240_CaseStudy2

November 6, 2023

[1]:
```
pip install gurobipy
```

```
Requirement already satisfied: gurobipy in /srv/conda/lib/python3.9/site-
packages (10.0.3)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
import gurobipy as gp
from gurobipy import GRB
```

[3]:
```
m = gp.Model("CaseStudy2")

# Create Variables
x1 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x1") # Grand␣
 ↪Estates-The Trump-On the lake
x2 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x2") # Grand␣
 ↪Estates-The Trump-Not on the lake
x3 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x3") # Grand␣
 ↪Estates-The Vanderbilt-On the lake
x4 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x4") # Grand␣
 ↪Estates-The Vanderbilt-Not on the lake
x5 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x5") # Grand␣
 ↪Estates-The Hughes-On the lake
x6 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x6") # Grand␣
 ↪Estates-The Hughes-Not on the lake
x7 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x7") # Grand␣
 ↪Estates-The Jackson-On the lake
x8 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x8") # Grand␣
 ↪Estates-The Jackson-Not on the lake

x9 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x9") # Glen Wood␣
 ↪Collection-Grand Cypress-On Premium
x10 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x10") # Glen Wood␣
 ↪Collection-Grand Cypress-Standarded
x11 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x11") # Glen Wood␣
 ↪Collection-Lazy Oak
x12 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x12") # Glen Wood␣
 ↪Collection-Wind Row
```

```
x13 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x13") # Glen Wood␣
 ↪Collection-Orangewood

x14 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x14") # Lakeview Patio␣
 ↪Homes-Bayview-On Premium
x15 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x15") # Lakeview Patio␣
 ↪Homes-Bayview-Standarded
x16 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x16") # Lakeview Patio␣
 ↪Homes-Storeline
x17 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x17") # Lakeview Patio␣
 ↪Homes-Docks Edge
x18 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x18") # Lakeview Patio␣
 ↪Homes-Golden Pier

x19 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x19") # Country␣
 ↪Condominiums-Country Stream
x20 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x20") # Country␣
 ↪Condominiums-Weeping Willow
x21 = m.addVar(lb = 0.0, vtype = GRB.CONTINUOUS, name = "x21") # Country␣
 ↪Condominiums-Picket Fence
```

```
Restricted license - for non-production use only - expires 2024-10-28
```

0.1 Selling price for exclusive home will be an additional 30% plus 50,000 more than the models not on the lake

x1 700,000*1.3+50,000 = $960,000

x2 $700,000

x3 $934,000

x4 $680,000

x5 $895,000

x6 $650,000

x7 $817,000

x8 $590,000

0.2 Built on "premium" will have higher selling price

x9 420,000+40,000 = $460,000

x10 $420,000

x11 $380,000

x12 $320,000

x13 $280,000

x14 300,000+30,000 = $330,000

x15 $300,000

x16 $270,000

x17 $240,000

x18 $200,000

x19 $220,000

x20 $160,000

x21 $140,000

[4]:
```
# Set Objective
m.setObjective(0.
 ↪22*(960000*x1+700000*x2+934000*x3+680000*x4+895000*x5+650000*x6+817000*x7+590000*x8)
              +0.18*(460000*x9+420000*x10+380000*x11+320000*x12+280000*x13)
              +0.2*(330000*x14+300000*x15+270000*x16+240000*x17+200000*x18)
              +0.25*(220000*x19+160000*x20+140000*x21), GRB.MAXIMIZE)
```

[5]:
```
# Set Constrains
# Each of the Grand Estate series plans must have at least eight units on the␣
 ↪lake
m.addConstr (x1>=8, "c0")
m.addConstr (x3>=8, "c1")
m.addConstr (x5>=8, "c2")
m.addConstr (x7>=8, "c3")

# 50 half-acre lots on the lake are to be used exclusively by the Grand Estate␣
 ↪Series homes
m.addConstr (x1+x3+x5+x7==50, "c4")

# No more than 25% of the total Grand Cypress models and 25% of the total␣
 ↪Bayview models may be built on the premium lots
m.addConstr (x9<=0.25*(x9+x10), "c5")
m.addConstr (x14<=0.25*(x14+x15), "c6")

#Each outside parking space will occupy 200 square feet of space
m.addConstr ((2*(x1+x2+x11+x14+x15+x20+x21) + 3*x19 +␣
 ↪(x3+x4+x5+x6+x9+x10+x12+x13+x16+x17))*200<=15*43560, "c7")
```

[5]:
```
<gurobi.Constr *Awaiting Model Update*>
```

[6]:
```
# Variety
m.addConstr((x18+x20+x21)<=0.
 ↪25*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c8")
m.addConstr((x18+x20+x21)>=0.
 ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c9")
```

```
m.addConstr((x7+x8+x12+x13+x16+x17+x19)>=0.
  ↪25*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c10")
m.addConstr((x7+x8+x12+x13+x16+x17+x19)<=0.
  ↪4*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c11")

m.addConstr((x3+x4+x5+x6+x9+x10+x11+x14+x15)>=0.
  ↪25*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c12")
m.addConstr((x3+x4+x5+x6+x9+x10+x11+x14+x15)<=0.
  ↪4*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c13")

m.addConstr((x1+x2)<=0.
  ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c14")
m.addConstr((x1+x2)>=0.
  ↪05*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c15")

# None of the four products is to make up more than 35% or less than 15% of the␣
  ↪units built in the development
m.addConstr((x1+x2+x3+x4+x5+x6+x7+x8)<=0.
  ↪35*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c16")
m.addConstr((x1+x2+x3+x4+x5+x6+x7+x8)>=0.
  ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c17")

m.addConstr((x9+x10+x11+x12+x13)<=0.
  ↪35*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c18")
m.addConstr((x9+x10+x11+x12+x13)>=0.
  ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c19")

m.addConstr((x14+x15+x16+x17+x18)<=0.
  ↪35*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c20")
m.addConstr((x14+x15+x16+x17+x18)>=0.
  ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c21")

m.addConstr((x19+x20+x21)>=0.
  ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c22")
m.addConstr((x19+x20+x21)<=0.
  ↪35*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c23")

# Each plan must occupy between 20% and 35% of the total units of that products
m.addConstr((x1+x2)<=0.35*(x1+x2+x3+x4+x5+x6+x7+x8),"c24")
m.addConstr((x1+x2)>=0.2*(x1+x2+x3+x4+x5+x6+x7+x8),"c25")
m.addConstr((x3+x4)<=0.35*(x1+x2+x3+x4+x5+x6+x7+x8),"c26")
m.addConstr((x3+x4)>=0.2*(x1+x2+x3+x4+x5+x6+x7+x8),"c27")
m.addConstr((x5+x6)<=0.35*(x1+x2+x3+x4+x5+x6+x7+x8),"c28")
m.addConstr((x5+x6)>=0.2*(x1+x2+x3+x4+x5+x6+x7+x8),"c29")
m.addConstr((x7+x8)<=0.35*(x1+x2+x3+x4+x5+x6+x7+x8),"c30")
```

```
m.addConstr((x7+x8)>=0.2*(x1+x2+x3+x4+x5+x6+x7+x8),"c31")

m.addConstr((x9+x10)<=0.35*(x9+x10+x11+x12+x13),"c32")
m.addConstr((x9+x10)>=0.2*(x9+x10+x11+x12+x13),"c33")
m.addConstr(x11<=0.35*(x9+x10+x11+x12+x13),"c34")
m.addConstr(x11>=0.2*(x9+x10+x11+x12+x13),"c35")
m.addConstr(x12<=0.35*(x9+x10+x11+x12+x13),"c36")
m.addConstr(x12>=0.2*(x9+x10+x11+x12+x13),"c37")
m.addConstr(x13<=0.35*(x9+x10+x11+x12+x13),"c38")
m.addConstr(x13>=0.2*(x9+x10+x11+x12+x13),"c39")

m.addConstr((x14+x15)<=0.35*(x14+x15+x16+x17+x18),"c40")
m.addConstr((x14+x15)>=0.2*(x14+x15+x16+x17+x18),"c41")
m.addConstr(x16<=0.35*(x14+x15+x16+x17+x18),"c42")
m.addConstr(x16>=0.2*(x14+x15+x16+x17+x18),"c43")
m.addConstr(x17<=0.35*(x14+x15+x16+x17+x18),"c44")
m.addConstr(x17>=0.2*(x14+x15+x16+x17+x18),"c45")
m.addConstr(x18<=0.35*(x14+x15+x16+x17+x18),"c46")
m.addConstr(x18>=0.2*(x14+x15+x16+x17+x18),"c47")

m.addConstr(x19<=0.35*(x19+x20+x21),"c48")
m.addConstr(x19>=0.2*(x19+x20+x21),"c49")
m.addConstr(x20<=0.35*(x19+x20+x21),"c50")
m.addConstr(x20>=0.2*(x19+x20+x21),"c51")
m.addConstr(x21<=0.35*(x19+x20+x21),"c52")
m.addConstr(x21>=0.2*(x19+x20+x21),"c53")

# No more than 70% of the single-family homes (all homes except the Country␣
 ↪Condominiums) may be two-story homes
m.addConstr((x1+x2+x3+x4+x9+x10+x11+x12+x14+x15+x16)<=0.
 ↪7*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18),"c54")

# Affordable Housing
m.addConstr((x18+x20+x21)>=0.
 ↪15*(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21),"c55")
```

[6]:
```
<gurobi.Constr *Awaiting Model Update*>
```

[7]:
```
# Lotsize + Outside Parking + Road/Greenbelts <= 300 Acres
# Lotsize = Sum up half,1/4,1/6,1/10, and all others
# Lotsize = Ground + Yard + Garage
# Outside Parking = (2*(x1+x2+x11+x14+x15+x20+x21) + 3*x19 +␣
 ↪(x3+x4+x5+x6+x9+x10+x12+x13+x16+x17))*200
# Road/Greenbelts =␣
 ↪(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21)*1000
m.addConstr(0.5*43560*(x1+x2+x3+x4+x5+x6+x7+x8)
            +0.25*43560*x9
```

```
            +4950*x10
            +0.1*43560*(x11+x12+x13+x15+x16+x17+x18)
            +(43560/6)*x14
            +1500*(x19+x20+x21)
            +((2*(x1+x2+x11+x14+x15+x20+x21) + 3*x19 +␣
 ↪(x3+x4+x5+x6+x9+x10+x12+x13+x16+x17))*200)
          ␣
 ↪+((x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12+x13+x14+x15+x16+x17+x18+x19+x20+x21)*1000)<=(300*
```

[7]:
```
<gurobi.Constr *Awaiting Model Update*>
```

[8]:
```
m.optimize()

for v in m.getVars():
    print('%s %g' % (v.VarName, v.X))

print ('obj:%g' % m.ObjVal)
```

```
Gurobi Optimizer version 10.0.3 build v10.0.3rc0 (linux64)

CPU model: Intel(R) Xeon(R) CPU @ 2.80GHz, instruction set
[SSE2|AVX|AVX2|AVX512]
Thread count: 4 physical cores, 8 logical processors, using up to 8 threads

Optimize a model with 57 rows, 21 columns and 588 nonzeros
Model fingerprint: 0x4f86c312
Coefficient statistics:
  Matrix range     [5e-02, 2e+04]
  Objective range  [4e+04, 2e+05]
  Bounds range     [0e+00, 0e+00]
  RHS range        [8e+00, 1e+07]
Presolve removed 5 rows and 0 columns
Presolve time: 0.01s
Presolved: 52 rows, 21 columns, 563 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.4888752e+09   1.095723e+06   0.000000e+00      0s
      25    1.2260933e+08   0.000000e+00   0.000000e+00      0s

Solved in 25 iterations and 0.01 seconds (0.00 work units)
Optimal objective  1.226093338e+08
x1 26
x2 66.62
x3 8
x4 58.1571
x5 8
x6 44.9257
```

```
x7 8
x8 44.9257
x9 0
x10 216.113
x11 154.367
x12 123.493
x13 123.493
x14 0
x15 102.911
x16 67.6273
x17 64.687
x18 58.8063
x19 205.822
x20 205.822
x21 176.419
obj:1.22609e+08
```

[ ]:

[ ]:

[ ]:

[ ]: