Final Exam

Mark Bruner

12/2/20

Defining the Problem, Providing the Context, and Discussing Factors

1. **Problem Definition**

- **Problem to Solve:** Forming equitable groups for a class project.
- **Primary Objective: Maximize** the chance that a group performs well on a class project.

2. **Class Project Context**

- **Class:** Business Analytics Course
- **Class Project:** The groups are asked to create a model to predict whether a customer will churn or not based on a dataset provided by a cellular company.

3. **Problem Type:** Assignment Problem

Factors

1. **Factors of a Group**

- **1. Visualization Knowledge** (variable column header: visual_know)
  - The group member will create the visualizations for the report and perform the EDA.
- **2. Programming Skill** (variable column header: prog_skill)
  - This group member will create the model for predicting churn.
- **3. Presentation Ability** (variable column header: pres_ability)
  - This group member will create the presentation as well as give the presentation.

2. **Data Collection and Definitions**

- The professor created a survey to assess a student's comfort level with public speaking. The levels they could choose from are as follows:
  - 1: Significant fear and not confident
  - 2: Some fear and has adequate confidence
  - 3: Comfortable and confident
  - 4: Enthusiastic, excited, and higly confident
- The professor will assign a value to each student based upon submitted work on how capable a student is in programming and visualization knowledge since there are an adequate amount of previous assignment that students have completed to base this rating. The values are defined as follow:

  - 1: Poor
  - 2: Adequate
  - 3: Good
  - 4: Excellent

3. **Decision Variables**

- The decision variables represents the possibility of each of the 12 students occupying one of the 3 roles in the 4 groups.

$$x_{11},\ x_{12},\ x_{13}, ...,\ x_{ij}, \qquad where\ i = 1, 2, 3, ...12\ and\ j = 1, 2, 3, ..., 12$$

4. **Objective Function**

- The objective function represents the ability of each of the 12 students for the 3 roles in the 4 groups. It needs to be maximized since we want the best 3 students combination of their abilities for each group.

$$maximize : \sum_{i=1}^{12} \sum_{j=1}^{12} c_{ij} x_{ij}$$

5. **Constraints**

- Only one student can occupy one role in one group.
- Only one role can be assigned to one student in each group.
- I want to have at least one 4 in each group and at the very least two 3's. I made the sum of each group's total score ">=" 10 to achieve that goal.
- All decision variables will be non-negative.

such that

$$\sum_{j=1}^{12} x_{ij} = 1, \qquad for\ all\ i = 1, 2, ..., 12$$

$$\sum_{i=1}^{12} x_{ij} = 1, \qquad for\ all\ j = 1, 2, ..., 12$$

$$Groups\ Constraints :$$

$$Group\ 1 : \sum_{i=1}^{12} c_{ij} x_{ij} = 10 \qquad where\ j = 1, 2,\ and\ 3$$

$$Group\ 2 : \sum_{i=1}^{12} c_{ij} x_{ij} = 10 \qquad where\ j = 1, 2,\ and\ 3$$

$$Group\ 3 : \sum_{i=1}^{12} c_{ij} x_{ij} = 10 \qquad where\ j = 1, 2,\ and\ 3$$

$$Group\ 4 : \sum_{i=1}^{12} c_{ij} x_{ij} = 10 \qquad where\ j = 1, 2,\ and\ 3$$

$$and\ x_{ij} \geq 0.$$

```r
rm(list=ls())
library(lpSolveAPI)
```

```r
# Assuring that the data set does not change every time I run the entire code.
set.seed(15)

# Randomly generating the data set using the guidelines from the 'Data Collection and
# Definition' section.
pres_ability <- sample.int(4, 12, replace = TRUE)  # Allowing for duplicate numbers
# in the dataset since multiple students could have the same variable skill, knowledge, or
# ability.
prog_skill <- sample.int(4, 12, replace = TRUE)
vis_know <- sample.int(4, 12, replace = TRUE)

# Creating objective function vector for assignment formulation.
group_obj <- c(pres_ability, prog_skill, vis_know, pres_ability, prog_skill, vis_know, pres_ability,
    prog_skill, vis_know, pres_ability, prog_skill, vis_know)

# creating a df for the group objective function coefficients to be able to use later to
# check the results.
group1 <- as.data.frame(cbind(pres_ability, prog_skill, vis_know))
group2 <- group1
group3 <- group1
group4 <- group1

# Naming the columns and rows.
group <- as.data.frame(as.matrix(cbind(group1, group2, group3, group4), 12, 12))
colnames(group) <- c("VisGrp1", "ProgGrp1", "PresGrp1", "VisGrp2", "ProgGrp2", "PresGrp2",
    "VisGrp3", "ProgGrp3", "PresGrp3", "VisGrp4", "ProgGrp4", "PresGrp4")

rownames(group) <- c("Student1", "Student2", "Student3", "Student4", "Student5", "Student6",
    "Student7", "Student8", "Student9", "Student10", "Student11", "Student12")
group
```

Objective Function

```r
# Creating the lprec for this problem.
lprec <- make.lp(0,144)
lp.control(lprec, sense = "max")
set.type(lprec, 1:144, type = c("integer"))
set.objfn(lprec, group_obj)
```

Constraints

```r
# Only one role per student.
add.constraint(lprec, c(rep(1, 12)), indices = c(1:12), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(13:24), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(25:36), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(37:48), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(49:60), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(61:72), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(73:84), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(85:96), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(97:108), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(109:120), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(121:132), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices = c(133:144), "=", 1)

# Only one student per role.
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(1, 13, 25, 37, 49, 61, 73, 85, 97, 109, 121, 133), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(2, 14, 26, 38, 50, 62, 74, 86, 98, 110, 122, 134), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(3, 15, 27, 39, 51, 63, 75, 87, 99, 111, 123, 135), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(4, 16, 28, 40, 52, 64, 76, 88, 100, 112, 124, 136), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(5, 17, 29, 41, 53, 65, 77, 89, 101, 113, 125, 137), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(6, 18, 30, 42, 54, 66, 78, 90, 102, 114, 126, 138), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(7, 19, 31, 43, 55, 67, 79, 91, 103, 115, 127, 139), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(8, 20, 32, 44, 56, 68, 80, 92, 104, 116, 128, 140), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(9, 21, 33, 45, 57, 69, 81, 93, 105, 117, 129, 141), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(10, 22, 34, 46, 58, 70, 82, 94, 106, 118, 130, 142), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(11, 23, 35, 47, 59, 71, 83, 95, 107, 119, 131, 143), "=", 1)
add.constraint(lprec, c(rep(1, 12)), indices =
                 c(12, 24, 36, 48, 60, 72, 84, 96, 108, 120, 132, 144), "=", 1)

# At least one 4 rating and two 3 ratings per group for each role.
add.constraint(lprec, c(pres_ability, prog_skill, vis_know),
               indices =
                 c(1, 13, 25, 37, 49, 61, 73, 85, 97, 109, 121, 133,
                   2, 14, 26, 38, 50, 62, 74, 86, 98, 110, 122, 134,
                   3, 15, 27, 39, 51, 63, 75, 87, 99, 111, 123, 135), ">=", 10)
add.constraint(lprec, c(pres_ability, prog_skill, vis_know),
               indices = c(4, 16, 28, 40, 52, 64, 76, 88, 100, 112, 124, 136,
                           5, 17, 29, 41, 53, 65, 77, 89, 101, 113, 125, 137,
                           6, 18, 30, 42, 54, 66, 78, 90, 102, 114, 126, 138), ">=", 10)
add.constraint(lprec, c(pres_ability, prog_skill, vis_know),
               indices =
                 c(7, 19, 31, 43, 55, 67, 79, 91, 103, 115, 127, 139,
                   8, 20, 32, 44, 56, 68, 80, 92, 104, 116, 128, 140,
                   9, 21, 33, 45, 57, 69, 81, 93, 105, 117, 129, 141), ">=", 10)
add.constraint(lprec, c(pres_ability, prog_skill, vis_know),
               indices = c(10, 22, 34, 46, 58, 70, 82, 94, 106, 118, 130, 142,
                           11, 23, 35, 47, 59, 71, 83, 95, 107, 119, 131, 143,
                              12, 24, 36, 48, 60, 72, 84, 96, 108, 120, 132, 144), ">=", 10)
```

```
write.lp(lprec, filename = "final.lp", type = "lp")
lprec
```

```
## Model name:
##   a linear program with 144 decision variables and 28 constraints
```

```
# Solve LP model
solve(lprec)
```

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

```
# The maximum value for the objective function.
get.objective(lprec)

# Decision Variables
get.variables(lprec) -> assign
solution <- matrix(assign, nrow = 12, ncol = 12, byrow = TRUE)

# Multiplying the original group ratings matrix by the solution group assignments.
group*solution -> group_assign
group_assign
```

```
# Group member combinations.
group1 <- cbind(group_assign[c(1, 7, 11), 1:3], group[c(1, 7, 11), 1:3])
group1
```

```
##           VisGrp1 ProgGrp1 PresGrp1 VisGrp1 ProgGrp1 PresGrp1
## Student1        0        4        0       1        4        1
## Student7        0        0        2       1        1        2
## Student11       4        0        0       4        2        4
```

```
group2 <- cbind(group_assign[c(2, 8, 6), 4:6], group[c(2, 8, 6), 4:6])
group2
```

```
##          VisGrp2 ProgGrp2 PresGrp2 VisGrp2 ProgGrp2 PresGrp2
## Student2       3        0        0       3        2        1
## Student8       0        3        0       1        3        2
## Student6       0        0        4       2        1        4
```

```
group3 <- cbind(group_assign[c(9,5,3), 7:9], group[c(9,5,3), 7:9])
group3
```

```
##          VisGrp3 ProgGrp3 PresGrp3 VisGrp3 ProgGrp3 PresGrp3
## Student9       0        0        0       3        2        3
## Student5       0        4        0       1        4        2
## Student3       0        0        3       2        1        3
```

```r
group4 <- cbind(group_assign[c(12,4,10), 10:12], group[c(12,4,10), 10:12])
group4
```

```
##           VisGrp4 ProgGrp4 PresGrp4 VisGrp4 ProgGrp4 PresGrp4
## Student12       0        0        0       3        2        3
## Student4        0        4        0       2        4        1
## Student10       0        0        3       1        1        3
```

Group 1: Student 11: Visual Role with skill of 4 Student 1: Programming skill of 4 Student 7: Presentation skill of 2

Group 2: Student 2: Visual Role with skill of 3 Student 8: Programming skill of 3 Student 6: Presentation skill of 4

Group 3: Student 9: Visual Role with skill of 3 Student 5: Programming skill of 3 Student 3: Presentation skill of 4

Group 4: Student 12: Visual Role with skill of 3 Student 4: Programming skill of 4 Student 10: Presentation skill of 3

The above groups represents and equitable split of the class for the project. Group 1 has two members with 4 but to balance that they have a group member with a 2. The rest of the groups have one 4 rating and two 3's which I believe makes the groupings equitable and achieve the result that I was hoping to accomplish.