

APRENDIZAGEM POR PROJETO INTEGRADOR:

SPRINT 3

Autores:

Daniel Miguel da Silva

Emily Moreira Job

Gabriel Martins Gazaneo

José Ezenildo de Oliveira Campo

Júlia Vitória Oliveira da Silva

Marcelo Uchôas de Oliveira

Márcia Soares de Almeida

Professor M2 ou Orientador: Jean Carlos Lourenco Costa

Professor P2: Marcus Vinicius do Nascimento

Resumo do projeto:

Este trabalho está localizado na área de logística e tem por objetivo otimizar as rotas de três plantas (sites) pertencentes à mesma empresa por meio da aplicação de técnicas de Pesquisa Operacional. Pretende-se desenvolver um modelo que maximize a eficiência logística, minimizando custos e tempo de transporte entre as plantas e seus respectivos clientes, garantindo, assim, uma distribuição mais ágil e econômica dos produtos. O desenvolvimento deste projeto requer a criação de um modelo matemático robusto, capaz de analisar e otimizar as rotas de transporte, esse modelo possibilitará simulações e análises de cenários, fornecendo à empresa uma visão de cenário ideal.

Para alcançar esses objetivos, o projeto adota a metodologia Scrum uma metodologia ágil utilizada no gerenciamento de projetos, especialmente em contextos em que a complexidade e a incerteza são predominantes. Ele se baseia em ciclos de trabalho chamados de "sprints", nos quais uma equipe multifuncional trabalha de forma colaborativa para alcançar metas específicas dentro de um período definido, geralmente de duas a quatro semanas. O projeto está estruturado em quatro entregas, cada uma agregando novas funcionalidades e refinamentos ao modelo em desenvolvimento, nesta terceira entrega do projeto, será apresentado o resultado da modelagem matemática

desenvolvida no ambiente Colab usando Python, juntamente com uma visualização dos resultados obtidos em dashboard no Power BI.

Palavras-Chave: Pesquisa Operacional; Otimização; Cenário ideal, Modelo matemático.

Abstract:

This work is located in the logistics area, aiming to optimize the routes of three plants belonging to the same company through the application of Operational Research techniques. The goal is to develop a model that maximizes logistical efficiency, minimizing costs and transportation time between the plants and their respective clients, thus ensuring a more agile and economical distribution of products. The development of this project requires the creation of a robust mathematical model capable of analyzing and optimizing transportation routes; this model will enable simulations and scenario analyses, providing the company with an ideal scenario view.

To achieve these objectives, the project adopts the Scrum methodology, an agile methodology widely used in project management, especially in contexts where complexity and uncertainty are predominant. It is based on work cycles called "sprints," in which a multifunctional team collaboratively works to achieve specific goals within a defined period, usually two to four weeks. The project is structured into four deliverables, each adding new features and refinements to the model under development. In this third deliverable of the project, the solver for the mathematical formula developed in Colab using Python will be presented, along with a dashboard visualization in Power BI.

Keywords: Operational Research; Optimization; Ideal scenario, Mathematical model.

Contextualização do projeto

A Pesquisa Operacional é uma disciplina aplicada que visa resolver problemas relacionados à condução e coordenação das operações em uma organização. Ela é utilizada em diversas áreas, como manufatura, transporte, construção, entre outras, e utiliza métodos científicos para investigar e modelar problemas empresariais complexos.

No contexto deste projeto, a PO será aplicada para encontrar “soluções ótimas” que permitam a melhoria das rotas de transporte entre três plantas da mesma empresa. Isso envolve a consideração de variáveis como distância, tempo de percurso, capacidade de produção e demanda de cada planta.

Segundo Hillier & Lieberman (1995), o Modelo do Problema de Transporte descreve a distribuição de commodities de diferentes grupos, originadas de centros de fornecimento (chamados de origens), para grupos de centros de distribuição (chamados de destinos), visando minimizar o custo total de distribuição. Cada origem tem sua capacidade e cada destino tem sua demanda.

No contexto desta entrega, o Modelo do Problema de Transporte irá auxiliar na criação de uma função que tenha como objetivo determinar e configurar a otimização dos dados tratados.

Esse relatório apresenta a terceira fase (Sprint) de um projeto que visa otimizar os custos operacionais de três plantas pertencentes à mesma empresa. Utilizando a metodologia ágil Scrum, o

conteúdo neste relatório segue o escopo pré-definido na entrega da primeira fase, conforme o quadro a seguir:

Quadro 1 – Cronograma de entregas

Data da entrega	Proposta da entrega	Status
13/03/2024	Kick-Off (apresentação do problema)	Concluída
17/04/2024	Estabelecimento das plataformas de comunicação e documentação do projeto, Análise e tratamento da base de dados	Concluída
08/05/2024	Apresentação da estrutura do protótipo modelo matemático que apresenta o “cenário ideal” de 2023	Concluída
29/05/2024	Dashboard comparativo entre modelo real VS modelo ideal de 2023 e modelo ideal para 2024	Concluída
19/06/2024	Correção de bugs e possíveis melhorias de acordadas com o cliente	

Objetivos da entrega

Os objetivos estabelecidos para essa entrega do projeto consistem em:

- i) Atualizar e verificar a estrutura atual do Colab a fim de obter o resultado do problema de programação linear das rotas de 2023;
- ii) Desenvolver um modelo ideal de roteirização visando minimizar os custos de 2024;
- iii) Elaborar um dashboard interativo no Power BI, com o intuito de comparar as operações feitas em 2023 com o modelo ideal proposto para 2023 e a apresentar um modelo ideal para 2024.

1. Fundamentação dos métodos analíticos e das tecnologias utilizadas

A análise de rotas é uma tarefa importante em diversos setores, principalmente na indústria, com o objetivo de otimizar trajetos, identificar ineficiências e melhorar a eficácia operacional. Na segunda entrega deste projeto, a equipe concentrou seus esforços em corrigir as anomalias encontradas na base de dados, trabalhando nelas, transformando-as e exportando para colaborar com o objetivo de demonstrar uma função matemática que busca a melhor otimização para o valor do frete por unidade, uma variável usada para calcular o cenário ideal de comercialização/desempenho.

1.1. Tecnologias da Informação

Durante o curso deste projeto, nossa equipe empregou uma variedade de tecnologias da informação para facilitar a análise de dados, a colaboração eficiente e a visualização de resultados. A seguir, descrevemos as principais tecnologias utilizadas:

a. Jira Software: O Jira Software foi a plataforma central para o gerenciamento desta sprint. Utilizamos suas funcionalidades para criar e priorizar tarefas, acompanhar o progresso das atividades, gerenciar as entregas e facilitar a comunicação entre os membros da equipe. A integração contínua com o fluxo de trabalho ágil permitiu uma abordagem flexível e adaptativa para lidar com os desafios e mudanças ao longo da sprint.

b. GitHub: O GitHub foi escolhido como repositórios para armazenar, visionar e colaborar no desenvolvimento do projeto. A capacidade de ramificação e mesclagem do GitHub contribuiu para coordenar o trabalho em equipe e garantir a integridade do projeto.

c. SQL (Structured Query Language): É uma linguagem padrão para manipulação de registros em bancos de dados relacionais, no contexto desta entrega foi utilizado para encontrar anomalias na base de dados e relacionar informações.

d. Power BI: O Power BI foi utilizado para confirmar os resultados da base de dados SQL e principalmente visualizar as rotas com valores e um mapa, além de proporcionar uma tela que permite especular possíveis otimizações.

e. Google Colab: O Google Colab foi escolhido porque oferece um ambiente de desenvolvimento Python na nuvem, permitindo acesso gratuito a GPUs e TPUs para cálculos intensivos, além de ser acessível de qualquer lugar, sem a necessidade de instalar software localmente.

f. Python: Python foi escolhido como o software para calcular a função variável visando a otimização dos dados trabalhados e corrigidos.

g. Microsoft Power BI: O Microsoft Power BI foi escolhido como base para criar um dashboard interativo, com ênfase em contar uma história dos dados tratados e obtidos

2. Desenvolvimento do Projeto

O desenvolvimento do projeto foi baseado na remodelação e na aquisição dos dados otimizados, através do modelo de transportes, utilizando Colab e Python, para sua integração em um comparativo entre 2023 e 2024. Utilizando o valor unitário do frete como variável na base anual, os dados foram apresentados em um dashboard interativo no Power BI:

2.1. Otimização do Cenário ideal 2023:

Foi utilizado o ambiente do Colab para resolver o problema de transporte, empregando o software Python para otimização das rotas. O foco foi obter o menor custo possível do frete dentro da base ano. Segue a explicação das fórmulas e códigos utilizados:

Imagem 1 Biblioteca Pulp.

```
!pip install pulp

Collecting pulp
  Downloading PuLP-2.8.0-py3-none-any.whl (17.7 MB)
    17.7/17.7 MB 37.1 MB/s eta 0:00:00
Installing collected packages: pulp
Successfully installed pulp-2.8.0
```

Instalar biblioteca “pulp” = !pip Install pulp

É uma linha de comando usada para instalar a biblioteca PuLP. O comando pip install pulp baixará e instalará a biblioteca PuLP e suas dependências em seu sistema.

A biblioteca PuLP é uma ferramenta de modelagem e resolução de problemas de programação linear (LP) em Python. Ela fornece uma interface conveniente para definir e resolver problemas de otimização linear, permitindo que os usuários criem modelos matemáticos de forma fácil e eficiente.

Imagem 2 Importação da Biblioteca e Criação do Problema

```
[ ] from pulp import *

[ ] # Criação do problema

[ ] problema1 = LpProblem('mizimize Custos',LpMinimize)

/usr/local/lib/python3.10/dist-packages/pulp/pulp.py:1316: UserWarning: Spaces are not permitted in the name. Converted to '_'
warnings.warn("Spaces are not permitted in the name. Converted to '_'")
```

Pedir para todas a funções da pulp funcionarem = from pulp import *

Este comando é usado para importar todos os símbolos (funções, classes, etc.) do módulo pulp para o seu programa Python atual. Isso permite que você use os recursos da biblioteca PuLP em seu código.

Criação do problema = problema1 = LpProblem('Minimiza Custos', LpMinimize)

Foi declarado a variável problema1 em que ela recebe os seguintes parâmetros, ‘Produção’ que serve como o nome do problema e tem como objetivo identificar o problema, no caso estamos trabalhando com múltiplos problemas.

LpProblem é uma classe fornecida pela biblioteca PuLP para representar um problema de programação linear (LP, do inglês Linear Programming). Você usa esta classe para criar um novo problema LP, especificando seu nome e se é um problema de maximização ou minimização.

A função LpMinimize, tem como função minimizar o resultado do código para ter um melhor desempenho na execução do comando, em programação linear.

Imagem 3 Variáveis de Decisão

```
[ ] # variáveis de decisão

[ ] x15 = LpVariable('x15', lowBound = 0)
    x18 = LpVariable('x18', lowBound = 0)
    x19 = LpVariable('x19', lowBound = 0)
    x110 = LpVariable('x110', lowBound = 0)
    x111 = LpVariable('x111', lowBound = 0)
    x124 = LpVariable('x124', lowBound = 0)
    x125 = LpVariable('x125', lowBound = 0)
    x126 = LpVariable('x126', lowBound = 0)
    x127 = LpVariable('x127', lowBound = 0)
    x128 = LpVariable('x128', lowBound = 0)
    x129 = LpVariable('x129', lowBound = 0)
    x130 = LpVariable('x130', lowBound = 0)
    x131 = LpVariable('x131', lowBound = 0)
    x132 = LpVariable('x132', lowBound = 0)
    x133 = LpVariable('x133', lowBound = 0)
    x134 = LpVariable('x134', lowBound = 0)
    x135 = LpVariable('x135', lowBound = 0)
    x136 = LpVariable('x136', lowBound = 0)
```

Variável de Decisão = x15 = LpVariable ('x15', lowBound = 0)

Foi declarada a função na qual vai receber o valor adquirido, após isso aplicamos a função LpVariable que tem como objetivo criar uma variável de decisão em programação linear, após isso temos o índice para encontrar o valor da célula desejada, e após temos o lowBound a função com o objetivo de definir uma variável com limite inferior de 0, neste caso esta função faz com que o limite seja no mínimo 0.

Imagem 4 Função objetivo

```
# Função objetivo

[ ] problema1 += 0.26*x15+0.26*x18+0.27*x19+0.30*x110+0.27*x111+0.29*x124+0.25*x125+0.30*x126+0.28*x127+0.31*x128+0.33*x129+0.44*x130+0.29*x131+0.35*x132+0.60*x133+0.27*x134+0.27*x135+0.48*x136+0.47*x137+0.46*
```

Função objetivo = problema1 += "Custo Unitário "*"variável" + "Custo Unitário "*"variável"

A função objetivo é uma expressão matemática que você deseja minimizar ou maximizar. No caso de um problema de minimização, como no projeto, queremos minimizar o "Custo ". A função objetivo é uma combinação linear das variáveis de decisão, ponderadas pelos seus respectivos coeficientes.

Imagem 5 Restrições do problema

```
[ ] # Restrições

[ ] problema1 += x15 + x18 + x19 + x110 + x111 + x124 + x125 + x126 + x127 + x129 + x130 + x131 + x132 + x133 + x134 + x135 + x136 + x137 + x138 + x139
    problema1 += x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 + x29 + x210 + x211 + x212 + x213 + x214 + x215 + x216 + x217 + x218 + x219 + x220 + x221 + x22
    problema1 += x31 + x32 + x33 + x34 + x35 + x36 + x37 + x38 + x39 + x310 + x311 + x319 + x320 + x321 + x322 + x323 + x324 + x325 + x326 + x327 + x328 + x32
    problema1 += x21 + x31 == 2707800
    problema1 += x22 + x32 == 1836600
    problema1 += x23 + x33 == 1274700
    problema1 += x24 + x34 == 2439900
    problema1 += x15 + x25 + x35 == 5714100
    problema1 += x26 + x36 == 2725500
    problema1 += x27 + x37 == 2887500
    problema1 += x18 + x28 + x38 == 3846300
    problema1 += x19 + x29 + x39 == 7242300
    problema1 += x110 + x210 + x310 == 4628400
    problema1 += x111 + x211 + x311 == 6508200
    problema1 += x212 == 1214700
    problema1 += x213 == 9900
```

```
# Restrição = problema1 += x21 + x31 == 5973721
```

As restrições são condições que limitam as soluções válidas para o problema. Elas podem ser igualdades, desigualdades ou intervalos.

```
# Resolvendo o Problema = problema1.solve()
```

Imagem 6 Resolução do Problema

```
[ ] # Resolvendo o problema

[ ] problema1.solve()

↔ 1
```

É responsável por resolver o problema de otimização definido anteriormente usando o solver associado à biblioteca PuLP.

problema1: Refere-se ao objeto do tipo LpProblem que representa o problema de otimização que queremos resolver.

.solve(): É o método que aciona o solver para resolver o problema.

```
# Imprimir variáveis de otimização = for v in problema1.variables():
```

```
print (v.name, "=", v.varValue).
```

Imagem 7 Imprimir variáveis de otimização

```
[ ] # Imprimir variáveis de otimização

[ ] for v in problema1.variables():
    print (v.name, "=", v.varValue)

↔ x110 = 0.0
   x111 = 6508200.0
   x124 = 1571700.0
   x125 = 35100.0
   x126 = 0.0
   x127 = 0.0
   x128 = 5429400.0
   x129 = 0.0
   x130 = 0.0
   x131 = 0.0
```

Este código percorre todas as variáveis de otimização no problema. Para cada variável, ele imprime o nome da variável e o valor otimizado correspondente.

```
# Mostrar resultado do custo mínimo = print('Resultado do custo mínimo
=',value(problema1.objective)).
```

Imagem 8 Mostrar resultado do custo mínimo

```
[ ] # Mostrar resultado do custo mínimo

[ ] print('Resultado do custo mínimo =',value(problema1.objective))

Resultado do custo mínimo = 42740709.0
```

Esse código exibiu na tela o texto 'Resultado do custo mínimo =' seguido pelo valor calculado da função objetivo do problema de otimização `problema1`.

Gerar Unidade multiplicado por Valor unitário = lista : [0.30, 0.27, 0.29, 0.25, ...]

Imagem 9 Gerar unidade multiplicada por valor unitário

```
[ ] # Gerar Unidade multiplicado por Valor unitário

[ ] lista = [0.30, 0.27, 0.29, 0.25, 0.30, 0.28, 0.31, 0.33, 0.44, 0.29, 0.35, 0.

[ ] v_otim = []
    result = []
    nomes = []

x = 0
for v in problema1.variables():
    print(v.name, "=", lista[x] * v.varValue)
    x+=1
```

Este código calcula e imprime o produto de valores unitários (da lista) pelos valores das variáveis do problema de otimização problema1, iterando sobre cada variável e incrementando um índice para acessar os elementos da lista.

v.name = nome da variável.

v.varValue = valor da variável após a otimização.

lista[x] * v.varValue = calcula o produto do valor unitário (da lista) pelo valor da variável.

print(v.name, "=", lista[x] * v.varValue) = imprime o nome da variável seguido pelo produto calculado.

x += 1 = incrementa o índice x em 1 a cada iteração para acessar o próximo valor na lista.

Imagem 10 Geração de DataFrame com Resultados Otimizados


```
[ ] x = 0
    for v in problema1.variables():
        nomes.append(v.name)
        v_otim.append(v.varValue)
        result.append(lista[x] * v.varValue)
        x += 1

[ ] import pandas as pd

df = pd.DataFrame({
    'nomes': nomes,
    'variaveis_de_otimizacao': v_otim,
    'funcao': lista,
    'resultado': result
})
df
```

Este código percorre as variáveis de um problema de otimização, armazena seus nomes e valores em listas, calcula o produto desses valores com uma lista de valores unitários, e armazena os resultados. Em seguida, usa a biblioteca pandas para criar um DataFrame contendo essas informações organizadas em colunas.

`x = 0`: Inicializa o índice `x` em 0.

`for v in problema1.variables()`: Itera sobre cada variável `v` no problema de otimização `problema1`.

`nomes.append(v.name)`: Adiciona o nome da variável à lista `nomes`.

`v_otim.append(v.varValue)`: Adiciona o valor da variável otimizada à lista `v_otim`.

`result.append(lista[x] * v.varValue)`: Calcula o produto do valor unitário (da lista) pelo valor da variável e adiciona à lista `result`.

`x += 1`: Incrementa o índice `x` em 1 a cada iteração para acessar o próximo valor na lista.

`import pandas as pd`: Importa a biblioteca pandas, que é utilizada para manipulação de dados.

`df = pd.DataFrame({...})`: Cria um DataFrame `df` com colunas 'nomes', 'variaveis_de_otimizacao', 'funcao', e 'resultado', usando as listas `nomes`, `v_otim`, `lista`, e `result` respectivamente.

`df`: Exibe o DataFrame.

Imagem 11 Salvar e Baixar DataFrame como CSV no Google Colab

```
[ ] df.to_csv('listas_ajustadas.csv', index=False)

[ ] from google.colab import files
    files.download('listas_ajustadas.csv')
```

Este código salva um DataFrame `df` em um arquivo CSV chamado `listas_ajustadas.csv` sem incluir os índices das linhas. Em seguida, utiliza a funcionalidade do Google Colab para baixar esse arquivo CSV para o computador local.

`df.to_csv('listas_ajustadas.csv', index=False)`: Este comando salva o DataFrame `df` em um arquivo CSV chamado `listas_ajustadas.csv`. O parâmetro `index=False` garante que os índices das linhas do DataFrame não sejam incluídos no arquivo CSV.

`from google.colab import files`: Importa o módulo `files` da biblioteca `google.colab`, que é específica para o ambiente Google Colab.

`files.download('listas_ajustadas.csv')`: Usa o método `download` do módulo `files` para baixar o arquivo `listas_ajustadas.csv` para o computador local.

Deste modo, após obter a solução e o resultado otimizado do modelo de transportes, foi desenvolvida uma proposta para a otimização dos valores de 2024. Nesta proposta, juntamente com a demanda consultada, haveria um aumento de 5% na carga das rotas. Essa contabilização tem como objetivo dimensionar o problema desenvolvido para uma perspectiva futura.

2.2. Criação da Planilha:

Com base na otimização dos dados realizada utilizando Colab e Python, foram geradas planilhas em formato CSV. Essas planilhas servem para comparar a base de dados que será representada no dashboard interativo no Power BI.

Para a criação dessas planilhas, foi utilizado o Google Colab. Embora a maior parte do código seja semelhante, há uma diferença no final, onde os seguintes códigos são utilizados para gerar a planilha:

Imagem 12 Visualização dos dados em Lista

```
[ ] # Mostrar resultado do custo mínimo

[ ] print('Resultado do custo mínimo =',value(problema1.objective))

Resultado do custo mínimo = 42740709.0

[ ] lista = [0.30, 0.27, 0.29, 0.25, 0.30, 0.28, 0.31, 0.33, 0.44, 0.29,
v_otim = []
result = []
nomes = []

[ ] x = 0
for v in problema1.variables():
    print(v.name, "=", lista[x] * v.varValue)
    x += 1
```

Este código é útil para ver como os valores otimizados das variáveis interagem com um conjunto de dados (`lista`) e exibir os resultados dessas interações. Segue as expressões aplicadas:

Função: Exibe o valor do objetivo (custo mínimo) do problema de otimização `problema1`.

value(problema1.objective): Obtém o valor do objetivo após a otimização.

Saída: Mostra na tela o custo mínimo calculado pela função objetivo do problema.

lista: Contém valores que serão utilizados na multiplicação com os valores otimizados das variáveis.

v_otim: Lista vazia para armazenar os valores otimizados das variáveis.

result: Lista vazia para armazenar os resultados das multiplicações.

nomes: Lista vazia para armazenar os nomes das variáveis.

x = 0: Inicializa o índice x com 0.

for v in problema1.variables(): Itera sobre todas as variáveis do problema de otimização problema1.

Dentro do laço for:

print(v.name, "=", lista[x] * v.varValue): Exibe o nome da variável (v.name) e o resultado da multiplicação do valor correspondente em lista pelo valor otimizado da variável (v.varValue).

x += 1: Incrementa x para acessar o próximo valor em lista na próxima iteração.

Imagem 13 Criação do dataframe

```
[ ] x = 0
    for v in problema1.variables():
        nomes.append(v.name)
        v_otim.append(v.varValue)
        result.append(lista[x] * v.varValue)
        x += 1

[ ] df = pd.DataFrame({
    'nomes': nomes,
    'variaveis_de_otimizacao': v_otim,
    'funcao': lista,
    'resultado': result
})
df
```

O primeiro bloco de código itera sobre as variáveis de um problema de otimização, preenchendo três listas: nomes com os nomes das variáveis, v_otim com os valores otimizados das variáveis, e result com os resultados da multiplicação de valores da lista lista pelos valores otimizados das variáveis.

O segundo bloco de código cria um DataFrame usando pandas, organizando os dados coletados nas listas em uma estrutura tabular com quatro colunas: 'nomes', 'variaveis_de_otimizacao', 'funcao', e 'resultado'. Em seguida, exibe esse DataFrame.

x = 0: Inicializa o índice x com 0.

for v in problema1.variables(): Itera sobre todas as variáveis do problema de otimização problema1.

Dentro do laço for:

`nomes.append(v.name)`: Adiciona o nome da variável `v` à lista `nomes`.

`v_otim.append(v.varValue)`: Adiciona o valor otimizado da variável `v` à lista `v_otim`.

`result.append(lista[x] * v.varValue)`: Calcula o produto do valor na lista `lista` na posição `x` com o valor otimizado da variável `v` e adiciona esse resultado à lista `result`.

`x += 1`: Incrementa `x` para acessar o próximo valor em `lista` na próxima iteração.

`pd.DataFrame({ ... })`: Cria um `DataFrame`, que é uma estrutura de dados bidimensional fornecida pela biblioteca `pandas`.

Dicionário passado como argumento para `pd.DataFrame`:

'`nomes`': `nomes`: Cria uma coluna chamada '`nomes`' que contém a lista `nomes`.

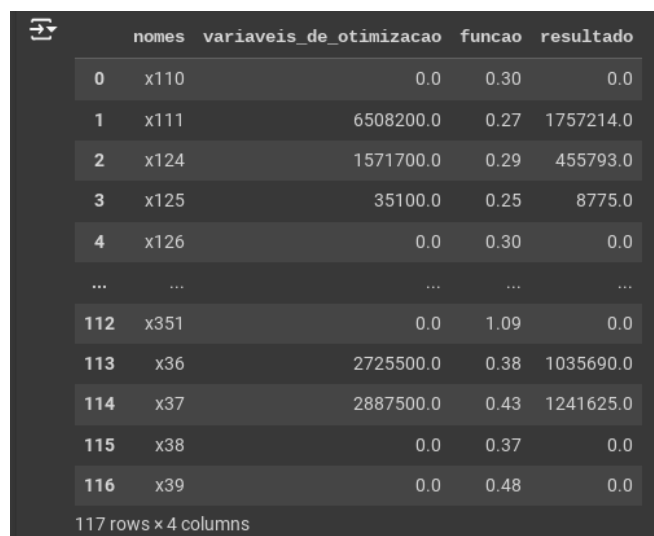
'`variaveis_de_otimizacao`': `v_otim`: Cria uma coluna chamada '`variaveis_de_otimizacao`' que contém a lista `v_otim`.

'`funcao`': `lista`: Cria uma coluna chamada '`funcao`' que contém a lista `lista`.

'`resultado`': `result`: Cria uma coluna chamada '`resultado`' que contém a lista `result`.

`df`: Exibe o `DataFrame` criado.

Imagem 14 Visualização do Dataframe



	nomes	variaveis_de_otimizacao	funcao	resultado
0	x110	0.0	0.30	0.0
1	x111	6508200.0	0.27	1757214.0
2	x124	1571700.0	0.29	455793.0
3	x125	35100.0	0.25	8775.0
4	x126	0.0	0.30	0.0
...
112	x351	0.0	1.09	0.0
113	x36	2725500.0	0.38	1035690.0
114	x37	2887500.0	0.43	1241625.0
115	x38	0.0	0.37	0.0
116	x39	0.0	0.48	0.0

117 rows x 4 columns

Imediatamente após a elaboração do `DataFrame`, procedeu-se à sua exportação e armazenamento em um arquivo `CSV`. A seguir, será apresentado o código que realizou essa operação.

Imagem 15 Conversão do dataframe para csv

```
df.to_csv('listas_ajustadas.csv', index=False)

from google.colab import files
files.download('listas_ajustadas.csv')
```

Aqui estão as expressões trabalhadas e suas explicações detalhadas:

`df.to_csv('listas_ajustadas.csv', index=False)`: Salva o DataFrame `df` em um arquivo CSV chamado 'listas_ajustadas.csv', sem incluir o índice das linhas.

`from google.colab import files`: Importa o módulo `files` do Google Colab para manipular arquivos.

`files.download('listas_ajustadas.csv')`: Baixa o arquivo 'listas_ajustadas.csv' para o computador local.

2.3. Representação dos Dados no Power BI:

A plataforma Power BI foi utilizada no contexto dessa entrega para visualização dos dados relacionados à capacidade dos caminhões, produtividade dos veículos, custos de rotas, comparação de cenários econômicos e previsão de custos para 2024.

2.3.1 Criação do campo “percentual de capacidade utilizada”

Este campo tem como objetivo principal verificar quantos por cento da capacidade total dos caminhões P12 e P24 está sendo efetivamente utilizada. A análise desse percentual permite uma melhor gestão dos recursos logísticos, identificando oportunidades para otimização do uso dos veículos e, consequentemente, a redução de custos operacionais. O cálculo foi feito utilizando a ferramenta DAX do Power BI usando como parâmetros a capacidade máxima pré definida, 1800 unidades para o caminhão P12 e 3600 unidades para caminhão P24.

Imagem 16 Cálculo da taxa de ocupação dos veículos

The screenshot shows the DAX formula bar with the following code:

```
1 Porcentagem Capacidade Utilizada =
2 VAR CapacidadeVeiculo =
3 SWITCH(
4 'CIF E FOB'[VEICULO_ROTAS],
5 "P12", 1800,
6 "P24", 3600,
7 BLANK() -- Caso o tipo de veículo não seja nem A nem B, retorna BLANK()
8 )
9 RETURN
10 IF(
11 CapacidadeVeiculo <> BLANK(),
12 DIVIDE('CIF E FOB'[Quantidade transportada], CapacidadeVeiculo, 0),
13 0
14 ) * 100
15
```

Below the formula bar, a table is displayed with columns: E_ROTAS, Valor unitario, Valor Unitário, Quant*Valor unita, Coluna, Valor unitario 2024, Porcentagem Capacidade Utilizada, and Porcentagem de produtividade. The table contains data for various routes (E_ROTAS) and their corresponding utilization percentages (Porcentagem Capacidade Utilizada).

E_ROTAS	Valor unitario	Valor Unitário	Quant*Valor unita	Coluna	Valor unitario 2024	Porcentagem Capacidade Utilizada	Porcentagem de produtividade
16	0	\$0	0	\$0	0	83,00	83%
79	0	\$0	0	\$0	0	83,00	83%
89	0	\$0	0	\$0	0	83,00	83%
147	0	\$0	0	\$0	0	83,00	83%
153	0	\$0	0	\$0	0	83,00	83%
158	0	\$0	0	\$0	0	83,00	83%
163	0	\$0	0	\$0	0	83,00	83%
168	0	\$0	0	\$0	0	83,00	83%
169	0	\$0	0	\$0	0	83,00	83%

2.3.2 Criação do campo Valores para 2024

Foi aplicado um aumento de 5% no valor unitário da mercadoria (valor esse dado pelo cliente), com o intuito de projetar o valor estimado para o ano de 2024. Para fazer a coluna foi aplicada uma multiplicação de 1.05 sobre o valor de 2023.

Imagem 17 Previsão 2024

```
1 Previsão de 2024 = 'Burndown das Sprints'[Arredondar] * 1.05
```

3. Resultados Obtidos

Os resultados alcançados pela equipe são altamente satisfatórios, evidenciando a utilização de diversas tecnologias. Através do uso do Python, foi possível modelar um problema de programação linear, resultando na minimização dos custos de frete. Além disso, essa solução foi integrada ao Power BI, permitindo a visualização detalhada dos dados e facilitando a análise dos resultados obtidos.

3.1. Python

Com base nas aplicações descritas no desenvolvimento do projeto, após a otimização e a resolução do problema de transportes, foi possível obter os seguintes dados:

Resultado da Otimização 2023:

Custo real total do frete em 2023	R\$ 59.434.968,00
Custo no cenário ideal em 2023	R\$ 42.766.380,00
Diferença	R\$ 16.668.588,00

Além disso, houve uma redução de 39,28% nas rotas, passando de 117 rotas para apenas 51 com a otimização.

O resultado da otimização para 2024 foi de R\$ 105.917.933,78 , com a projeção do melhor cenário em 2024 indicando um aumento de 5% na demanda dos clientes.

Assim, observamos que o valor gasto com frete em 2023 foi aproximadamente 28% mais caro em comparação com o cenário ideal, que é de R\$ 42.766.380,00. A diferença entre o custo real total do frete (R\$ R\$ 59.434.968,00) e o cenário ideal foi de R\$ 16.668.588,00. Além disso, houve uma redução de 39,28% nas rotas, passando de 117 rotas para apenas 51 com a otimização. O resultado da otimização para 2024 foi de R\$ 92.625.307,93, com a projeção do melhor cenário em 2024 indicando um aumento de 5% na demanda dos clientes.

3.2. Criação das planilhas em csv:

Os resultados obtidos através da consulta no Google Colab com Python levaram à criação de tabelas que auxiliam na construção de dashboards interativos no Power BI. Seguem os resultados provenientes das otimizações realizadas:

Imagem 18 Tabela Cenário Ideal 2023

Rota	Unidade a ser enviada	Valor unitário	Resultado (Uni X Vlr U)	Fábrica CO	V. Fábrica	V. Cliente	Cliente CO
X15	1116900	0.26	290394	3423909	1	5	2305
X18	1042500	0.26	271050	3423909	1	8	2308
X19	657600	0.27	177552	3423909	1	9	2309
X110	2517000	0.30	755100	3423909	1	10	2310
X111	1068900	0.27	288603	3423909	1	11	2311
X124	16800	0.29	4872	3423909	1	24	2324
X125	3000	0.25	750	3423909	1	25	2325

A planilha exibe os resultados de uma otimização de rotas, apresentando os seguintes dados organizados por colunas:

Rota: Identifica a rota concatenada específica.

Unidade a ser enviada: Número de unidades a serem enviadas em cada rota.

Valor unitário: Custo unitário associado a cada rota.

Resultado (Unidade X Valor): Resultado da multiplicação das unidades enviadas pelo valor unitário.

Fábrica CO: Código ou identificação da fábrica.

V. Fábrica: Valor da fábrica associado a cada rota.

V. Cliente: Valor cliente associado a cada rota.

Cliente CO*: Código ou identificação do cliente.

Imagem 19 Cenário ideal 2024

Rota	Unidade a ser enviada	Valor unitário	Resultado (Uni X Vlr Un)	Fábrica CO	V. Fábrica	V. Cliente	Cliente CO
x110	2098730	0.27	566,657	3423909	1	10	2310
x111	2119265	0.28	593,394	3423909	1	11	2311
x124	20427048	0.29	5923843.92	3423909	1	24	2324
x125	7828763	0.31	2426916.53	3423909	1	25	2325
x126	1614173	0.28	45,196,844	3423909	1	26	2326
x127	0	0.3	0	3423909	1	27	2327
x128	6145143	0.26	1,597,737	3423909	1	28	2328

A planilha exibe os resultados de uma otimização de rotas, apresentando os seguintes dados organizados por colunas:

Rota: Identifica a rota concatenada específica.

Unidade a ser enviada: Número de unidades a serem enviadas em cada rota.

Valor unitário: Custo unitário associado a cada rota.

Resultado (Unidade X Valor): Resultado da multiplicação das unidades enviadas pelo valor unitário.

Fábrica CO: Código ou identificação da fábrica.

V. Fábrica: Valor da fábrica associado a cada rota.

V. Cliente: Valor cliente associado a cada rota.

Cliente CO: Código ou identificação do cliente.

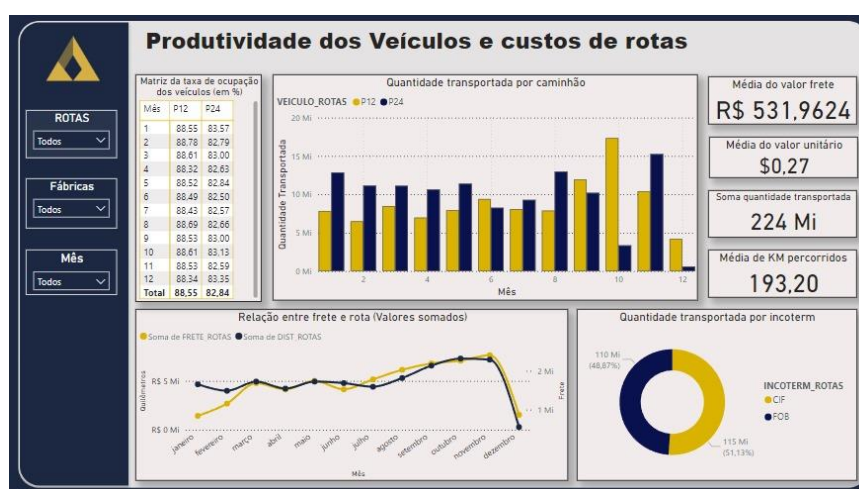
Nota-se que a estrutura das planilhas é a mesma, porém, cada uma contém os valores correspondentes aos seus respectivos anos.

3.3. Visualização em Dashboard interativo no Power BI:

3.3.1 Dashboard de Produtividade e Custos:

Foi desenvolvida uma tela (dashboard) no Power BI destinada à visualização da produtividade dos veículos e custos de rotas. Esta dashboard oferece uma visão sobre o desempenho operacional, permitindo monitorar e analisar a eficiência dos veículos em serviço e os custos associados às rotas percorridas. Com essas informações, é possível identificar pontos de melhoria e tomar decisões informadas para aumentar a produtividade e reduzir despesas.

Imagem 20 Dashboard “Produtividade e Custo”



3.3.1 Dashboard da Comparação de Cenários Econômicos e previsão de 2024:

Outra tela criada no Power BI é dedicada à comparação entre o cenário econômico real de 2023 e o cenário otimizado de 2023. Esta comparação permite visualizar as diferenças entre a realidade observada e as projeções feitas com base em otimizações, facilitando a identificação de “gaps” e oportunidades de melhoria. A análise comparativa ajuda a entender os impactos das estratégias adotadas e a ajustar planos futuros com maior precisão.

Por fim, foi desenvolvida uma tela específica para a previsão de custos para o ano de 2024. Esta tela utiliza os dados atualizados e o aumento projetado no valor das mercadorias para fornecer uma estimativa dos custos esperados no próximo ano. A previsão de custos é uma ferramenta essencial para o planejamento financeiro, permitindo que a organização se prepare adequadamente para o futuro e tome decisões estratégicas com base em dados precisos e atualizados.

Imagem 21 Dashboard Comparação de Cenários Econômicos e previsão de 2024



Referências

Ballou H. R. Gerenciamento da cade de suprimentos /Logística empresarial, 2004.

BELFIORE, Patrícia; FÁVERO, Luiz Paulo. Pesquisa operacional para cursos de Engenharia.

Hillier, Frederick S. - Introdução à pesquisa operacional/ Frederick S. Hillier, Gerald J. Lieberman;

tradução Ariovaldo Griesi; revisão técnica João Chang Junior. - São Paulo: McGraw-Hill, 2006.