Оптимизация в ценообразовании

Стажировка по математической оптимизации
GlowByte

Белый Олег

Постановка задачи №1

Имеется некоторая модель прогноза продаж, необходимо оптимизировать ожидаемую выручку, описываемую следующей формулой:

$$F(\boldsymbol{R}) = \sum_{s=1}^{S}\sum_{p=1}^{P}\sum_{w=1}^{W} r_{pw} * \left(M_{sp} * \frac{-\left(\frac{r_{pw}}{r_{pw}^{seasonal}}\right)^2 - 0.075 * \frac{r_{pw}}{r_{pw}^{seasonal}} + 2.85}{1.775} + L_{spw}\right)$$

где
S – число магазинов
s – индекс магазина
P – число продуктов
p – индекс продукта
W – число недель
w – индекс недели
M_{sp} – среднее число продаж продукта p в магазине s
r_{pw} – цена продукта p в неделю w
$r_{pw}^{seasonal}$ – сезонная ожидаемая цена продукта p в неделю w
L_{spw} – добавочное количество продаж к продукту p в неделю w в магазине s
$\boldsymbol{R}$ – матрица цен r_{pw}

$$L_{spw} = \sum_{o=1}^{P} \left(\frac{r_{ow}^{seasonal}}{r_{ow}}\right)^2 * C_{po} * M_{so}$$

где
P – число продуктов
o – индекс связанного продукта
C_{po} – коэффициент взаимосвязи продукта p и связанного продукта o
M_{so} – среднее число продаж продукта o в магазрине s

Выбор инструментария

- Данная задача относится к классу нелинейных, а именно, является задачей квадратичного программирования
- В качестве решателя был выбран солвер IPOPT, использующий прямо-двойственный метод внутренней точки для решения нелинейных задач
- Фреймворк – Pyomo

Получение данных

	apple	banana	pear	orange	lemon
0	825.0	309.0	416.0	522.0	571.0
1	618.0	358.0	459.0	460.0	657.0
2	2144.0	1020.0	1147.0	1660.0	1599.0
3	727.0	349.0	418.0	556.0	437.0
4	667.0	272.0	409.0	556.0	634.0

Средние цены продуктов по магазинам

	apple	banana	pear	orange	lemon
apple	0.00	0.15	-0.15	0.17	0.17
banana	0.15	0.00	0.12	0.10	0.10
pear	-0.15	0.12	0.00	0.12	0.12
orange	0.17	0.10	0.12	0.00	-0.19
lemon	0.17	0.10	0.12	-0.19	0.00

Коэффициенты взаимосвязи продуктов

	price_apple	price_banana	price_pear	price_orange	price_lemon
0	90.0	80.0	130.0	100.0	110.0
1	90.0	80.0	130.0	100.0	110.0
2	90.0	80.0	130.0	100.0	110.0
3	90.0	80.0	130.0	100.0	110.0
4	90.0	80.0	130.0	100.0	110.0
5	80.0	80.0	120.0	95.0	105.0
6	80.0	80.0	120.0	95.0	105.0
7	80.0	80.0	120.0	95.0	105.0
8	80.0	80.0	120.0	95.0	105.0
9	80.0	80.0	120.0	90.0	100.0
10	80.0	80.0	120.0	90.0	100.0
11	80.0	80.0	120.0	90.0	100.0
12	80.0	80.0	120.0	90.0	100.0
13	85.0	80.0	125.0	85.0	95.0
14	85.0	80.0	125.0	85.0	95.0
15	85.0	80.0	125.0	85.0	95.0

Сезонные цены продуктов по неделям

Построение оптимизационной модели

Инициализация вспомогательных переменных

```
#число магазинов
S = df1.shape[0]
#число продуктов
P = df2.shape[0]
#число недель
W = df3.shape[0]
#матрица продаж продуктов в магазинах
M = df1.values
#матрица связанных продуктов
C = df2.values
#матрица сезонных ожидаемых цен
#(транспонирую, чтобы недели стали столбцами,
r_seasonal = df3.values.T
```

Задание матрицы переменных задачи (цен)

```
#задание переменных задачи
model.p = pyomo.Set(initialize=range(P))
model.w = pyomo.Set(initialize=range(S))
model.R = pyomo.Set(initialize=model.p*model.w)
#матрица цен продукта p в неделю w
model.r = pyomo.Var(model.R, domain=pyomo.NonNegativeReals)
```

Ограничения

```
#диапазоны на цены
for p in range(P):
    for w in range(W):
        model.c.add(model.r[p,w] >= limits['left']*r_seasonal[p,w])
        model.c.add(model.r[p,w] <= limits['right']*r_seasonal[p,w])
```

Целевая функция

```
#целевая функция
F = sum(sum(sum(model.r[p,w] * (M[s,p] * (-(model.r[p,w]/r_seasonal[p,w])**2 - 0.075 * model.r[p,w]/r_seasonal[p,w] + 2.85)/1.775
                        + sum((r_seasonal[o,w]/model.r[o,w])**2 * C[p,o] * M[s,o] for o in range(P)))\
                        for w in range(W)) for p in range(P)) for s in range(S))

model.obj = pyomo.Objective(expr=F, sense=pyomo.maximize)
```

Результаты

Информация о решении

```
Problem:
- Lower bound: -inf
  Upper bound: inf
  Number of objectives: 1
  Number of constraints: 240
  Number of variables: 120
  Sense: unknown
Solver:
- Status: ok
  Message: Ipopt 3.5.4\x3a Optimal Solution Found
  Termination condition: optimal
  Id: 0
  Error rc: 0
  Time: 4.988047361373901
Solution:
- number of solutions: 0
  number of solutions displayed: 0

718206607.4962041
```

Формирование цен другими способами

Функция вычисления цен в зависимости от способа

```
import random
def get_prices(which):
    P = product_rels.shape[0]
    W = prices.shape[0]
    r_seasonal = prices.values.T

    if(which == 'rand'):
        return [[random.uniform(0.75*r_seasonal[p,w],1.25*r_seasonal[p,w]) for w in range(W)]for p in range(P)]
    elif(which == 'min'):
        return [[0.75*r_seasonal[p,w] for w in range(W)]for p in range(P)]
    elif(which == 'max'):
        return [[1.25*r_seasonal[p,w] for w in range(W)]for p in range(P)]
    elif(which == 'nop'):
        return r_seasonal
```

Функция вычисления недельной выручки всех магазинов

```
def get_revenue(r: list) -> list:
    S = shop_info.shape[0]
    P = product_rels.shape[0]
    W = prices.shape[0]
    M = shop_info.values
    C = product_rels.values
    r_seasonal = prices.values.T

    return [sum(sum(r[p][w] * (M[s,p] * (-(r[p][w]/r_seasonal[p,w])**2 - 0.075 * r[p][w]/r_seasonal[p,w] + 2.85)/1.775
                            + sum((r_seasonal[o,w]/r[o][w])**2 * C[p,o] * M[s,o] for o in range(P))) \
                    for p in range(P))for s in range(S))for w in range(W)]
```

Проверка

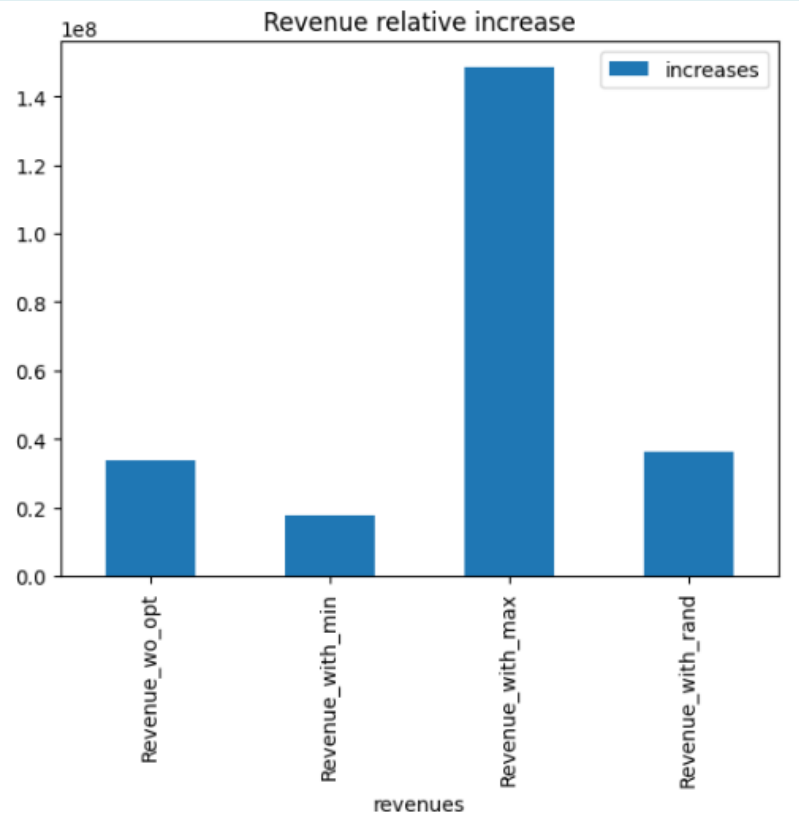
```
#проверка, что сумма выручки по неделям равна общей выручке
sum(Revenue_with_opt)
```

```
718206607.4962015
```

Сравнение выручек при различных подходах к формированию цен

Ожидаемая выручка по дням недели

Прирост выручки при оптимальном способе относительно остальных



Выводы

- Приведённые графики показывают, что при найденном решении достигается максимальная выручка относительно других способов формирования цен, значит, оно является оптимальным

- Наибольший прирост выручки наблюдается относительно модели, в которой были использованы максимально допустимые значения цен, что свидетельствует о наличии эффекта каннибализма акций взаимосвязанных товаров

Постановка задачи №2

- Модель прогноза продаж идентична представленной в задаче №1 с той лишь разницей, что изменять цену теперь можно только у ограниченного числа товаров в неделю

- Т.к. число товаров, которым можно изменять цену, всегда является целым, то задача преобразуется в задачу класса MINLP, в связи с чем был выбран солвер Bonmin

Построение оптимизационной модели

Замена переменной

$$r_{pw} = y_{pw} \cdot x_{pw} \cdot r_{pw}^{seasonal} + r_{pw}^{seasonal}$$

Преобразования

$$r_{pw} = y_{pw} \cdot x_{pw} \cdot r_{pw}^{seasonal} + r_{pw}^{seasonal}$$

$$1.25\, r_{pw}^{seasonal} \geq y_{pw} \cdot x_{pw} \cdot r_{pw}^{seasonal} + r_{pw}^{seasonal} \geq 0.75\, r_{pw}^{seasonal}$$

$$1.25\, r_{pw}^{seasonal} \geq r_{pw}^{seasonal} \cdot (x_{pw} + 1) \geq 0.75\, r_{pw}^{seasonal}$$

$$1.25 \geq x_{pw} + 1 \geq 0.75$$

$$0.25 \geq x_{pw} \geq -0.25$$

Ограничения

$$x \in [-0.25; 0.25]$$

$$y_{pw} = \{0, 1\}$$

$$\sum_{p=1}^{P} y_{pw} \leq lim,$$

где lim – это число продуктов в неделю, которым разрешено изменить цену

Реализация в Pyomo

Инициализация переменных задачи

```
#матрица коэффициентов для цен продукта p в неделю w
model.x = pyomo.Var(model.X, domain=pyomo.Reals, bounds=(-0.25, 0.25))
#матрица наличия акции на продукт p в неделю w
model.y = pyomo.Var(model.Y, domain=pyomo.Binary, initialize=0)
```

Ограничения

```
#ограничение количества продуктов в неделю, на которые действует акция
for w in range(W):
    model.c.add(sum(model.y[p,w] for p in range(P)) <= lim)
```

Целевая функция

```
#целевая функция
F = sum(sum(sum((model.y[p,w] * model.x[p,w] * r_seasonal[p,w] + r_seasonal[p,w]) *
                (M[s,p] * (-((model.y[p,w] * model.x[p,w] * r_seasonal[p,w] + r_seasonal[p,w])/r_seasonal[p,w])**2
                           - 0.075 * (model.y[p,w] * model.x[p,w] * r_seasonal[p,w] + r_seasonal[p,w])/r_seasonal[p,w] + 2.85)/1.775
                          + sum((r_seasonal[o,w]/(model.y[o,w] * model.x[o,w] * r_seasonal[o,w] + r_seasonal[o,w]))**2 * C[p,o] * M[s,o] for o in range(P)))\
                          for w in range(W)) for p in range(P)) for s in range(S))
```

Результаты решения при различных значениях числа акций в неделю

N_promo = 5

```
Problem:
- Lower bound: -inf
  Upper bound: inf
  Number of objectives: 1
  Number of constraints: 0
  Number of variables: 240
  Sense: unknown
Solver:
- Status: ok
  Message: bonmin\x3a Optimal
  Termination condition: optimal
  Id: 3
  Error rc: 0
  Time: 865.1423587799072
Solution:
- number of solutions: 0
  number of solutions displayed: 0

716417947.0125335
```

N_promo = 4

```
Problem:
- Lower bound: -inf
  Upper bound: inf
  Number of objectives: 1
  Number of constraints: 0
  Number of variables: 240
  Sense: unknown
Solver:
- Status: ok
  Message: bonmin\x3a Optimal
  Termination condition: optimal
  Id: 3
  Error rc: 0
  Time: 429.2070779800415
Solution:
- number of solutions: 0
  number of solutions displayed: 0

716416786.6237118
```

N_promo = 3

```
Problem:
- Lower bound: -inf
  Upper bound: inf
  Number of objectives: 1
  Number of constraints: 0
  Number of variables: 240
  Sense: unknown
Solver:
- Status: ok
  Message: bonmin\x3a Optimal
  Termination condition: optimal
  Id: 3
  Error rc: 0
  Time: 1125.9936499595642
Solution:
- number of solutions: 0
  number of solutions displayed: 0

716403961.2647941
```

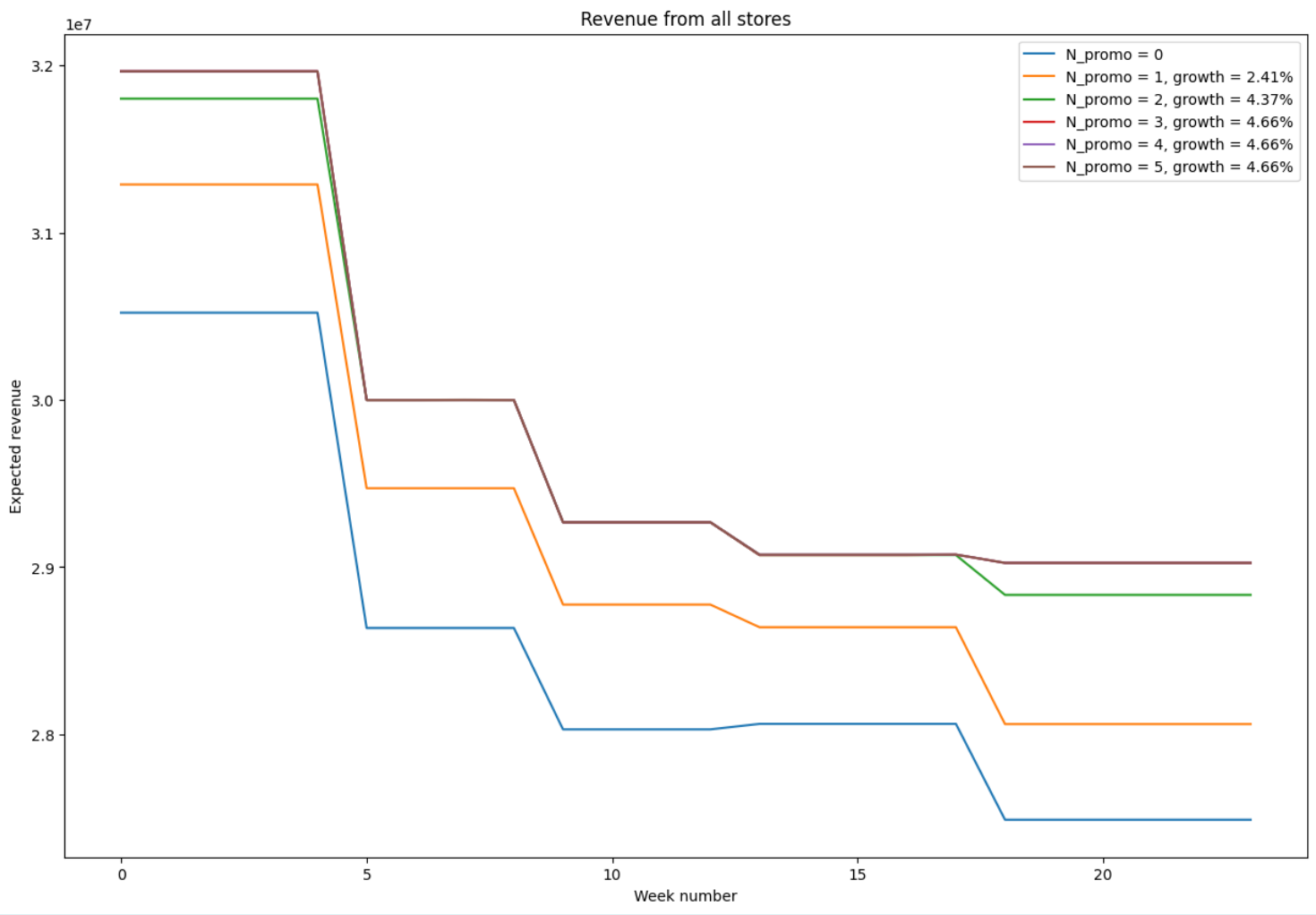
N_promo = 2

```
Problem:
- Lower bound: -inf
  Upper bound: inf
  Number of objectives: 1
  Number of constraints: 0
  Number of variables: 240
  Sense: unknown
Solver:
- Status: ok
  Message: bonmin\x3a Optimal
  Termination condition: optimal
  Id: 3
  Error rc: 0
  Time: 49.88869047164917
Solution:
- number of solutions: 0
  number of solutions displayed: 0

714435755.8769426
```

N_promo = 1

```
Problem:
- Lower bound: -inf
  Upper bound: inf
  Number of objectives: 1
  Number of constraints: 0
  Number of variables: 240
  Sense: unknown
Solver:
- Status: ok
  Message: bonmin\x3a Optimal
  Termination condition: optimal
  Id: 3
  Error rc: 0
  Time: 161.07398438453674
Solution:
- number of solutions: 0
  number of solutions displayed: 0

701002996.30018
```

Ожидаемая выручка и её относительный прирост при различных значениях числа акций в неделю



Проверка выполнения ограничений

Матрица переменных y решения при N_ promo = 4

```
array([[1., 1., 1., 0., 0.],
       [1., 1., 1., 0., 0.],
       [1., 1., 1., 0., 0.],
       [1., 1., 1., 0., 0.],
       [1., 1., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 0., 0.],
       [1., 0., 1., 1., 1.],
       [1., 0., 1., 1., 1.],
       [1., 0., 1., 1., 1.],
       [1., 0., 1., 1., 1.],
       [1., 0., 1., 1., 1.],
       [0., 1., 0., 1., 1.],
       [0., 1., 0., 1., 1.],
       [0., 1., 0., 1., 1.],
       [0., 1., 0., 1., 1.],
       [0., 1., 0., 1., 1.],
       [0., 1., 0., 1., 1.]])
```

Матрица переменных x решения при N_ promo = 2

```
array([[-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
        -6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
         6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
         6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
         6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
         6.64778562e-16, -6.64778562e-16],
       [-2.50000000e-01, -6.64778562e-16, -2.50000000e-01,
         6.64778562e-16, -6.64778562e-16],
       [-6.64778562e-16,  6.64778562e-16, -6.64778562e-16,
        -2.50000000e-01, -2.50000000e-01],
       [-6.64778562e-16,  6.64778562e-16, -6.64778562e-16,
```

Вывод

График ожидаемой выручки показал, что относительный прирост для 5-ти, 4-х и 3-х акций в неделю совпадает до сотых долей процента, есть предположение, что это происходит из-за того что переменная x может принимать нулевые или очень близкие к нулю значения, в следствие чего, если даже при этом соответствующая переменная y будет равна единице, цена на продукт всё равно останется неизменной.

Способ модификации оптимизационной модели

Для устранения представленной на предыдущем слайде проблемы, можно попробовать использовать встроенную в подмодуль pyomo.gdp функцию Disjunction, которая позволяет задать границы переменных в виде интервалов.

Так, если задать для переменной x интервал $x \in [-0.25; -0.0001] \cup [0.0001; 0.25]$

то во-первых, алгоритм теперь не сможет попасть в те области пространства решений, а во-вторых, оптимизационная модель станет ближе к реальной, т.к. теперь цена не сможет изменяться на величину порядка 10^{-16} ед.

Спасибо за внимание!