Diet Problem

Μαρία Εσκιόγλου
3237
Σοφία Βαρταλάμη 3133

Περιεχόμενα

01

Μαθηματική Προσέγγιση

Τα περισσότερα πρακτικά προβλήματα μπορούν να μειωθούν σε προβλήματα μεγαλύτερης και μικρότερης σημασίας..και μόνο λύνοντας αυτά τα προβλήματα μπορούμε να ικανοποιήσουμε τις απαιτήσεις της πράξης η οποία πάντοτε ψάχνει το καλύτερο, το πιο βολικό.
P.L . Chebyshev (1821–1894)

Εισαγωγή και σκοπός εργασίας:

Ως βελτιστοποίηση ορίζουμε τη διαδικασία επίλυσης ενός προβλήματος με τον καλύτερο δυνατό τρόπο(**εύρεση μεγίστου, ελαχίστου**).

Μια κατηγορία προβλημάτων βελτιστοποίησης ανήκουν στο γραμμικό προγραμματισμό ή γραμμική βελτιστοποίηση, μέθοδος για την επίτευξη του καλύτερου αποτελέσματος (για παράδειγμα: μέγιστο κέρδος ή ελάχιστο κόστος) σε ένα μαθηματικό υπόδειγμα, του οποίου οι προϋποθέσεις (περιορισμοί) είναι ένα σύνολο γραμμικών σχέσεων των μεταβλητών του.

Αντικείμενο μελέτης μας είναι το Diet Problem, το οποίο χρονολογείται από τη δεκαετία του 1930 και του 1940 και παρουσιάζει γραμμικό (LP) μοντέλο, στόχος του οποίου είναι η **ελαχιστοποίηση του κόστους και οι περιορισμοί είναι η ικανοποίηση των καθορισμένων διατροφικών απαιτήσεων**.

Ανάλυση Προβλήματος:

Υποθέτουμε ότι το μενού μας αποτελείται από τα εξής τρόφιμα: μήλο(μερίδα: 1 αποφλοιωμένο μήλο), κοτόπουλο(μερίδα: 4 oz κοτόπουλο), καστανό ρύζι(μερίδα: 1 κούπα ρύζι), πατατάκια(μερίδα: 1 oz) και σπανάκι(μερίδα: 1 κούπα σπανάκι).

Αναλυτικός Πίνακας για τη θρεπτική αξία κάθε τροφίμου:

food	calories	fat	sodium	vitamin C	vitamin A	protein
apple	65	0	0	5.7	20.25	0.3
chicken	328	4	512	32	0	17.6
brown rice	216	0	10	0	0	5
chips	137	2	210	9.5	20.27	2.2
spinach	6.9	0	42	8.4	843.9	1

Στόχος του διατροφικού προβλήματος αποτελεί η επιλογή ενός συγκεκριμένου αριθμού μερίδων κάθε τροφής για αγορά και κατανάλωση με κριτήριο την ελαχιστοποίηση του κόστους αγοράς ενώ ταυτόχρονα θα ικανοποιούνται οι καθορισμένες διατροφικές απαιτήσεις.

θα ασχοληθούμε με τους εξής περιορισμούς : την ποσότητα των θερμίδων, την περιεκτικότητα σε λιπαρά, νάτριο ,πρωτεΐνη, βιταμίνη A και βιταμίνη C κάθε μερίδας για κάθε τρόφιμο που θα εμπεριέχεται στο παραπάνω καθορισμένο μενού.

Μαθηματική Υλοποίηση:

Έστω F το πλήθος των τροφίμων και N ο αριθμός των θρεπτικών ουσιών.
Στη συνέχεια ορίζουμε τις παραμέτρους:
Aij : ποσότητα θρεπτικής ουσίας j στο τρόφιμα i, $\forall i \in F$ και $\forall j \in N$
Ci: κόστος κάθε μερίδας φαγητού i, $\forall i \in F$
Nminj: ελάχιστο απαιτούμενο επίπεδο θρεπτικών συστατικών j, $\forall j \in N$
Nmaxj: μέγιστο επιτρεπόμενο επίπεδο θρεπτικών συστατικών j, $\forall j \in N$

Η μεταβλητή xi δείχνει τον αριθμό των μερίδων φαγητού για αγορά και κατανάλωση

- ➔ Ελαχιστοποίηση του συνολικού κόστους φαγητού: minimize $\sum Cixi$

Περιορισμοί:

- ➢ Για κάθε θρεπτικό $j \in N$, τουλάχιστον να ικανοποιεί το ελάχιστο απαιτούμενο επίπεδο:
  $\sum Aijxi \geq Nminj \ , \forall jN$
- ➢ Για κάθε θρεπτικό $j \in N$, να μην υπερβαίνει το μέγιστο επιτρεπόμενο επίπεδο:
  $\sum Aijxi \leq Nmaxj \ , \forall jN$

Μαθηματική Υλοποίηση:

Έστω ότι έχουμε τις τροφές:

x1 (apple) with C1=1.5

x2 (chicken) with C2=2.75

x3 (brown ice) with C3=0.5

x4 (chips) with C4=1

x5 (spinach) with C5=2

- **Αντικειμενική Συνάρτηση**

Ας ορίσουμε $f(x1,x2,x3,x4,x5)=1.5x1+2.75x2+0.5x3+x4+2x5$

- **Περιορισμοί**

Για τις θερμίδες έχουμε: $65x1+328x2+216x3+137x4+6.9x5= 2000$

$g1(x1,x2,x3,x4,x5)=65x1+328x2+216x3+137x4+6.9x5-2000 =0$

Για τα λιπαρά: $4x2+2x4\leq 10$

$g2(x1,x2,x3,x4,x5)=-4x2-2x4+10\geq 0$

\

Για το νάτριο: $512x2+10x3+210x4+24x5\leq 2200$

$g3(x1,x2,x3,x4,x5)=-512x2-10x3-210x4-24x5+2200\geq 0$

Για τη βιταμίνη C: $5.7x1+32x2+9.5x4+8.4x5\geq 100$

$g4(x1,x2,x3,x4,x5)=5.7x1+32x2+9.5x4+8.4x5-100\geq 0$

Για τη βιταμίνη A: $20.25x1+20.27x4+843.9x5\geq 700$

$g5(x1,x2,x3,x4,x5)=20.25x1+20.27x4+843.9x5-700\geq 0$

Για την πρωτεΐνη: $0.3x1+17.6x2+5x3+2.2x4+x5\geq 50$

$g6(x1,x2,x3,x4,x5)=0.3x1+17.6x2+5x3+2.2x4+x5-50\geq 0$

με x1,x2,x3,x4,x51

Συνάρτηση Lagrange:

Εφαρμόζουμε τη **συνθήκη μέθοδο πολλαπλασιαστή Lagrange**

$L(x1,x2,x3,x4,x5,\lambda1,\lambda2,\lambda3,\lambda4,\lambda5,\lambda6)=1.5x1+2.75x2+0.5x3+x4+2x5-$
$\lambda1(65x1+328x2+216x3+137x4+6.9x5-2000)-\lambda2(-4x2-2x4+10)-\lambda3(-512x2-10x3-210x4-24x5+2200)-$
$\lambda4(5.7x1+32x2+9.5x4+8.4x5-100)-\lambda5(20.25x1+20.27x4+843.9x5-700)-$
$\lambda6(0.3x1+17.6x2+5x3+2.2x4+x5-50)$

Η L είναι παραγωγίσιμη:

$Lx1=1.5-65\lambda1-5.7\lambda4-20.25\lambda5-0.3\lambda6=0$

$Lx2=2.75-328\lambda1+4\lambda2+512\lambda3-32\lambda4-17.6\lambda6=0$

$Lx3=0.5-216\lambda1+10\lambda3-5\lambda6=0$

$Lx4=1-137\lambda1+2\lambda2+210\lambda3-9.5\lambda4-20.27\lambda5-2.2\lambda6=0$

$Lx5=2-6.9\lambda1+24\lambda3-8.4\lambda4-843.9\lambda5-\lambda6=0$

KKT conditions:

$1.5-65\lambda1-5.7\lambda4-20.25\lambda5-0.3\lambda6=0$

$2.75-328\lambda1+4\lambda2+512\lambda3-32\lambda4-17.6\lambda6=0$

$0.5-216\lambda1+10\lambda3-5\lambda6=0$

$1-137\lambda1+2\lambda2+210\lambda3-9.5\lambda4-20.27\lambda5-2.2\lambda6=0$

$2-6.9\lambda1+24\lambda3-8.4\lambda4-843.9\lambda5-\lambda6=0$

$65x1+328x2+216x3+137x4+6.9x5-2000 =0$

$-4x2-2x4+10\geq0$

$-512x2-10x3-210x4-24x5+2200\geq0$

$5.7x1+32x2+9.5x4+8.4x5-100\geq0$

$20.25x1+20.27x4+843.9x5-700\geq0$

$0.3x1+17.6x2+5x3+2.2x4+x5-50\geq0$

με $x1,x2,x3,x4,x5\geq1$

$\lambda1,\lambda2,\lambda3,\lambda4,\lambda5,\lambda6\geq0$

$\lambda1(65x1+328x2+216x3+137x4+6.9x5-2000)=0$

$\lambda2(-4x2-2x4+10)=0$

$\lambda3(-512x2-10x3-210x4-24x5+2200)=0$

$\lambda4(5.7x1+32x2+9.5x4+8.4x5-100)=0$

$\lambda5(20.25x1+20.27x4+843.9x5-700)=0$

$\lambda6(0.3x1+17.6x2+5x3+2.2x4+x5-50)=0$

Σημείωση: λύση του παραπάνω συστήματος μπορεί να γίνει με τη μέθοδος Gauss Jordan.

Η λύση μας, **x***, είναι αυτή που πληροί τις συνθήκες KKT.

Για την τιμή **x*** της f που βρήκαμε κατασκευάζουμε τον εσσιανό πίνακα

∇^2L(x1,x2,x3,x4,x5,λ1,λ2,λ3,λ4,λ5,λ6)

Αν |∇^2L(x1,x2,x3,x4,x5,λ1,λ2,λ3,λ4,λ5,λ6)|>0 (θετικά ορισμένος στον εφαπτόμενο κώνο) τότε η f έχει στο σημείο **x*** δεσμευμένο τοπικό ελάχιστο.

❖ Simplex:

minimize $z=1.5x1+2.75x2+0.5x3+x4+2x5+MA1+MA2+MA3+MA4$

subject to $65x1+328x2+216x3+137x4+6.9x5+A1= 2000$

$4x2+2x4+S1=10$

$512x2+10x3+210x4+24x5+S2=2200$

$5.7x1+32x2+9.5x4+8.4x5-S3+A2=100$

$20.25x1+20.27x4+843.9x5-S4+A3=700$

$0.3x1+17.6x2+5x3+2.2x4+x5-S5+A4=50$

Cj		1.5	2.75	0.5	1	2	0	0	0	0	0	M	M	M	M		
CB	basic variables	x1	x2	x3	x4	x5	S1	S2	S3	S4	S5	A1	A2	A3	A4	Solution	Ratio
M	A1	65	328	216	137	6.9	0	0	0	0	0	1	0	0	0	2000	6.09
0	S1	0	4	0	2	0	1	0	0	0	0	0	0	0	0	10	2.5
0	S2	0	512	10	210	24	0	1	0	0	0	0	0	0	0	2200	4.29
M	A2	5.7	32	0	9.5	8.4	0	0	-1	0	0	0	1	0	0	100	3.12
M	A3	20.25	0	0	20.27	843.9	0	0	0	-1	0	0	0	1	0	700	
M	A4	0.3	17.6	5	2.2	1	0	0	0	0	-1	0	0	0	1	50	2.84
Zj		91.52M	377.6M	221M	168.97M	860.2M	0	0	-M	-M	-M	M	M	M	M		
Cj-Zj		1.5-91.52M	2.75-377.6M	0.5-221M	1-168.97M	2-860.2M	0	0	M	M	M	0	0	0	0		

Αν το Cj-Zj ≥0 η λύση είναι βέλτιστη ☺αλλιώς

Επιλέγουμε το ελάχιστο Cj-Zj, στην περίπτωση μας το 2.75-377.6M και εστιάζουμε σε εκείνη τη στήλη.

Έπειτα υπολογίζουμε το RATIO : αποτέλεσμα της διαίρεσης της λύσης στη γραμμή i διά του του αντίστοιχου στοιχείου στην στήλη j που επιλέξαμε και στην γραμμή που βρισκόμαστε i και επιλέγουμε το μικρότερο από αυτά επικεντρώνοντας σε αυτή τη γραμμή.

Στην περίπτωση μας , αυτό σημαίνει ότι εστιάζουμε στην πορτοκαλί γραμμή και έτσι φεύγει η μεταβλητή S1.

Το κόκκινο τετραγωνάκι ονομάζεται pivot και η μεταβλητή που θα εισάγουμε θα είναι η x2 με βάση τη στήλη που ήμασταν.

Έπειτα ξαναδημιουργείται ένας πίνακα μόνο που τώρα αντί για S1 θα έχουμε την x2.

Στη γραμμή του x2 το 0 θα γίνει 2.75 και στη συνέχεια τα νέα στοιχεία θα είναι τα στοιχεία της προηγούμενης διά το pivot ενώ το RATIO θα ναι κενό.

Για τα υπόλοιπα δεδομένα θα τροποποιηθούν με βάση την παρακάτω φόρμουλα:

New Value= Old Value-(element on the line of pivot*element on the column of the pivot)/pivot

Μόλις συμπληρώσουμε τα στοιχεία αυτά, υπολογίζουμε τα Zj, Cj-Zj .

Εάν το **Cj-Zj είναι ≥0** τότε σταματάμε τη διαδικασία-αλγόριθμο και έχουμε βρει βέλτιστη λύση με τα x1,x2,x3,x4,x5 να έχουν ως τιμές αυτές που αντιστοιχούν στη στήλη Solution αλλιώς επαναλαμβάνουμε τη διαδικασία.

02

Υλοποίηση

PuLP

Μία βιβλιοθήκη για Γραμμικό Προγραμματισμό.

```
$ sudo pip install pulp              # PuLP
$ sudo apt-get install glpk-utils   # GLPK
$ sudo apt-get install coinor-cbc   # CoinOR
```

Περιγραφή Υλοποίησης (I)

```
1.   Δημιουργία Προβλήματος
prob = LpProblem("Minimizing Cost",LpMinimize)

2.   Δήλωση Μεταβλητών
choice1 = pulp.LpVariable("Food",lowBound=1,cat='Continuous')
...

3.   Δήλωση Θρεπτικών Συστατικών
choices = [choice1, choice2, choice3, choice4, choice5]
calories = [65, 328, 216, 137, 6.9]
fat = [0, 4, 0, 2, 0]
sodium = [0, 512, 10, 210, 24]
vitC = [5.7, 32, 0, 9.5, 8.4]
vitA = [20.25, 0, 0, 20.27, 843.9]
protein = [0.3, 17.6, 5, 2.2, 1]
```

Έστω:

choice1= apples
choice2= chicken
choice3= rice
choice4= chips
choice5= spinach

Περιγραφή Υλοποίησης (2)

4. **Objective Function**

```
prob += 1.50 * choice1 + 2.75*choice2 + 0.5*choice3 + 1*choice4 + 2*choice5
```

5. **Δήλωση Περιορισμών**

```
prob += pulp.lpSum([calories[i]*choices[i] for i in range(len(choices))]) == 2000
prob += pulp.lpSum([fat[i]*choices[i] for i in range(len(choices))]) <= 10
prob += pulp.lpSum([sodium[i]*choices[i] for i in range(len(choices))]) <= 2200
prob += pulp.lpSum([vitC[i]*choices[i] for i in range(len(choices))]) >= 100
prob += pulp.lpSum([vitA[i]*choices[i] for i in range(len(choices))]) >= 700
prob += pulp.lpSum([protein[i]*choices[i] for i in range(len(choices))]) >= 50
```

6. ***Επίλυση***

```
prob.solve()
```

Έξοδος Προγράμματος

```
Servings of apples = 1.0
Servings of chicken = 1.8625
Servings of brown rice = 5.4
Servings of chips = 1.0
Servings of spinach = 3.0
```

Μερίδες

```
Total Calories: 2000/2000
Total Fat: 9/10
Total Sodium: 1290/2200
Total Vitamin C: 100/100
Total Vitamin A: 2572/700
Total Protein: 65/50
```

Σύνολο Θρεπτικών Συστατικών

```
The total cost is $16.32187 for us to get the required daily nutrients
```

Ευχαριστούμε πολύ!