Rotterdam School of Management
Erasmus University

Prescriptive Analytics Assignment 3 – Report

Group 15

Khoi Pham – 523755kg
Nam Hoang Giap – 534598hg
Hong Minh Nguyen – 532460mn

Table of Contents

Question 1

In this assignment, we set set.seed(15)

1a. Random swap between two items

Heuristic steps

```
Create start solution initial_sol

Best found solution best_sol_1  = initial_sol

Best found revenue best_revenue_1= obj(initial_sol)

While (7000 iterations)

        h <- vector index of 2 random to-be-swapped items

        initial_sol_1 = neighborhood(best_sol_1)  (takes vector h to swap items)

        if obj(initial_sol_1) > obj(best_sol_1) (for maximization problem)

                best_revenue_1= obj(initial_sol_1) (to store the best found revenue)

                best_sol_1  = initial_sol_1 (to store the best found solution)

        End if

End while

Return best_sol_1 as best found solution and best_revenue_1 as best found objective value
```

Best solution

```
> best_sol_1
 [1]   4   7   8   9  16  17  21  24  27  40  41  43  46  64  68  70  82  94  95  96 104
[22] 109 111 117 122 127 128 130 146 151 156 159 163 171 183 184 185 192 194 198
```

Total revenue

```
> best_revenue_1
[1] 431.6935
```

Initial solution

```
> initial_sol
 [1]  37 106 162  38 177   5 107  84 128 140 200  25 138 181  99 130 198  19  72 194  30
```

```
[22]  21  59 165  79  27  86   2 142  10  33 192  39  92   6  98  23 123  49  75
```

Initial revenue

```
> initial_revenue
[1] 325.0633
```

1b. Remove an item with the lowest value and add a random item

Heuristic steps

```
Create start solution initial_sol
Best found solution best_sol_2  = initial_sol
Best found revenue best_revenue_2= obj(initial_sol)
While (7000 iterations)
    h <- vector index of an lowest value chosen item and a random to-be-swapped item
    initial_sol_2 = neighborhood(best_sol_2)  (takes vector h to swap items)
    if obj(initial_sol_2) > obj(best_sol_2) (for maximization problem)
        best_revenue_2= obj(initial_sol_2) (to store the best found revenue)
        best_sol_2  = initial_sol_2 (to store the best found solution)
    End if
End while
Return best_sol_2 as best found solution and best_revenue_2 as best found objective value
```

Best solution

```
> best_sol_2
 [1]   4   6   7  18  21  29  35  37  41  43  46  52  61  70  72  76  77  80  94  96  99
[22] 104 109 111 121 135 139 145 150 162 165 172 173 176 180 183 185 189 194 198
```

Total revenue

```
> best_revenue_2
[1] 598.8685
```

Conclusion

The solution with the heuristic search implemented in sub-question b (Method B) has improved compared to that of sub-question a (Method A).

Method B generates better solutions because of its more advanced method and requirements when choosing and replacing items (Pisinger & Røpke, 2010). Indeed,

- Method B helps find reasonable solutions in a smarter way. By continuously replacing the lowest-revenue item with a random one in the non-selected set of items, Method B can create a set of revenue a_i (**35.01865**) higher than Method A (**18.12953**), under 7000 iterations in both cases.
- Method B requires a smaller number of iterations to reach the optimal solution.
    - Instead of replacing two random items, Method B finds the lowest revenue item and replaces it continuously, which lessens the process of finding the list of items with the highest total revenue. On the other hand, Method A has a greater pool of item-replacement options because every item is considered, which could be more time-consuming and require more iterations to generate optimal revenue.
    - We also try increasing the number of iterations. Consequently, Method A results in different revenue. For example, Method A will generate a revenue of **438.6444** with **10,000** iterations and **446.3676** with **20,000** iterations. Meanwhile, the revenue of Method B with **10,000** iterations and **20,000** iterations remains unchanged, **598.8685**, compared to that of **7000** iterations.

Thus, in this case, the search heuristics method in sub-question b would generate better solutions.

***References**:*

Pisinger, D., & Røpke, S. (2010). Large Neighborhood Search. In M. Gendreau (Ed.), Handbook of Metaheuristics (2 ed., pp. 399-420). Springer. http://www.springerlink.com.globalproxy.cvt.dk/content/978-14419-1663-1#section=777803&page=1

1c. Remove a random item and add an item with the highest value

These heuristic steps are motivated by our intentions to reflect on Neighborhood search's tendency to end up in local optimal.

Heuristics steps

```
Create start solution initial_sol

Best found solution best_sol_3 = initial_sol

Best found revenue best_revenue_3 = obj(initial_sol)

While (7000 iterations)

    h <- vector index of a random chosen item and a highest value unchosen item

    initial_sol_3 = neighborhood(best_sol_3)  (takes vector h to swap items)

    if obj(initial_sol_3) > obj(best_sol_3) (for maximization problem)

        best_revenue_3= obj(initial_sol_3) (to store the best found revenue)

        best_sol_3 = initial_sol_3 (to store the best found solution)

    End if

End while

Return best_sol_3 as best found solution and best_revenue_3 as best found objective value
```

Best solution

```
> best_sol_3
 [1]   4   6   7  18  21  29  35  37  41  43  46  52  61  70  72  76  77  80  94  96  99
[22] 104 109 111 121 135 139 150 162 164 172 173 176 180 183 185 189 194 198 200
```

Total revenue

```
> best_revenue_3
[1] 593.4217
```

1d. Scenario that it is possible to include an item up to 2 pages

Changes in the model

- The quadratic objective function remains the same, only the decision variable xi to whether to include an item or not changes into:

$x_i \in \{0,1,2\}$ *for* $i = 1, \dots, 200$

- As for the combination effect:

If item i is on 2 pages and item j on 1 page, the additional revenue bij changes by delta = 2*bij

If both i and j are on 2 pages, the additional revenue bij changes by delta = 4*bij

Initial solution

The method to generate initial solution changes into: "40 items are randomly selected to be included in the folder with a maximum two times duplication, instead of one". *(In our R script, we duplicate the vector index of 200 items, then we draw sample the same as from previous questions)*

```
> initial_sol_1d
 [1]  37 162 162  94 177  61  17   4 163  84  30 193 184 140  45 200  15  81  18  13 181
[22] 155  19 128 107  86  21 115 165  79  83  86   2 142 186  10   1 173   9 189
```

The item that has duplicate:

```
> initial_sol_1d[duplicated(initial_sol_1d)]
[1] 162  86
```

Initial revenue

```
> initial_revenue_1d
[1] 341.6166
```

Heuristics steps

```
Create start solution initial_sol_1d (items that are duplicates will have decision variable = 2)
Best found solution best_sol_4 = initial_sol_1d
Best found revenue best_revenue_4 = obj(initial_sol_1d)
While (7000 iterations)
    h <- vector index of a random chosen item and a highest value unchosen item
    initial_sol_4 = neighborhood(best_sol_4)  (takes vector h to swap items, if row index = column index: decision variable = decision variable of item i )
    if obj(initial_sol_4) > obj(best_sol_4) (for maximization problem)
        best_revenue_4= obj(initial_sol_4) (to store the best found revenue)
        best_sol_4 = initial_sol_4 (to store the best found solution)
    End if
End while
Return best_sol_4 as best found solution and best_revenue_4 as best found objective value
```

Best solution

```
> best_sol_4
 [1]   1   2   4   9  10  13  15  17  18  19  21  30  37  45  61  79  81  83  84  94 107
[22] 115 121 121 128 140 142 155 162 162 163 165 173 177 181 184 186 189 193 200
```

Total revenue

```
> best_revenue_4
[1] 378.4786
```

Conclusion

- The initial revenue in this scenario is higher than the initial revenue calculated in the previous - question (initial_revenue = 325.0633). The difference can be explained by the fact that the diagonal value ai (single item value) is always positive. Thus, when a single item is duplicated, the total revenue may avoid the possibility of negative effects from combinations of items.

- However, the optimal solution after performing the Heuristic search in this scenario is lower than that in question 1d, for the following reasons. First, the heuristic steps in 1c lead to optimal solution being trapped in local optima (the loops always choose the highest item value from outside to swap with a randomly selected item). Second, when combined, these items, although have higher single value ai, may worsen the total revenue due to the negative effect of item combinations. Thirdly, as described in section' changes in model' above, when items are duplicated, marginal changes in additional revenue would escalate up to 4 times.

Question 2:

The package we used to draw random numbers from triangular distribution is "EnvStats", function rtri().

Psuedocode

```
Create a list containing 100 Monte Carlo simulated dataframes: MC100
Create start solution initial_sol (same initial solution from 1a)
For (l in 1:number of realizations)
    Best found solution best_sol_q2 = initial_sol
    Best found revenue best_revenue_q2 = obj(matrix at l th position in list MC100 )
    While (200 iterations)
        h <- vector index of a random chosen item and a highest value unchosen item
        initial_sol_q2 = neighborhood(best_sol_q2) (takes vector h to swap items)
        if obj(initial_sol_q2) > obj(best_sol_q2) (for maximization problem)
```

```
                best_revenue_q2 = obj(best_sol_q2) (to store the best found revenue)
                best_sol_q2 = initial_sol_q2 (to store the best found solution)
            End if
        End while
Store best found revenue best_revenue_q2 of each realization to a vector mc_obj
End for
Return mean(mc_obj) as average objective values over 100 realizations
```

```
> mc_obj
  [1] 423.4707 354.3165 380.3303 423.5385 430.7509 401.4235 404.9080 447.2792 382.4643
 [10] 349.4348 407.6136 386.6500 401.8501 460.2476 411.3773 345.9782 365.8266 423.3285
 [19] 443.9307 392.8508 384.8464 411.6703 389.4290 411.4534 432.5695 360.9686 486.4821
 [28] 424.7738 422.7804 411.3222 392.2788 368.7088 368.9813 509.0720 424.4658 350.9292
 [37] 391.0210 388.9570 415.3823 441.3279 347.1596 389.8642 402.8282 372.0169 412.7454
 [46] 356.9055 456.3011 379.0888 381.4316 424.8854 381.3380 339.8884 430.8598 424.3145
 [55] 428.1607 493.3718 372.2260 467.8907 358.1335 406.4103 440.0337 392.3221 467.0681
 [64] 378.8331 389.1685 407.3497 374.2119 374.7841 346.9731 446.9027 405.8711 311.2482
 [73] 432.9277 440.0182 442.4556 447.4350 414.8713 353.0529 435.0235 382.2632 361.6120
 [82] 432.8461 410.6990 365.8429 388.6450 425.5520 455.8668 329.2443 389.2454 358.5332
 [91] 469.9540 319.6806 463.7605 378.6406 391.8857 406.1760 378.7486 351.2600 392.1713
[100] 361.4408
> avg_mc_obj
[1] 401.6543
> best_revenue_1
[1] 431.6935
```

Conclusion

In this assignment solely, the objective value from question 1a (best_revenue_1) is higher than the average objective value over 100 Monte Carlo simulated realizations (avg_mc_obj). Since our coefficients are uncertain (be randomly drawn from triangular distribution), the average objective value may vary when we set a different random seed.