pianificazione-di-investimenti-ver-2

July 12, 2021

1 pianificazione di investimenti ver 2

Si vogliono realizzare n progetti nei prossimi T anni. Di ogni progetto i si conosce un indice di redditività p_i che esprime il guadagno finale atteso (in Euro) e un profilo di costo $a_i = (a_{i1}, a_{i2}, \ldots, a_{iT})$ per ogni anno del periodo considerato.

Inoltre, in ogni anno j del periodo si dispone di un budget di b_j €.

Quali progetti occorre selezionare per massimizzare il guadagno atteso rispettando i vincoli di budget?

		costi a (K€)			
	redditività p (K€)	anno 1	anno 2	anno 3	anno 4
progetto 1	30	10	5	-2	-1
progetto 2	20	12	2	-2	-5
progetto 3	25	15	-1	5	5
budget		30	6	6	6

1.1 dati del problema

[1]:
```
from pyomo.environ import *
from itertools import islice

def skip_first_dict(d):
    return dict(islice(d.items(), 1, len(d)))

def init():
    i = [ 'pr1', 'pr2', 'pr3' ]
    P = [   5.0,   3.0,  4.0  ]

    j = { 1: 'an1', 2: 'an2', 3: 'an3', 4: 'an4' }
    b = [      5.0,      7.0,      3.0,     4.0  ]

    c = [[3,  5,  2,   1],
         [2,   2,  2,  4],
         [5,  3,  5,   5]]
    return i, P, j, b, c
```

1.2 tipo 1

Ogni nuovo progetto i comporta un costo globale di gestione c_i.

Si vuole massimizzare il ricavo, cioè la differenza tra il guadagno atteso e i costi di gestione.

$$Z = \max\left(\sum_j \left(\sum_i \left(P_i\ Pr_j - a_{ij}\ Pr_j\right)\right)\right)$$

$$C1 : \sum_i \left(a_{ij} Pr_j\right) \leq b_j\ \forall j$$

$$C2 : \sum_i pr_{ij} \geq 1\ \ \forall j$$

[2]:
```
model = ConcreteModel()
model.name = 'tipo 1'
i, P, j, b, c = init()

model.i = Set(initialize=i)
model.j = Set(initialize=j.values())

c_dict = {}
for i, mi in enumerate(model.i):
    for j, mj in enumerate(model.j):
        c_dict[mi, mj] = c[i][j]
P = {mi: P[i] for i, mi in enumerate(model.i)}
b = {mj: b[j] for j, mj in enumerate(model.j)}

model.P = Param(model.i, initialize=P)
model.b = Param(model.j, initialize=b)
model.c = Param(model.i, model.j, initialize=c_dict)
model.Pr = Var(model.i, model.j, domain=Boolean, initialize=0)

obj_expr = sum(sum(model.P[i]*model.Pr[i, j] - model.c[i, j]*model.Pr[i, j] for␣
 ↪i in model.i) for j in model.j)

model.ricavo = Objective(expr = obj_expr, sense=maximize)

model.constraints = ConstraintList()
for j in model.j:
    model.constraints.add(expr = sum(model.c[i, j]*model.Pr[i, j] for i in model.
 ↪i) <= model.b[j])
for j in model.j:
    model.constraints.add(expr = sum(model.Pr[i, j] for i in model.i) >= 1)

results = SolverFactory('glpk').solve(model)
model.display()
```

```
Model tipo 1

  Variables:
    Pr : Size=12, Index=Pr_index
        Key            : Lower : Value : Upper : Fixed : Stale : Domain
        ('pr1', 'an1') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr1', 'an2') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr1', 'an3') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr1', 'an4') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an1') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an2') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an3') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr2', 'an4') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr3', 'an1') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr3', 'an2') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr3', 'an3') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr3', 'an4') :     0 :   0.0 :     1 : False : False : Boolean

  Objectives:
    ricavo : Size=1, Index=None, Active=True
        Key  : Active : Value
        None :   True :  12.0

  Constraints:
    constraints : Size=8
        Key : Lower : Body : Upper
          1 :  None :  5.0 :   5.0
          2 :  None :  5.0 :   7.0
          3 :  None :  2.0 :   3.0
          4 :  None :  1.0 :   4.0
          5 :   1.0 :  2.0 :  None
          6 :   1.0 :  2.0 :  None
          7 :   1.0 :  1.0 :  None
          8 :   1.0 :  1.0 :  None
```

1.3 Tipo 2

Il budget disponibile in ogni anno j è pari ad una quota fissa bj sommata al budget residuo dei periodi precedenti.

$$Z = \max\left(\sum_j \left(\sum_i \left(P_i\ Pr_j - a_{ij}\ Pr_j\right)\right)\right)$$

$$C1 : \sum_i \left(a_{ij} Pr_j\right) \leq bg_j \ \ \forall j$$

$$C2 : bg_0 = b_0$$

$$C3 : bg_j = \sum_i \left(P_i\ Pr_{j-1} - a_{ij-1}\ Pr_{j-1}\right) + b_j \ \ \forall j \in (2...T)$$

$$C4 : \sum_i pr_{ij} \geq 1 \ \ \forall j$$

[3]:
```
model = ConcreteModel()
model.name = 'tipo 2'

i, P, j, b, c = init()

model.i = Set(initialize=i)
model.j = Set(initialize=j.values())

c_dict = {}
for il, mi in enumerate(model.i):
    for jl, mj in enumerate(model.j):
        c_dict[mi, mj] = c[il][jl]

P = {mi: P[il] for il, mi in enumerate(model.i)}
b = {mj: b[jl] for jl, mj in enumerate(model.j)}

model.P = Param(model.i, initialize=P)
model.b = Param(model.j, initialize=b)
model.c = Param(model.i, model.j, initialize=c_dict)

model.Pr = Var(model.i, model.j, domain=Boolean, initialize=0)
model.bg = Var(model.j, domain=Reals, initialize=0)
model.bg[j[1]] = model.b[j[1]]

obj_expr = sum(sum(model.P[i]*model.Pr[i, j] - model.c[i, j]*model.Pr[i, j] for␣
 ↪i in model.i) for j in model.j)

model.ricavo = Objective(expr = obj_expr, sense=maximize)

model.constraints = ConstraintList()
for mj in model.j:
    model.constraints.add(expr = sum(model.c[i, mj]*model.Pr[i, mj] for i in␣
 ↪model.i) <= model.bg[mj])
for mj in model.j:
    model.constraints.add(expr = sum(model.Pr[i, mj] for i in model.i) >= 1)
for mj in skip_first_dict(j):
    model.constraints.add(expr =
                model.bg[j[mj]] == sum(model.P[i]*model.Pr[i, j[mj-1]]
                                       - model.c[i, j[mj-1]]*model.Pr[i, j[mj-1]]
                                          for i in model.i)
                                       + model.b[j[mj]])

results = SolverFactory('glpk').solve(model)

model.display()
```

```
Model tipo 2

  Variables:
    Pr : Size=12, Index=Pr_index
        Key            : Lower : Value : Upper : Fixed : Stale : Domain
        ('pr1', 'an1') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr1', 'an2') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr1', 'an3') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr1', 'an4') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an1') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an2') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an3') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr2', 'an4') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr3', 'an1') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr3', 'an2') :     0 :   1.0 :     1 : False : False : Boolean
        ('pr3', 'an3') :     0 :   0.0 :     1 : False : False : Boolean
        ('pr3', 'an4') :     0 :   0.0 :     1 : False : False : Boolean
    bg : Size=4, Index=j
        Key : Lower : Value : Upper : Fixed : Stale : Domain
        an1 :  None :   5.0 :  None : False : False :  Reals
        an2 :  None :  10.0 :  None : False : False :  Reals
        an3 :  None :   5.0 :  None : False : False :  Reals
        an4 :  None :   8.0 :  None : False : False :  Reals

  Objectives:
    ricavo : Size=1, Index=None, Active=True
        Key  : Active : Value
        None :   True :  13.0

  Constraints:
    constraints : Size=11
        Key : Lower : Body : Upper
          1 :  None :  0.0 :   0.0
          2 :  None : -5.0 :   0.0
          3 :  None : -1.0 :   0.0
          4 :  None : -7.0 :   0.0
          5 :   1.0 :  2.0 :  None
          6 :   1.0 :  2.0 :  None
          7 :   1.0 :  2.0 :  None
          8 :   1.0 :  1.0 :  None
          9 :   0.0 :  0.0 :   0.0
         10 :   0.0 :  0.0 :   0.0
         11 :   0.0 :  0.0 :   0.0
```

[ ]: