSums(Table)
```

```
##    Sunday    Monday   Tuesday Wednesday  Thursday    Friday  Saturday 
##        18        27        24        28        25        22        19
```