Instituto Federal de Goiás - Campus Goiânia

Bacharelado em Sistemas de Informação

Pesquisa Operacional

Aluno: Charles Lucas Santana de Souza

Professor: Eduardo Noronha de Andrade Freitas

Data: 29/10/2020

Santana Store

Charles Souza, Outubro de 2020

Conceitos

- Os três passos para resolver um problema
- Os três componentes de uma formulação
- Modelagem com variáveis de decisão coninuas
- Chamando um solucionador MIP

Declaração de problema

A Santana Store, uma loja de roupas feminina online que está com um problema para formar seu estoque. Precisa encontrar qual a melhor quantidade comprar de cada produto, suprindo as vendas mas tambem aumentando seu estoque a cada compra.

Formulação

Formulação utilizada será:

i. Variáveis de decisão
ii. Restrições
iii. Função objetiva

Variáveis de decisão

Para este problema, será utilizado duas variáveis continuas.

Variáveis de decisão:

x_1é a quantidade de blusas para comprar

x_2 é a quantidade de vestidos para comprar

Restrições

As únicas restrições são não estourar o orçamento de 1000,00 e não comprar quantidade menor de cada produto do que já é vendido hoje.

10x1 + 30x2 <= 1000

x1 >= 25

x2 >= 15

Objetivo

O objetivo aqui é otimizar a compra de estoque, comprando o mínimo que preciso para atender a demanda de vendas, mas aumentar meu estoque limitado ao capital que tenho disponível em cada compra.

Min E= 10x1 + 30x2

Formulação final

min E=10.x1 + 30.x2 <= 700

Sujeito a:

Restrições técnicas 10x1 + 30x2 <= 1000

x1 >= 25

x2 >= 15

Restrições de não negatividade

x1 >= 0

x2 >= 0

Implementação e otimização

A seguir, uma implementação utilizando python e a biblioteca PuLP da formulação que acabamos de ver. O código está disponível no repositório github.

```
1 # !pip install pulp
2 import pulp
3
4 # Definir o modelo
5 model = pulp.LpProblem('Otimizar_Estoque', sense=pulp.LpMaximize)
6
7 # Adicionar as variáveis
8 x = pulp.LpVariable.dicts(indexs=[1, 2], cat=pulp.LpContinuous, lowBound=0, name='x')
9
10 # Adicionar as Restrições
11 model.addConstraint(10*x[1] + 30*x[2] == 1000, name='restricao_1')
12 model.addConstraint(x[1] >= 25, name='restricao_2')
13 model.addConstraint(x[2] >= 15, name='restricao_3')
14
15 # Função Objetivo
16 model.setObjective(10* x[1] + 30* x[2])
17
18 # Optimizar
19 model.solve()
20
21 # Obter e imprimir a solução
22 x_sol = {i: x[i].value() for i in [1, 2]}
23 print(f'x = {x_sol}')
24
```

```
x = {1: 25.0, 2: 25.0}
```

- Na Linha 2, eu importo o pacote.
- Na Linha 5, ele inicia uma instância modelo e dá-lhe um nome.
- Na Linha 8, ele adiciona as duas variáveis de decisão de uma vez a partir de uma lista de índices e dá-lhe um nome.
- Na Linha 11, 12 e 13, ele adiciona as restrições do modelo e dá-lhe um nome também.
- Na Linha 15, ele define a função objetiva.
- Na Linha 19, quando o modelo está totalmente preenchido, chama o otimizador.
- Por fim, nas Linhas 22 e 23, ele recupera e imprime a solução.

A saída é um dicionário que se parece com este:

{1: 25.0, 2: 25.0} .

De onde concluímos que a solução ideal é:

x1=25 e x2=25