Assignment 6 Integer programming

Ananth Kumar

19/11/2021

library(rmarkdown)

.libPaths("C:\Users\Ananth\OneDrive\Desktop\MSBA Kent\Fall 2021\Fundamentals of Machine Learning\Assignment\Ass 2")

Q1

Decision variable:

Xij = 1, the arc from node i to node j is chosen in the optimal (longest) path otherwise Xij = 0 Objective Function:

Maximize the total time required from node 1 to node 9: Max. Z = sigma(cij)(Xij)

Where, cij = time taken by arc (activity) from ith node and jth node

Max Z. = 5X12 + 3X13 + 3X35 + 2X25 + 4X24 + 4X47 + 1X46 + 2X58 + 6X57 + 5X69 + 4X79 + 7X89 Constraint:

For longest route problem, following constraint are to be satisfied, For origin Node 1, outgoing arc is equal to 1, X12 + X13 = 1

For intermediate nodes, Arc in = arc out

For node 2: X12 = X25 + X24 or X12 – X25 - X24 = 0
For node 3: X13 = X35 or X13 – X35 = 0 For node 4: X24 = X46 + X47 or X24 – X46 – X47 = 0 For node 5: X25 + X35 = X57 + X58 or X25 + X35 - X57 - X58 = 0 For node 6: X46 = X69 or X46 – X69 = 0 For node 7: X57 + X47 = X79 or X57 + X47 – X79 = 0 For node 8: X58 = X89 or X58 – X89 = 0

For destination node: Arc in = 1 For node 9, X69 + X79 + X89 = 1

```r
library(lpSolve)
library(lpSolveAPI)
Q1 <- read.lp("Ques1.lp")
```

```r
solve(Q1)
```

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

```r
get.objective(Q1) # longest path value is 17
```

```
## [1] 17
```

```
get.variables(Q1) # each variable x12,x13,x35,x25,x24,x47,x46,x58,x57,x69,x79,x89 values, 1 indicates t
```

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

LP equation for the problem 1

// Objective function

max: 5 * x12 + 3 * x13 + 3 * x35 + 2 * x25 + 4 * x24 + 4 * x47 + x46 + 2 * x58 + 6 * x57 + 5 * x69 + 4 * x79 + 7 * x89;

// Constraints x12 + x13 = 1; x12 - x24 - x25 = 0; x13 - x35 = 0; x24 - x46 - x47 = 0; x25 + x35 - x57 - x58 = 0; x46 - x69 = 0; x57 + x47 - x79 = 0; x58 - x89 = 0; x69 + x79 + x89 = 1;

bin x12,x13,x35,x25,x24,x47,x46,x58,x57,x69,x79,x89;

Q2

Selecting an Investment Portfolio

The objective function should show maximum return to the amount invested based on the given constraints.

/*Objective function*/ max: 4 s1 + 6.50 s2 + 5.9 s3 + 5.4 h1 + 5.15 h2 + 10 h3 + 8.4 c1 + 6.25 c2;

/*Constraints* /

Limiting the total amount invested <= 2.5 million, 40 s1 + 50 s2 + 80 s3 +60 h1 + 45 h2 + 60 h3 + 30 c1 + 25 c2 <= 2500000;

Limiting the total amount invested per sector to <= 1 million , the stock price * number of shares of each company in a sector <= 1 million 40 s1 + 50 s2 + 80 s3 <= 1000000; 60 h1 + 45 h2 + 60 h3<= 1000000; 30 c1 + 25 c2 <= 1000000;

Each company should atleat have 100,000 investment. the stock price * number of shares of a company <= 100,000 40 s1 >= 100000; 50 s2 >= 100000; 80 s3 >= 100000; 60 h1 >= 100000; 45 h2 >= 100000; 60 h3 >= 100000; 30 c1 >= 100000; 25 c2 >= 100000;

To get stock of each company which should be multiplier of 1000, we introduce a n1 which is an interger, which will make sure that the stock are multiples of 1000 s1 = 1000 n1; s2 = 1000 n2; s3 = 1000 n3; h1 = 1000 n4; h2 = 1000 n5; h3 = 1000 n6; c1 = 1000 n7; c2 = 1000 n8;

Minimum stock should be 1, making sure that the amount is positive. We do not want any negative values. n1 >= 1; n2 >= 1; n3 >= 1; n4 >= 1; n5 >= 1; n6 >= 1; n7 >= 1; n8 >= 1;

n1 to n8 should be a positive integer to get share multiplier of 1000 int n1, n2, n3, n4, n5, n6, n7, n8;

```
Question2P2 <-read.lp("Nointeger.lp")

solve(Question2P2)
```

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

```
get.objective(Question2P2)
```

```
## [1] 477050
```

```
get.variables(Question2P2)
```

```
##  [1]  4000  4000  2000  2000  3000 12000 30000  4000     4     4     2     2
## [13]     3    12    30     4
```

__1) Determine the maximum return on the portfolio. What is the optimal number of shares to buy for each of the stocks? What is the corresponding dollar amount invested in each stock?___

No integer restriction

Total stocks owned in s1 = 4000 , Amount = 4000 * 40 = 160,000 $ Total stocks owned in s2 = 4000 , Amount = 4000 * 50 = 200,000 $
Total stocks owned in s3 = 2000 , Amount = 2000 * 80 = 160,000 $ Total stocks owned in h1 = 2000 , Amount = 2000 * 60 = 120,000 $ Total stocks owned in h2 = 3000 , Amount = 3000 * 45 = 135,000 $ Total stocks owned in h3 = 12000, Amount = 12000 * 60= 720,000 $ Total stocks owned in c1 = 30000, Amount = 30000 * 30= 900,000 $ Total stocks owned in c2 = 4000 , Amount = 4000 * 25 = 100,000 $

Maximum return on portfolio: 477050

Return per share for the amount invested in s1 = 4 Return per share for the amount invested in s2 = 6.5 Return per share for the amount invested in s3 = 5.9 Return per share for the amount invested in h1 = 5.4 Return per share for the amount invested in h2 = 5.15 Return per share for the amount invested in h3 = 10 Return per share for the amount invested in c1 = 8.4 Return per share for the amount invested in c2 = 6.25

Maximum return = 4+6.5+5.9+5.4+5.15+10+8.4+6.25 = 51.6 (cumulative return including all stocks growth + dividend)

Limiting the total amount invested <= 2.5 million, 40 s1 + 50 s2 + 80 s3 +60 h1 + 45 h2 + 60 h3 + 30 c1 + 25 c2 <= 2500000;

Limiting the total amount invested per sector to <= 1 million , the stock price * number of shares of each company in a sector <= 1 million 40 s1 + 50 s2 + 80 s3 <= 1000000; 60 h1 + 45 h2 + 60 h3<= 1000000; 30 c1 + 25 c2 <= 1000000;

Each company should atleat have 100,000 investment. the stock price * number of shares of a company <= 100,000 40 s1 >= 100000; 50 s2 >= 100000; 80 s3 >= 100000; 60 h1 >= 100000; 45 h2 >= 100000; 60 h3 >= 100000; 30 c1 >= 100000; 25 c2 >= 100000;

To get stock of each company which should be multiplier of 1000, we introduce a n1 which is an interger, which will make sure that the stock are multiples of 1000 s1 = 1000 n1; s2 = 1000 n2; s3 = 1000 n3; h1 = 1000 n4; h2 = 1000 n5; h3 = 1000 n6; c1 = 1000 n7; c2 = 1000 n8;

Minimum stock should be 1, making sure that the amount is positive. We do not want any negative values. n1 >= 1; n2 >= 1; n3 >= 1; n4 >= 1; n5 >= 1; n6 >= 1; n7 >= 1; n8 >= 1;

int s1, s2, s3, h1, h2, h3, c1, c2;

adding integer restriction for stocks: int n1, n2, n3, n4, n5, n6, n7, n8;

Integer Restriction

```
integerrestriction <-read.lp("integerRestriction.lp")
solve(integerrestriction)
```

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

```
get.objective(integerrestriction)
```

```
## [1] 473050
```

```
get.variables(integerrestriction)
```

```
##  [1]  3000  4000  2000  2000  3000 12000 30000  4000     3     4     2     2
## [13]     3    12    30     4
```

Compare the solution in which there is no integer restriction on the number of shares invested.

Maximum Return : 473050 Total stocks owned in s1 = 3000 , Amount = 3000 * 40 = 120,000 $ Total stocks owned in s2 = 4000 , Amount = 4000 * 50 = 200,000 $
Total stocks owned in s3 = 2000 , Amount = 2000 * 80 = 160,000 $ Total stocks owned in h1 = 2000 , Amount = 2000 * 60 = 120,000 $ Total stocks owned in h2 = 3000 , Amount = 3000 * 45 = 135,000 $ Total stocks owned in h3 = 12000, Amount = 12000 * 60= 720,000 $ Total stocks owned in c1 = 30000, Amount = 30000 * 30= 900,000 $ Total stocks owned in c2 = 4000 , Amount = 4000 * 25 = 100,000 $

By how much (in percentage terms) do the integer restrictions alter the value of the optimal objective function?

```
             No Integer Restriction     Integer Restriction      The Change    Alter Percent
```

Optimum Value 477050 473050 -4000 -0.838%

By how much (in percentage terms) do they alter the optimal investment quantities?

```
No Integer Restriction       Integer Restriction       The Change       Alter Percent
```

S1 1,60,000 1,20,000 -40,000 -25% S2 2,00,000 2,00,000
S3 1,60,000 1,60,000
H1 1,20,000 1,20,000
H2 1,35,000 1,35,000
H3 7,20,000 7,20,000
C1 9,00,000 9,00,000
C2 1,00,000 1,00,000
——— ——— ——- ——

24,95,000 24,55,000 -40,000 -1.60%

Alter Percent Aggregate : -1.60%