QUEEN'S UNIVERSITY BELFAST

QUEEN'S BUSINESS SCHOOL

MGT7180 - Data Driven Decision Making

Assignment -1

Student Name : Suriya Subbiah Perumal

Student ID : 40423910

Table of Contents

1. ApTx-nova - Customized Automotive Tech 01
1.1 Assigning The Decision Variables 01
1.2 Objective Function 01
1.3 Constraints 02
1.3.1 Plant Capacity Constraint 02
1.3.2 Product Production Constraint 02
1.3.3 Product-Plant Compatibility 03
1.4 Final Model 03
1.5 Solving the Model 04
1.5.1 Implementation of Mathematical Model in R 04
1.5.2 R Output Solution 05
1.6 Managerial Statement 05

2. Teranikx - AI Chip 07
2.1 Given Data 07
2.2 Assigning The Decision Variables 08
2.3 Objective Function 08
2.4 Constraints 08
2.4.1 Production Capacity Constraint 08
2.4.2 Demand Constraint 08
2.5 Final Model 09
2.6 Solving the Model 09
2.6.1 Implementation of Mathematical Model in R 10
2.6.2 R Output Solution 10
2.7 Managerial Statement 11

3. Make-to-Stock Chemotherapy Drugs 12
3.1 Assigning The Decision Variables 12
3.2 Objective Function 12
3.3 Constraints 12
3.3.1 Maximum Demand Constraints 12
3.3.2 Minimum Delivery Constraints 13
3.3.3 Inventory Constraints 13
3.3.4 Maximum D-Metric Constraints 13
3.3.5 Minimum P-Metric Constraints 14
3.4 Final Model 14
3.5 Solving the Model 16
3.5 .1 Implementation of Mathematical Model in R 16
3.5.2 R Output Solution 17
3.6 Managerial Statement 18

1) ApTx-nova - Customized Automotive Tech

ApTx-nova is tasked with deciding which of their five plants should produce each of their four products to maximize total production, considering capacity constraints and product-plant compatibility.

1.1 Assigning The Decision Variables :

Let x_{ij} be a binary decision variable where i corresponds to product (i= 1,2,3,4,5) and j corresponds to plant (j=1,2,3,4). x_{ij} = 1 if product i is manufactured in plant j, and 0 otherwise. ($x_{ij} \in \mathbb{Z}^{+}$ for $i = 1,\ 2,3,4,5$ & $j = 1,2,3,4$)

1.2 Objective Function :

The objective is to maximize the total production. Therefore, the objective function can be written as:

$$Maximize\ Z = \sum_{i=1}^{5}\sum_{j=1}^{4}(c_{ij} * x_{ij})$$

$$x_{ij} \geq 0 \text{ for } i = 1,2,3,4,5\ \&\ j = 1,2,3,4$$

where,

- Z is the total number of batches manufactured during the production period
- c is the capacity of plant i for product j

1.3 Constraints :

1.3.1 Plant Capacity Constraint :

Each plant can be scheduled for at most one product.

$$\sum_{j=1}^{4} x_{ij} \leq 1, \forall i \in \{1, 2, 3, 4,5\}$$

1.3.2 Product Production Constraint :

Each product must be produced in exactly in one plant

$$\sum_{i=1}^{5} x_{ij} = 1, \quad \forall j \in \{1, 2, 3, 4\}$$

1.3.3 Product-Plant Compatibility :

Certain products are incompatible with specific plants for production. For instance, products 2 cannot be produced by plants 1, 3, and 4, and similarly, product 3 cannot be manufactured by plant 4.

- For product 2 which cannot be manufactured in plants 1, 3, and 4:

$$x_{1,2} = x_{3,2} = x_{4,2} = 0$$

- For product 3 which cannot be manufactured in plants 4:

$$x_{4,3} = 0$$

Product	1	2	3	4
Plant-1	1200	🟥	600	1000
Plant-2	1400	1200	800	1000
Plant-3	600	🟥	200	600
Plant-4	800	🟥	🟥	1200
Plant-5	800	1400	1000	1600

The red-highlighted cells indicate the plant-product pairings that are incapable of production, signifying that those specific plants cannot manufacture the corresponding products.

1.4 Final Model :

$$Maximize\ Z = \sum_{i=1}^{5}\sum_{j=1}^{4}(c_{ij} * x_{ij})$$

$$x_{ij} \geq 0 \text{ for } i = 1,2,3,4,5 \ \& j = 1,2,3,4$$

Subject to the following constraints :

Plant Capacity :

$$\sum_{i=1}^{5} x_{ij} \leq 1, \quad \forall j \in \{1, 2, 3, 4\}$$

$$x_{ij} \geq 0 \text{ for } i = 1,2,3,4,5 \ \& j = 1,2,3,4$$

Product Production :

$$\sum_{j=1}^{4} x_{ij} = 1, \forall i \ \in \{1, 2, 3, 4,5\}$$

Product-Plant Compatibility :

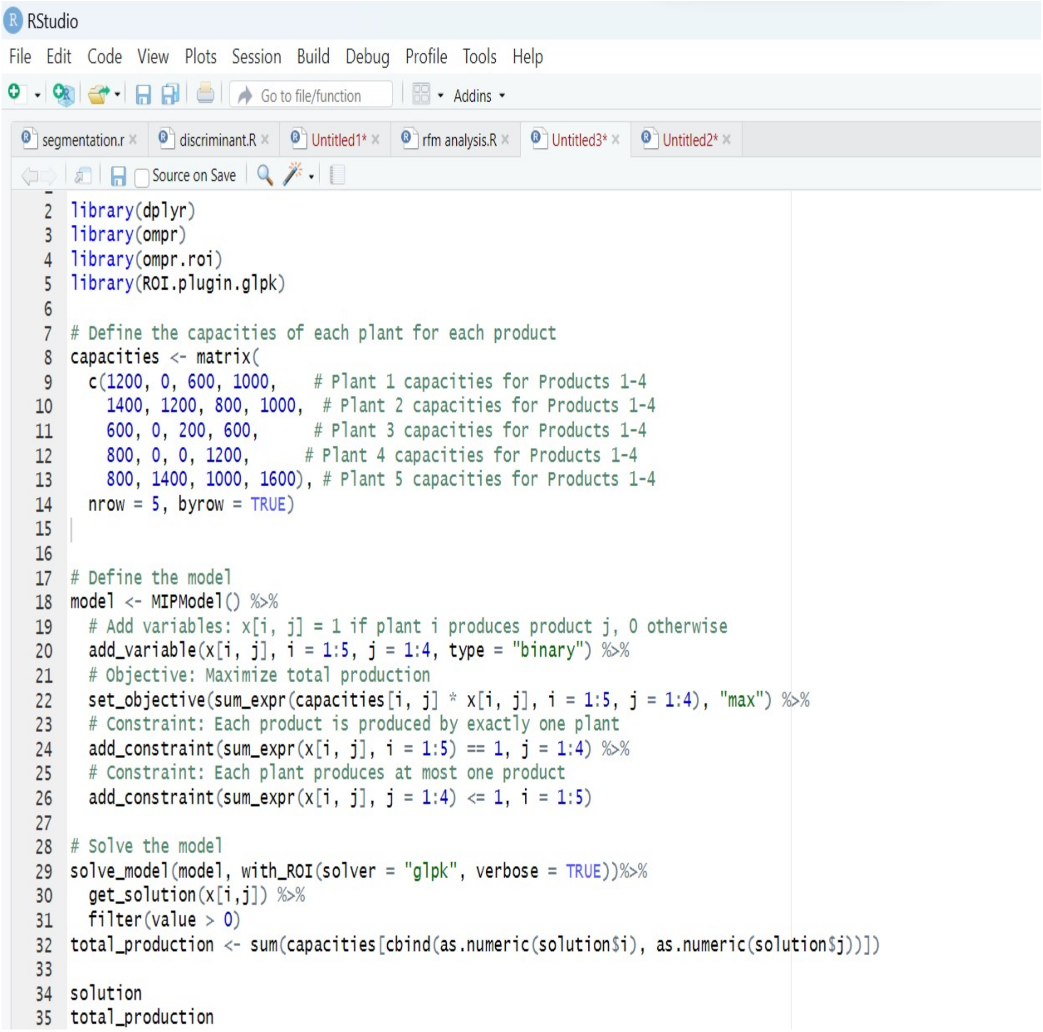
$$x_{2,1} = x_{2,3} = x_{2,3} = 0$$

$$x_{3,4} = 0$$

1.5 Solving the Model :

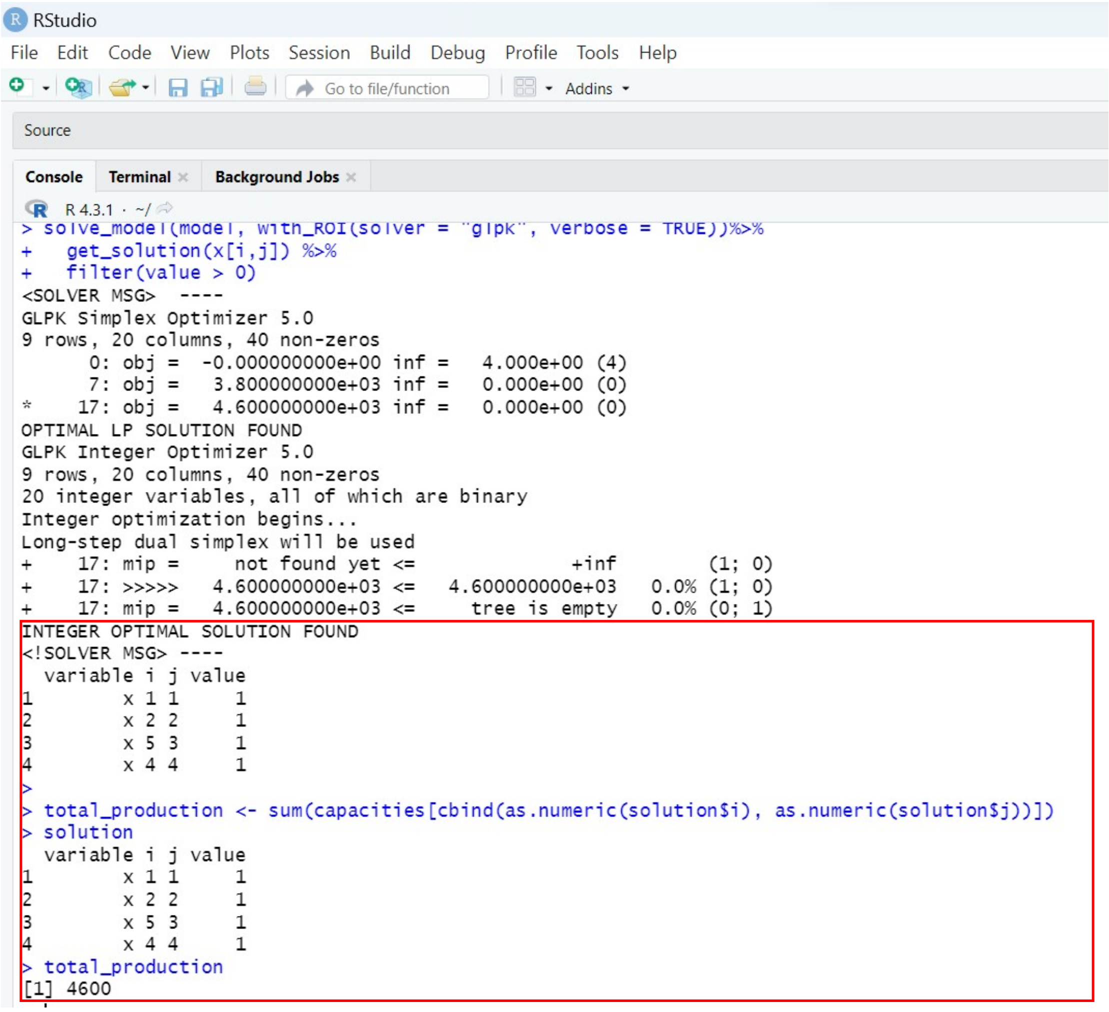
The mathematical model was crafted into an R script and executed using the `ompr` package, supplemented by additional packages like `ROI`, the `ROI plugin`, and `GLPK`, among others. This execution aimed to solve the mathematical model efficiently, aiming to ascertain the optimal solution that maximizes the number of batches produced while minimizing the company's expenditure.

1.5.1 Implementation of Mathematical Model in R :



```
library(dplyr)
library(ompr)
library(ompr.roi)
library(ROI.plugin.glpk)

# Define the capacities of each plant for each product
capacities <- matrix(
  c(1200, 0, 600, 1000,     # Plant 1 capacities for Products 1-4
    1400, 1200, 800, 1000,  # Plant 2 capacities for Products 1-4
    600, 0, 200, 600,       # Plant 3 capacities for Products 1-4
    800, 0, 0, 1200,       # Plant 4 capacities for Products 1-4
    800, 1400, 1000, 1600), # Plant 5 capacities for Products 1-4
  nrow = 5, byrow = TRUE)


# Define the model
model <- MIPModel() %>%
  # Add variables: x[i, j] = 1 if plant i produces product j, 0 otherwise
  add_variable(x[i, j], i = 1:5, j = 1:4, type = "binary") %>%
  # Objective: Maximize total production
  set_objective(sum_expr(capacities[i, j] * x[i, j], i = 1:5, j = 1:4), "max") %>%
  # Constraint: Each product is produced by exactly one plant
  add_constraint(sum_expr(x[i, j], i = 1:5) == 1, j = 1:4) %>%
  # Constraint: Each plant produces at most one product
  add_constraint(sum_expr(x[i, j], j = 1:4) <= 1, i = 1:5)

# Solve the model
solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))%>%
  get_solution(x[i,j]) %>%
  filter(value > 0)
total_production <- sum(capacities[cbind(as.numeric(solution$i), as.numeric(solution$j))])

solution
total_production
```

1.5.2 R Output Solution :



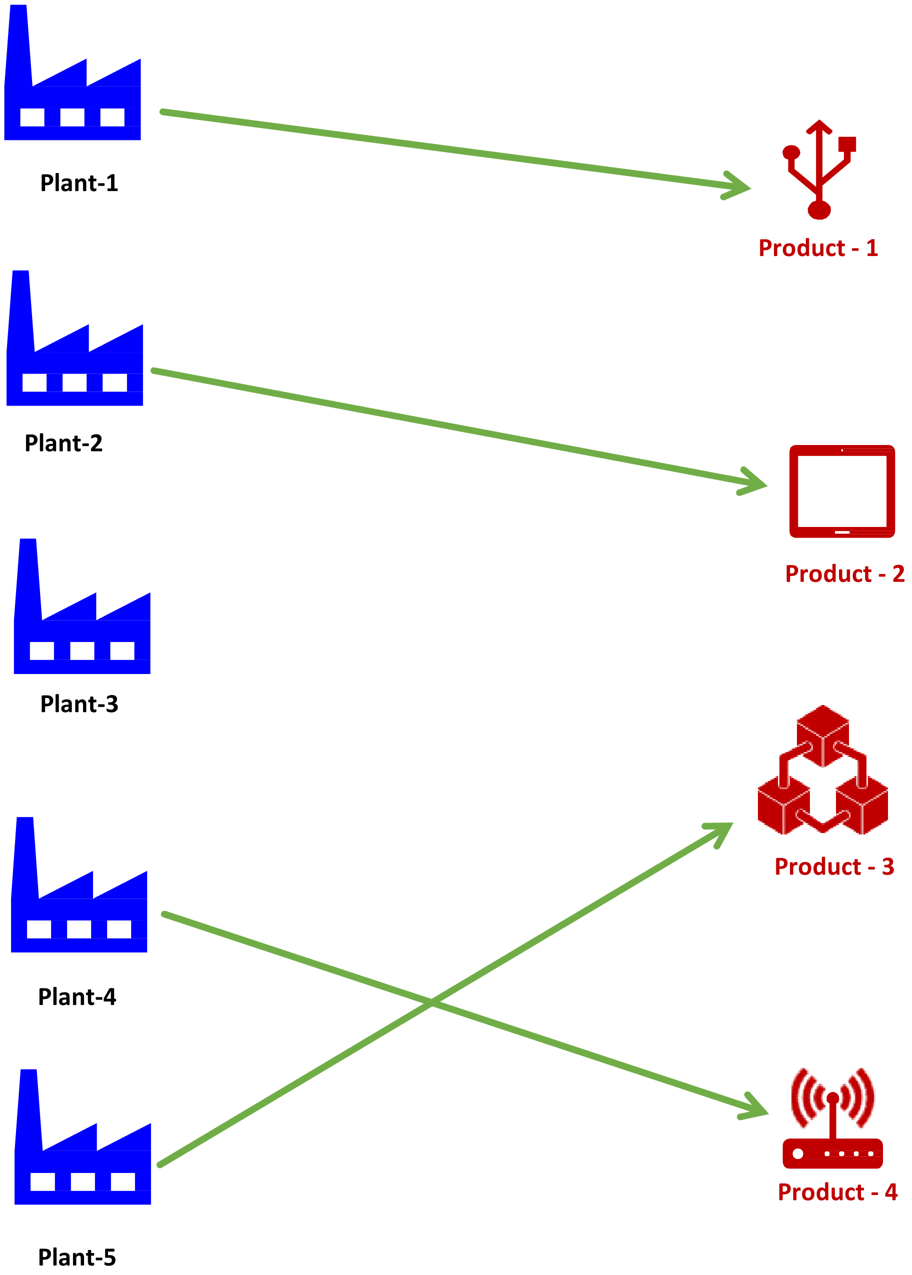
```
> solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))%>%
+   get_solution(x[i,j]) %>%
+   filter(value > 0)
<SOLVER MSG>  ----
GLPK Simplex Optimizer 5.0
9 rows, 20 columns, 40 non-zeros
      0: obj =  -0.000000000e+00 inf =   4.000e+00 (4)
      7: obj =   3.800000000e+03 inf =   0.000e+00 (0)
*    17: obj =   4.600000000e+03 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
GLPK Integer Optimizer 5.0
9 rows, 20 columns, 40 non-zeros
20 integer variables, all of which are binary
Integer optimization begins...
Long-step dual simplex will be used
+    17: mip =     not found yet <=              +inf        (1; 0)
+    17: >>>>>   4.600000000e+03 <=   4.600000000e+03   0.0% (1; 0)
+    17: mip =   4.600000000e+03 <=     tree is empty   0.0% (0; 1)
INTEGER OPTIMAL SOLUTION FOUND
<!SOLVER MSG> ----
  variable i j value
1        x 1 1     1
2        x 2 2     1
3        x 5 3     1
4        x 4 4     1
>
> total_production <- sum(capacities[cbind(as.numeric(solution$i), as.numeric(solution$j))])
> solution
  variable i j value
1        x 1 1     1
2        x 2 2     1
3        x 5 3     1
4        x 4 4     1
> total_production
[1] 4600
```

1.6 Managerial Statement :

In order to maximize the production ApTx-nova should following allocation of plants for the production of products

- **Plant - 1** should be allocated for the production of **product - 1**
- **Plant - 2** should be allocated for the production of **product - 2**
- **Plant - 4** should be allocated for the production of **product - 4**
- **Plant - 5** should be allocated for the production of **product - 3**

Product	1	2	3	4
Plant-1	1200		600	1000
Plant-2	1400	1200	800	1000
Plant-3	600		200	600
Plant-4	800			1200
Plant-5	800	1400	1000	1600



A total production target of 4,600 batches must be achieved within a given production period.

2) Teranikx - AI Chip

Teranikx , a company known for its economical production of ASIC chips, this analysis aims to determine the optimal production levels for two of their manufacturing facilities. Considering the operational constraints and demand, alongside the variable pricing and delivery charges from four distinct customers, this solution seeks to strategize production to not only meet customer demands but also to ensure maximization of profit. This involves a detailed evaluation of each facility's production capabilities, costs, and the financial implications of servicing different customer segments, including the provision of onsite support.

2.1 Given Data

Fab Capacities :

- The production capacities of Fab A is 50
- The production capacities of Fab B is42

Production Cost :

- The production cost of one chip at Fab A is ௹ 1150
- The production cost of one chip at Fab B is ௹ 1250

Demand :

- The maximum demand of customer - 1 is 36 million chips
- The maximum demand of customer - 2 is 46 million chips
- The maximum demand of customer - 3 is 11 million chips
- The maximum demand of customer - 4 is 35 million chips

Offered Prices :

- Price offered by customer - 1 for each chip is ௹ 1950
- Price offered by customer - 2 for each chip is ௹ 1850
- Price offered by customer - 3 for each chip is ௹ 2000
- Price offered by customer - 4 for each chip is ௹ 1800

Delivery Cost (Including Onsite Customer Support) :

From Fab - A :

- Cost of delivery to customer - 1 is ௹ 300
- Cost of delivery to customer - 2 is ௹ 400
- Cost of delivery to customer - 3 is ௹ 550
- Cost of delivery to customer - 4 is ௹ 450

From Fab - B :

- Cost of delivery to customer - 1 is ௹ 600
- Cost of delivery to customer - 2 is ௹ 300
- Cost of delivery to customer - 3 is ௹ 400
- Cost of delivery to customer - 4 is ௹ 250

2.2 Assigning Decision Variables :

Let x_{ij} be a binary decision variable where i corresponds to the Fab - A and Fab - B (i= 1,2) and j corresponds to customers (j=1,2,3,4). The decision is allocate the chips produced at Fab - A and Fab - B to the customers optimally in order to maximize the profit.

2.3 Objective Function :

$$Maximize\, Z = \sum_{i=1}^{2}\sum_{j=1}^{4}\left(P_j - Dc_{ij} - Pc_i\right) * \; x_{ij}$$

$$x_{ij} \in \mathbb{Z}^{+} \text{ for } i = 1{,}2 \;\& j = 1{,}2{,}3{,}4$$

Where ,

- ***P*** is the offered price by the customer.
- ***Dc*** is the delivery cost incurred for every chip produced based on the Fab (*i*) and the customer (*j*).
- ***Pc*** is the production of every chip based on the Fab (*i*) in which it is produced.

2.4 Constraints :

2.4.1 Production Capacity Constraint:

The total chips shipped from each fab should not exceed its production capacity.

$$\sum_{j=1}^{4} x_{ij} \leq C_i\,,\ \forall i \ \in \{1, 2\}$$

Where ,

- ***C*** is the production capacity of each Fab in millions (*i*) .

2.4.2 Demand Constraint:

The total chips shipped to each customer should not exceed their maximum demand.

$$\sum_{i=1}^{2} x_{ij} \leq \ D[j]\,, \forall j \ \in \{1,2,3,4\}$$

Where,

- ➢ ***D*** is the maximum demand of chips by each customer in millions (*j*) .

2.5 Final Model :

$$Maximize\ Z = \sum_{i=1}^{2}\sum_{j=1}^{4}\left(P_j - Dc_{ij} - Pc_i\right) * \ x_{ij}$$

$$x_{ij} \in \mathbb{Z}^{+} \ \text{for}\ i = 1,\ 2{,}3{,}4{,}5\ \&\, j = 1{,}2{,}3{,}4$$

Subject to following Constraints :

Production Capacity Constraint:

$$\sum_{j=1}^{4} x_{ij} \leq C_i\,,\ \forall i\ \in \{1,\ 2\}$$

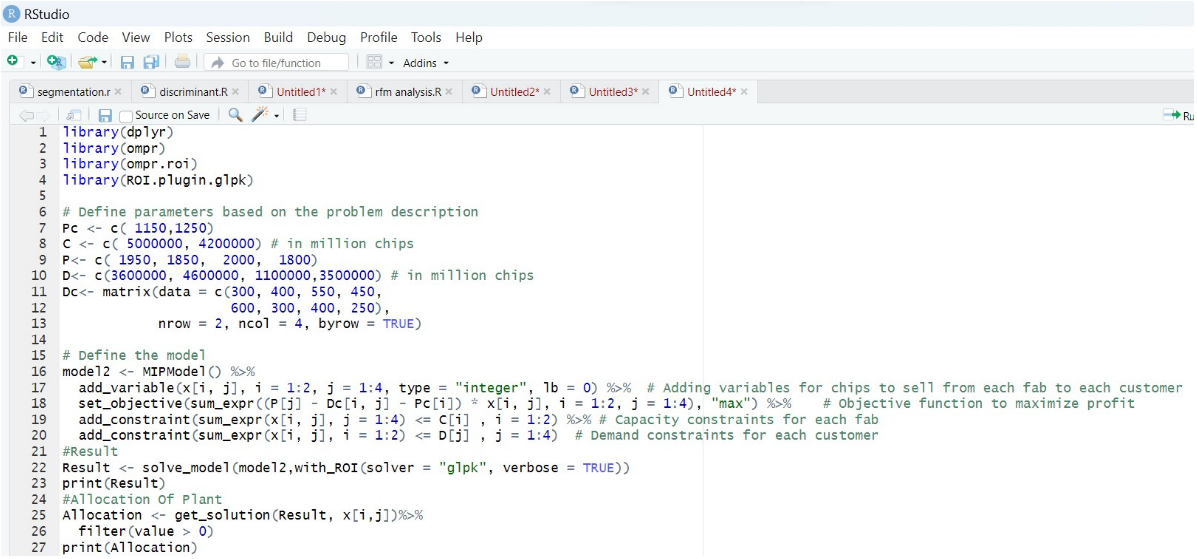
Demand Constraint:

$$\sum_{i=1}^{2} x_{ij} \leq\ D_j\,, \forall j\ \in \{1{,}2{,}3{,}4\}$$

2.6 Solving the Model :

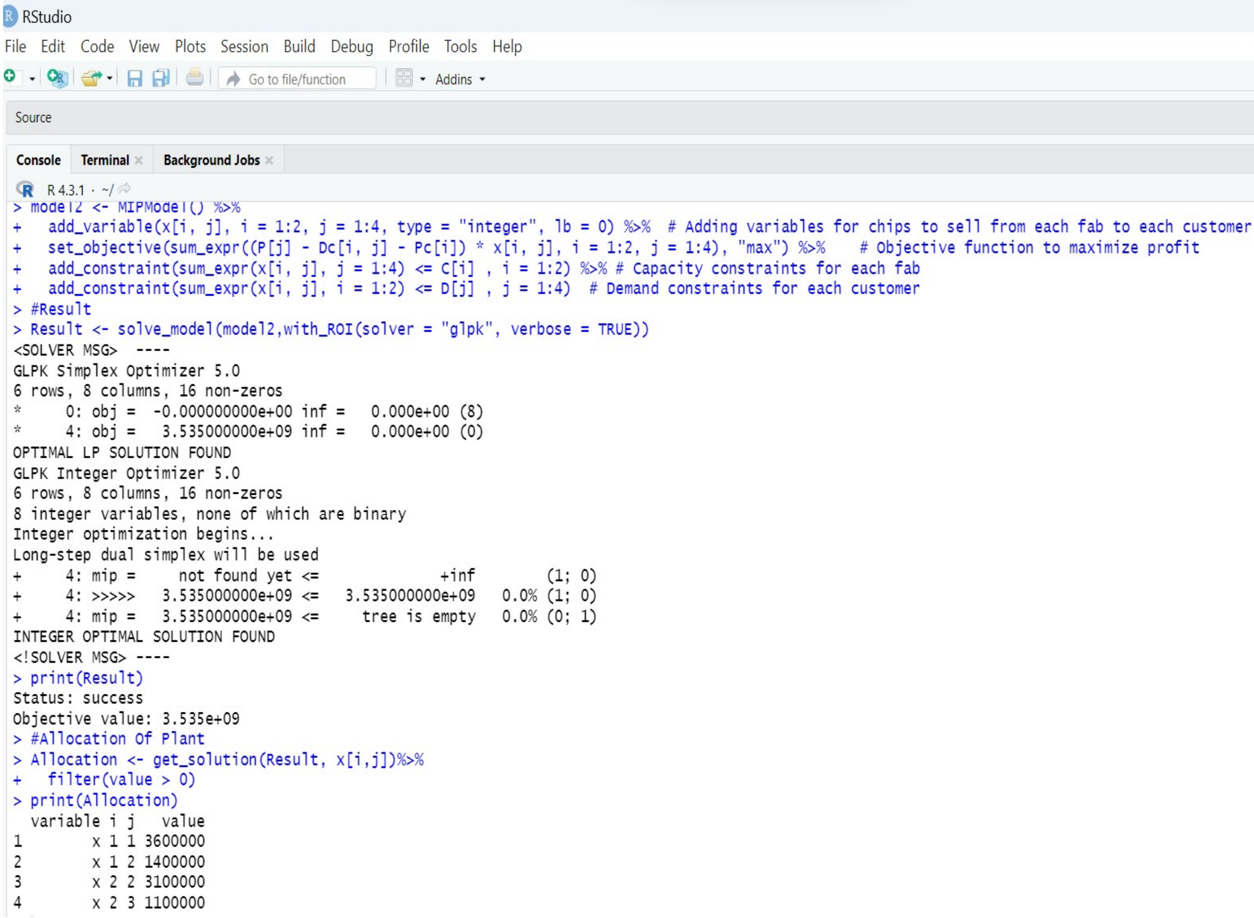
The mathematical model was crafted into an R script and executed using the `ompr` package, supplemented by additional packages like `ROI`, the `ROI plugin`, and `GLPK`, among others. This execution aimed to solve the mathematical model efficiently, aiming to ascertain the optimal solution that maximizes profit to Teranikx.

2.6.1 Implementation of Mathematical Model in R :



```
library(dplyr)
library(ompr)
library(ompr.roi)
library(ROI.plugin.glpk)

# Define parameters based on the problem description
Pc <- c( 1150,1250)
C <- c( 5000000, 4200000) # in million chips
P<- c( 1950, 1850,  2000,  1800)
D<- c(3600000, 4600000, 1100000,3500000) # in million chips
Dc<- matrix(data = c(300, 400, 550, 450,
                     600, 300, 400, 250),
            nrow = 2, ncol = 4, byrow = TRUE)

# Define the model
model2 <- MIPModel() %>%
  add_variable(x[i, j], i = 1:2, j = 1:4, type = "integer", lb = 0) %>%  # Adding variables for chips to sell from each fab to each customer
  set_objective(sum_expr((P[j] - Dc[i, j] - Pc[i]) * x[i, j], i = 1:2, j = 1:4), "max") %>%    # Objective function to maximize profit
  add_constraint(sum_expr(x[i, j], j = 1:4) <= C[i] , i = 1:2) %>% # Capacity constraints for each fab
  add_constraint(sum_expr(x[i, j], i = 1:2) <= D[j] , j = 1:4)  # Demand constraints for each customer
#Result
Result <- solve_model(model2,with_ROI(solver = "glpk", verbose = TRUE))
print(Result)
#Allocation Of Plant
Allocation <- get_solution(Result, x[i,j])%>%
  filter(value > 0)
print(Allocation)
```

2.6.2 R Output Solution :



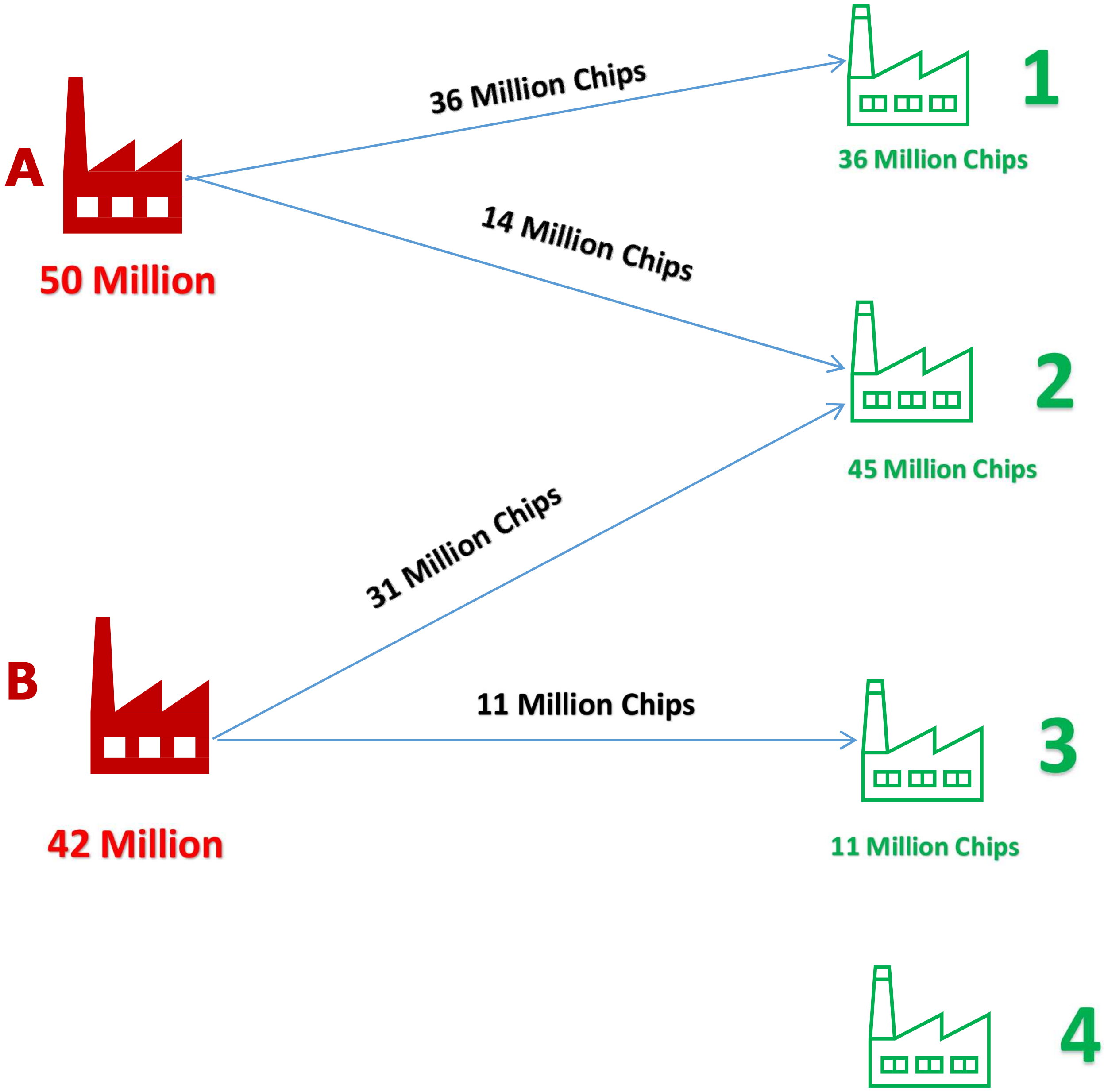
```
> model2 <- MIPModel() %>%
+   add_variable(x[i, j], i = 1:2, j = 1:4, type = "integer", lb = 0) %>%  # Adding variables for chips to sell from each fab to each customer
+   set_objective(sum_expr((P[j] - Dc[i, j] - Pc[i]) * x[i, j], i = 1:2, j = 1:4), "max") %>%    # Objective function to maximize profit
+   add_constraint(sum_expr(x[i, j], j = 1:4) <= C[i] , i = 1:2) %>% # Capacity constraints for each fab
+   add_constraint(sum_expr(x[i, j], i = 1:2) <= D[j] , j = 1:4)  # Demand constraints for each customer
> #Result
> Result <- solve_model(model2,with_ROI(solver = "glpk", verbose = TRUE))
<SOLVER MSG>  ----
GLPK Simplex Optimizer 5.0
6 rows, 8 columns, 16 non-zeros
*     0: obj =  -0.000000000e+00 inf =   0.000e+00 (8)
*     4: obj =   3.535000000e+09 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
GLPK Integer Optimizer 5.0
6 rows, 8 columns, 16 non-zeros
8 integer variables, none of which are binary
Integer optimization begins...
Long-step dual simplex will be used
+     4: mip =     not found yet <=              +inf        (1; 0)
+     4: >>>>>   3.535000000e+09 <=   3.535000000e+09   0.0% (1; 0)
+     4: mip =   3.535000000e+09 <=     tree is empty   0.0% (0; 1)
INTEGER OPTIMAL SOLUTION FOUND
<!SOLVER MSG> ----
> print(Result)
Status: success
Objective value: 3.535e+09
> #Allocation Of Plant
> Allocation <- get_solution(Result, x[i,j])%>%
+   filter(value > 0)
> print(Allocation)
  variable i j   value
1        x 1 1 3600000
2        x 1 2 1400000
3        x 2 2 3100000
4        x 2 3 1100000
```

2.7 Managerial Statement :

To maximize profits, Teranikx should adopt a strategy of selling the chip to its various customers accordingly,

- **36 million** Chips from **Fab - A** should be delivered to **customer -1**
- **14 million** Chips from **Fab - A** should be delivered to **customer -2**
- **31 million** Chips from **Fab - B** should be delivered to **customer -2**
- **11 million** Chips from **Fab - B** should be delivered to **customer -3**

Customer	1	2	3	4
Fab - A	3600000	1400000	-	-
Fab - B	-	3100000	1100000	-



The highest possible profit from this deal would amount to රු 35,350 million.

3) Make-to-Stock Chemotherapy Drugs

Apotheeker Pharmaceuticals, a producer of two kinds of chemotherapy medications named Chemo1 and Chemo2 for stock, aims to maximize its monthly earnings. To achieve this, the company must figure out the best amounts of basic drugs, procured from both the EU and the US, to mix into these two drug varieties. It must navigate various limitations including stock levels, market demand, shipping concerns, and the drugs' quality standards. Additionally, Apotheeker Pharmaceuticals has to determine the drugs' D-metrics and P-metrics.

3.1 Assigning The Decision Variables :

Let $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ represent the quantities of Chemo-1 and Chemo-2 vials, respectively, using the European Constituent.

Similarly $\boldsymbol{x_3}$ and $\boldsymbol{x_4}$ represent the quantities of Chemo-1 and Chemo-2 vials, respectively, using the US Constituent.

$$\boldsymbol{x_1, x_2, x_3, x_4} \in \mathbb{Z}^+$$

3.2 Objective Function :

The goal is to increase the monthly earnings to their highest level, calculated as the difference between the income generated from sales and the expenses associated with producing the drugs and acquiring their components. This income is determined by multiplying the selling price by the quantity sold for each drug, while the costs are figured by multiplying the per-unit price of each ingredient (EU and US constituents) by the amount used.

$$\boldsymbol{Maximize\ Z = ((1200 * (x_1 + x_3)) + (1400 * (x_2 + x_4))) - ((800 * (x_1 + x_2)) + (1500 * (x_3 + x_4)))}$$

$$x_1, x_2, x_3, x_4 \in \mathbb{Z}^+$$

3.3 Constraints :

3.3.1 Maximum Demand Constraints :

$$\boldsymbol{x_1 + x_3 \leq 200000}$$
$$\boldsymbol{x_2 + x_4 \leq 40000}$$

$$x_1, x_2, x_3, x_4 \in \mathbb{Z}^+$$

3.3.2 Minimum Delivery Constraints :

$$\boldsymbol{x_1 + x_3 \quad \geq 100000}$$

$$\boldsymbol{x_2 + x_4 \quad \geq 10000}$$

$$x_1, x_2, x_3, x_4 \in \mathbb{Z}^+$$

3.3.3 Inventory Constraints :

$$\boldsymbol{x_1 + x_2 \leq 80000}$$

$$\boldsymbol{x_3 + x_4 \quad \leq 120000}$$

$$x_1, x_2, x_3, x_4 \in \mathbb{Z}^+$$

3.3.4 Maximum D-Metric Constraints :

Chemo -1

$$\frac{25x_1}{x_1 + x_3} + \frac{15x_3}{x_1 + x_3} \leq 23$$

$$\frac{25x_1 + 15x_3}{x_1 + x_3} \leq 23$$

$$\boldsymbol{25x_1 + 15x_3 \leq 23 * (x_1 + x_3)}$$

Chemo -2

$$\frac{25x_2}{x_2 + x_4} + \frac{15x_4}{x_2 + x_4} \leq 23$$

$$\frac{25x_2 + 15x_4}{x_2 + x_4} \leq 23$$

$$\boldsymbol{25x_2 + 15x_4 \leq 23* (x_2 + x_4)}$$

3.3.5 Minimum P-Metric Constraints :

Chemo -1

$$\frac{87x_1}{x_1+x_3} + \frac{98x_3}{x_1+x_3} \geq 88$$

$$\frac{87x_1+98x_3}{x_1+x_3} \geq 88$$

$$\boldsymbol{87x_1 + 98x_3 \geq 88 * (x_1 + x_3)}$$

Chemo -2

$$\frac{87x_2}{x_2+x_4} + \frac{98x_4}{x_2+x_4} \geq 93$$

$$\frac{87x_2+98x_4}{x_2+x_4} \geq 93$$

$$\boldsymbol{87x_2 + 98x_4 \geq 93 * (x_2 + x_4)}$$

3.4 Final Model :

$$\boldsymbol{Maximize\ Z = ((1200 * (x_1 + x_3)) + (1400 * (x_2 + x_4))) - ((800 * (x_1 + x_2)) + (1500 * (x_3 + x_4)))}$$

$$x_1, x_2, x_3, x_4 \in \mathbb{Z}^+$$

Subject to following Constraints :

Maximum Demand Constraints :

$$x_1 + x_3 \ \leq \ 200000$$
$$x_2 + x_4 \leq 40000$$

Minimum Delivery Constraints :

$$x_1 + x_3 \quad \geq 100000$$
$$x_2 + x_4 \quad \geq 10000$$

Inventory Constraints :

$$x_1 + x_2 \leq 80000$$
$$x_3 + x_4 \quad \leq 120000$$

Maximum D-Metric Constraints :

$$25x_1 + 15x_3 \ \leq \ 23 * \ (x_1 + x_3)$$
$$25x_2 + 15x_4 \ \leq \ 23* (x_2 + x_4)$$

Minimum P-Metric Constraints :

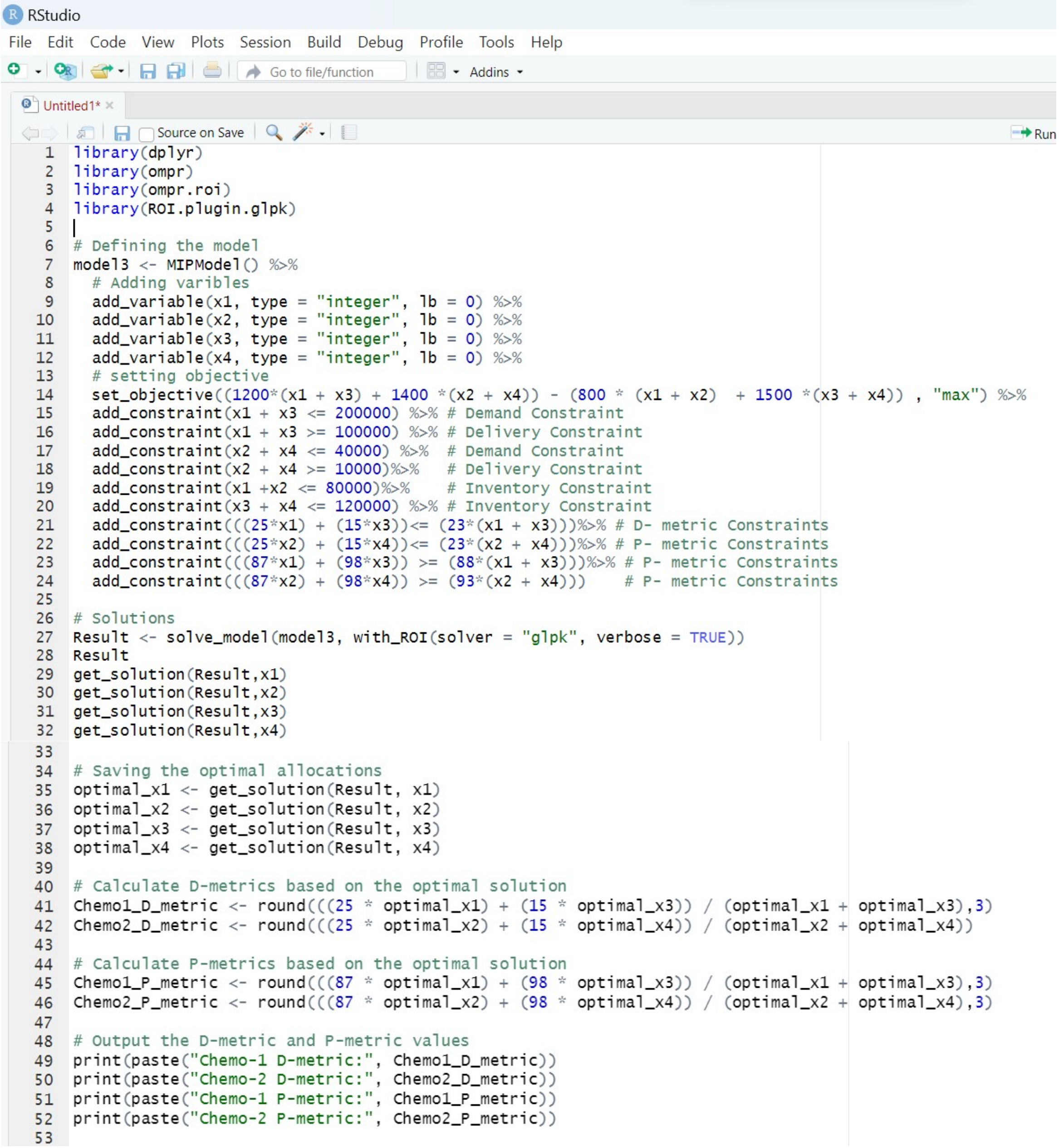
$$87x_1 + 98x_3 \ \geq 88 * \ (x_1 + x_3)$$

$$87x_2 + 98x_4 \ \geq 93 * \ (x_2 + x_4)$$

$$\boldsymbol{x_1, x_2, x_3, x_4} \in \mathbb{Z}^+$$

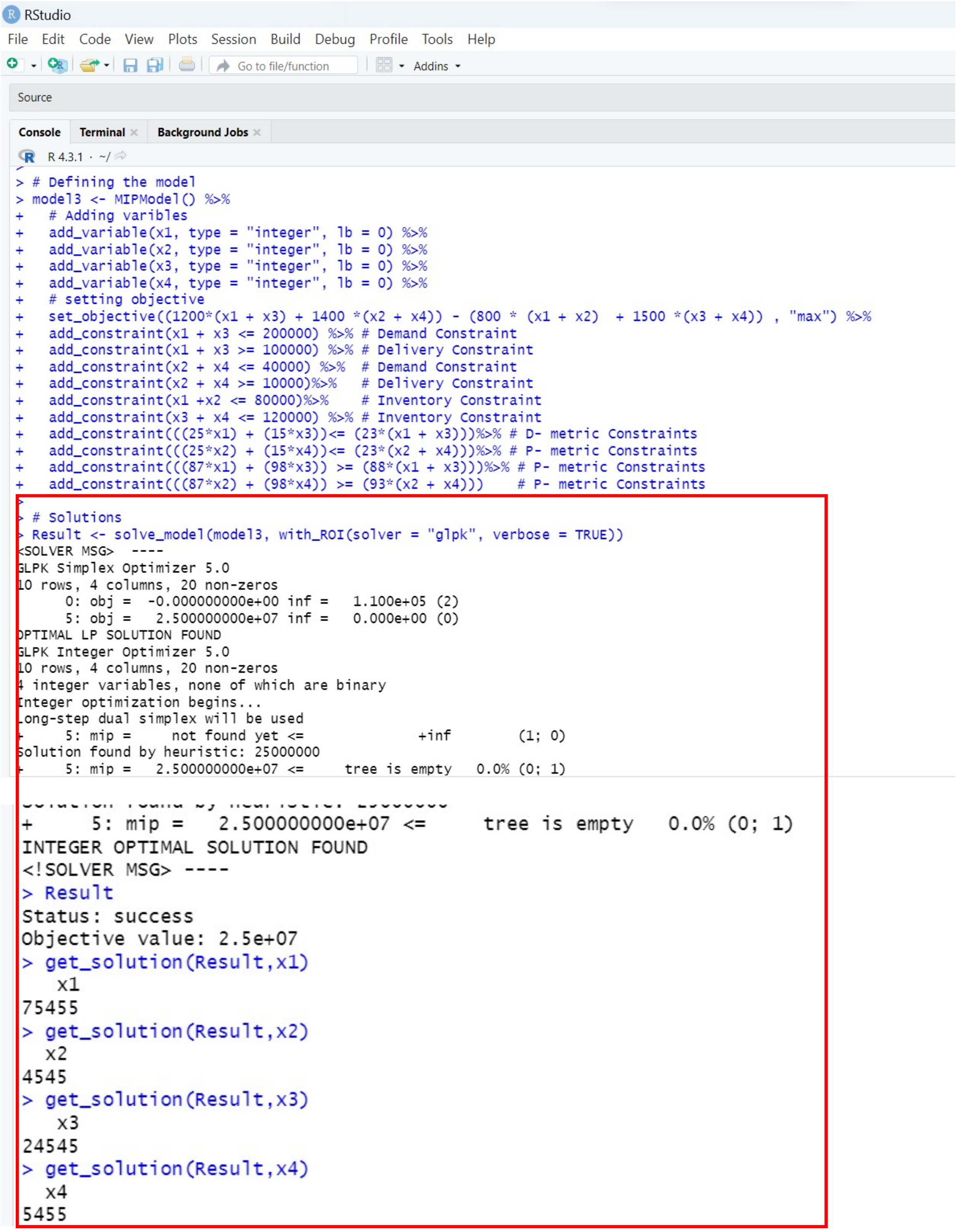
3.5 Solving the Model :

3.5 .1 Implementation of Mathematical Model in R :

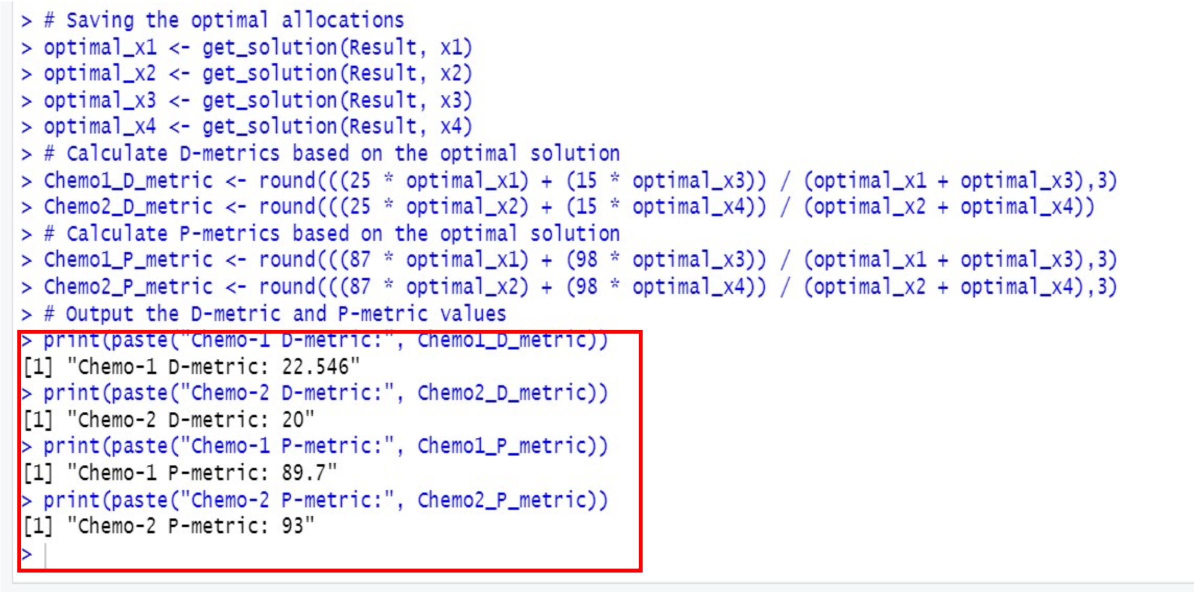


```
library(dplyr)
library(ompr)
library(ompr.roi)
library(ROI.plugin.glpk)

# Defining the model
model3 <- MIPModel() %>%
  # Adding varibles
  add_variable(x1, type = "integer", lb = 0) %>%
  add_variable(x2, type = "integer", lb = 0) %>%
  add_variable(x3, type = "integer", lb = 0) %>%
  add_variable(x4, type = "integer", lb = 0) %>%
  # setting objective
  set_objective((1200*(x1 + x3) + 1400 *(x2 + x4)) - (800 * (x1 + x2)  + 1500 *(x3 + x4)) , "max") %>%
  add_constraint(x1 + x3 <= 200000) %>% # Demand Constraint
  add_constraint(x1 + x3 >= 100000) %>% # Delivery Constraint
  add_constraint(x2 + x4 <= 40000) %>%  # Demand Constraint
  add_constraint(x2 + x4 >= 10000)%>%   # Delivery Constraint
  add_constraint(x1 +x2 <= 80000)%>%    # Inventory Constraint
  add_constraint(x3 + x4 <= 120000) %>% # Inventory Constraint
  add_constraint(((25*x1) + (15*x3))<= (23*(x1 + x3)))%>% # D- metric Constraints
  add_constraint(((25*x2) + (15*x4))<= (23*(x2 + x4)))%>% # P- metric Constraints
  add_constraint(((87*x1) + (98*x3)) >= (88*(x1 + x3)))%>% # P- metric Constraints
  add_constraint(((87*x2) + (98*x4)) >= (93*(x2 + x4)))    # P- metric Constraints

# Solutions
Result <- solve_model(model3, with_ROI(solver = "glpk", verbose = TRUE))
Result
get_solution(Result,x1)
get_solution(Result,x2)
get_solution(Result,x3)
get_solution(Result,x4)

# Saving the optimal allocations
optimal_x1 <- get_solution(Result, x1)
optimal_x2 <- get_solution(Result, x2)
optimal_x3 <- get_solution(Result, x3)
optimal_x4 <- get_solution(Result, x4)

# Calculate D-metrics based on the optimal solution
Chemo1_D_metric <- round(((25 * optimal_x1) + (15 * optimal_x3)) / (optimal_x1 + optimal_x3),3)
Chemo2_D_metric <- round(((25 * optimal_x2) + (15 * optimal_x4)) / (optimal_x2 + optimal_x4))

# Calculate P-metrics based on the optimal solution
Chemo1_P_metric <- round(((87 * optimal_x1) + (98 * optimal_x3)) / (optimal_x1 + optimal_x3),3)
Chemo2_P_metric <- round(((87 * optimal_x2) + (98 * optimal_x4)) / (optimal_x2 + optimal_x4),3)

# Output the D-metric and P-metric values
print(paste("Chemo-1 D-metric:", Chemo1_D_metric))
print(paste("Chemo-2 D-metric:", Chemo2_D_metric))
print(paste("Chemo-1 P-metric:", Chemo1_P_metric))
print(paste("Chemo-2 P-metric:", Chemo2_P_metric))
```

3.5 .2 R Output Solution :



```
> # Defining the model
> model3 <- MIPModel() %>%
+   # Adding varibles
+   add_variable(x1, type = "integer", lb = 0) %>%
+   add_variable(x2, type = "integer", lb = 0) %>%
+   add_variable(x3, type = "integer", lb = 0) %>%
+   add_variable(x4, type = "integer", lb = 0) %>%
+   # setting objective
+   set_objective((1200*(x1 + x3) + 1400 *(x2 + x4)) - (800 * (x1 + x2)  + 1500 *(x3 + x4)) , "max") %>%
+   add_constraint(x1 + x3 <= 200000) %>% # Demand Constraint
+   add_constraint(x1 + x3 >= 100000) %>% # Delivery Constraint
+   add_constraint(x2 + x4 <= 40000) %>%  # Demand Constraint
+   add_constraint(x2 + x4 >= 10000)%>%   # Delivery Constraint
+   add_constraint(x1 +x2 <= 80000)%>%    # Inventory Constraint
+   add_constraint(x3 + x4 <= 120000) %>% # Inventory Constraint
+   add_constraint(((25*x1) + (15*x3))<= (23*(x1 + x3)))%>% # D- metric Constraints
+   add_constraint(((25*x2) + (15*x4))<= (23*(x2 + x4)))%>% # P- metric Constraints
+   add_constraint(((87*x1) + (98*x3)) >= (88*(x1 + x3)))%>% # P- metric Constraints
+   add_constraint(((87*x2) + (98*x4)) >= (93*(x2 + x4)))    # P- metric Constraints
>
> # Solutions
> Result <- solve_model(model3, with_ROI(solver = "glpk", verbose = TRUE))
<SOLVER MSG>  ----
GLPK Simplex Optimizer 5.0
10 rows, 4 columns, 20 non-zeros
      0: obj =  -0.000000000e+00 inf =   1.100e+05 (2)
      5: obj =   2.500000000e+07 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
GLPK Integer Optimizer 5.0
10 rows, 4 columns, 20 non-zeros
4 integer variables, none of which are binary
Integer optimization begins...
Long-step dual simplex will be used
+     5: mip =     not found yet <=              +inf        (1; 0)
Solution found by heuristic: 25000000
+     5: mip =   2.500000000e+07 <=     tree is empty   0.0% (0; 1)
+     5: mip =   2.500000000e+07 <=     tree is empty   0.0% (0; 1)
INTEGER OPTIMAL SOLUTION FOUND
<!SOLVER MSG> ----
> Result
Status: success
Objective value: 2.5e+07
> get_solution(Result,x1)
   x1
75455
> get_solution(Result,x2)
  x2
4545
> get_solution(Result,x3)
   x3
24545
> get_solution(Result,x4)
  x4
5455
```



```
> # Saving the optimal allocations
> optimal_x1 <- get_solution(Result, x1)
> optimal_x2 <- get_solution(Result, x2)
> optimal_x3 <- get_solution(Result, x3)
> optimal_x4 <- get_solution(Result, x4)
> # Calculate D-metrics based on the optimal solution
> Chemo1_D_metric <- round(((25 * optimal_x1) + (15 * optimal_x3)) / (optimal_x1 + optimal_x3),3)
> Chemo2_D_metric <- round(((25 * optimal_x2) + (15 * optimal_x4)) / (optimal_x2 + optimal_x4))
> # Calculate P-metrics based on the optimal solution
> Chemo1_P_metric <- round(((87 * optimal_x1) + (98 * optimal_x3)) / (optimal_x1 + optimal_x3),3)
> Chemo2_P_metric <- round(((87 * optimal_x2) + (98 * optimal_x4)) / (optimal_x2 + optimal_x4),3)
> # Output the D-metric and P-metric values
> print(paste("Chemo-1 D-metric:", Chemo1_D_metric))
[1] "Chemo-1 D-metric: 22.546"
> print(paste("Chemo-2 D-metric:", Chemo2_D_metric))
[1] "Chemo-2 D-metric: 20"
> print(paste("Chemo-1 P-metric:", Chemo1_P_metric))
[1] "Chemo-1 P-metric: 89.7"
> print(paste("Chemo-2 P-metric:", Chemo2_P_metric))
[1] "Chemo-2 P-metric: 93"
>
```

3.6 Managerial Statement :

To achieve a maximum monthly profit of € **25 million**, Apotheeker needs to blend 75,455 vials of the EU constituent with 24,545 vials of the US constituent to create 100,000 vials of Chemo-1. Additionally, for the production of 10,000 vials of Chemo-2, 4,545 vials of the EU constituent must be mixed with 5,455 vials of the US constituent.

Constituents	Chemo-1	Chemo-2
EU	75455	4545
US	24545	5455
Total	100000	10000

The D-metrics of Chemo - 1 and Chemo - 2 are 22.54 and 20, respectively

The P-metrics of Chemo - 1 and Chemo - 2 are 89.7 and 93 , respectively

Metric	Chemo-1	Chemo-2
D- Metric	22.54	20
P- Metric	89.7	93