Otimização de Rotas de Entrega no Setor de FLV com Python: Um Estudo de Caso na Região dos Lagos Rio de Janeiro

Alan Victor Ferreira Modolo
Instituto Federal do Espírito Santo
E-mail: alanvfm2@hotmail.com

Denilton Macário de Paula
Instituto Federal do Espírito Santo
E-mail: denilton.macario@ifes.edu.br

1 PROBLEMA DE PESQUISA

Com a crescente preocupação com hábitos saudáveis e uma vida mais equilibrada, o setor de FLV (Frutas, Legumes e Verduras) vem assumindo um papel cada vez mais crucial dentro dos supermercados e na rotina das pessoas. Segundo a Organização Mundial da Saúde (2014), o consumo regular de FLV está associado a hábitos alimentares mais saudáveis. Ao incentivar as pessoas a consumirem mais frutas e verduras frescas, o setor de FLV contribui para uma alimentação equilibrada e variada, reduzindo assim o risco de obesidade e outras doenças relacionadas à dieta.

A distribuição de FLV no Brasil é desigual, com maior concentração nas regiões Sul e Sudeste. Segundo dados do IBGE (2010), o consumo per capita de FLV no Brasil ainda está abaixo do recomendado pela Organização Mundial da Saúde (2014), de 400g/dia.

Em resumo, o setor de FLV desempenha um papel vital na promoção da saúde, na sustentabilidade ambiental, no apoio à economia local e na diversificação das escolhas alimentares das pessoas. Por isso, é essencial valorizar e priorizar esse setor em supermercados e na vida cotidiana, sabendo também que o setor ocupa uma faixa que varia de 10 a 20% das vendas totais de um supermercado.

Dentro dessa importância, a Distribuidora Pomar teve sua origem em 1988, sob o nome de LeverFruti, com sede atual em Viana, Espírito Santo. Inicialmente, operava com duas lojas varejistas focadas na comercialização de hortifrútis. A partir de 1990, expandiu suas atividades para incluir a distribuição de FLV para outras redes varejistas, com volume médio de vendas de aproximadamente 3.200.000 Kg de FLV por mês. A empresa está presente nos principais grupos varejistas do país, como Assaí, Atacadão, Sam's Club, BH Supermercado, Gbarbosa, Cambuí Supermercados e Princesa Supermercados.

A Distribuidora Pomar busca promover soluções na distribuição de alimentos no segmento de Hortifruti Granjeiro, nas regiões do Espírito Santo, região dos lagos do Rio de Janeiro, sul da Bahia e interior de Minas Gerais (Vale do Rio Doce). No entanto, o transporte de FLV é um processo complexo que envolve diversos fatores,

como a distância entre os pontos de origem e destino, as condições de tráfego e as restrições de carga.

A região dos lagos do Rio de Janeiro representou 37,22% do faturamento bruto da Distribuidora Pomar em 2023, sendo uma região de suma importância para a empresa. Nesta região, estão situadas 6 lojas do grupo Atacadão, 16 lojas do grupo Princesa Supermercados e outras lojas de clientes menores.

O termo roteirização de veículos (VRP) deriva do inglês "routing" e refere-se ao processo de determinação de um ou mais roteiros, ou sequência de paradas, a serem percorridos por uma determinada frota de veículos. O objetivo é visitar um conjunto de pontos geometricamente dispersos em locais pré-determinados que necessitam de atendimento. A roteirização também visa a eliminação de falhas nos serviços e a redução de eventuais desperdícios que podem provocar aumento nos custos de distribuição (Cunha, 2000).

A roteirização e otimização de rotas é uma área essencial na logística, focada em determinar a sequência ideal de paradas e o trajeto mais eficiente para entregas ou coletas, considerando uma variedade de fatores. Esta área tem ganhado crescente importância devido aos seus inúmeros benefícios, que incluem a redução de custos, otimização da frota, melhoria na qualidade do serviço e diminuição do impacto ambiental (Arora e Chiang, 2015; Toth e Vigo, 2014).

Diversos métodos e algoritmos matemáticos podem ser utilizados para resolver problemas de VRP, cada um com suas vantagens e desvantagens. A escolha do método ideal depende das características específicas do problema, como o número de paradas, a capacidade dos veículos, as restrições de tempo e as condições do tráfego. Além disso, é importante analisar os recursos computacionais disponíveis e a precisão desejada (Toth e Vigo, 2014).

Para problemas de pequena a média escala, a programação linear e inteira (PLI) pode ser utilizada para encontrar soluções exatas, mas seu alto custo computacional a torna impraticável para problemas maiores (Bransetter & Eglese, 2005). Já os algoritmos genéticos são mais eficientes para problemas complexos e de grande escala, mas não garantem a solução ótima (Arora & Chiang, 2015). As heurísticas, como o método do vizinho mais próximo, são rápidas e fáceis de implementar, mas podem ter um desempenho inferior em problemas complexos (Toth & Vigo, 2014).

As metaheurísticas, por outro lado, são capazes de escapar de soluções insatisfatórias e encontrar boas aproximações da solução ótima, embora possam ter parâmetros complexos e tempo de execução longo (Bransetter & Eglese, 2005). Algoritmos de busca local, como o VND (Variable Neighborhood Descent) e o VNS (Variable Neighborhood Search), são simples de implementar e podem ser combinados com outras técnicas, mas podem ficar presos em soluções insatisfatórias em problemas grandes (Arora & Chiang, 2015). A programação

dinâmica, por sua vez, fornece soluções exatas para problemas específicos, como o problema do caixeiro-viajante (TSP) com poucas paradas, mas sua escalabilidade é limitada e exige alto poder computacional (Toth & Vigo, 2014).

Mesmo a Distribuidora Pomar sendo uma empresa de grande porte no segmento de distribuição de FLV, e tento várias opções de VRP disponíveis no mercado todo o processo de roteirização de transporte é realizado de forma manual, podendo demorar até 3 horas, apenas para agrupar as lojas da região dos lagos de maneira mais precisa e eficiente para o negócio. Sabendo ainda que o custo médio calculado do km rodado para a Distribuidora Pomar em 2023 foi de R$ 4,88, fica evidente que o custo de uma rota inadequada ou de veículos com baixa ocupação pode impactar significativamente a margem de lucro esperada para a viagem, prejudicando o andamento do negócio.

Além dos desafios específicos enfrentados pela Distribuidora Pomar, a otimização de rotas é um problema comum em todo o setor de distribuição de FLV. A complexidade logística associada à entrega de produtos perecíveis, como frutas, legumes e verduras, torna a roteirização uma etapa crítica que pode influenciar diretamente a qualidade dos produtos entregues, a satisfação do cliente e a rentabilidade das empresas.

Nesse sentido, este trabalho busca responder à seguinte questão: **Como desenvolver uma solução eficiente para a roteirização da Distribuidora Pomar em suas operações diárias utilizando a heurística do vizinho mais próximo em Python?**

2 OBJETIVOS

Objetivo Geral:

Analisar a utilização da heurística do Vizinho Mais Próximo na otimização de rotas para a distribuição de Frutas, Legumes e Verduras (FLV) na Região dos Lagos, com o intuito de aumentar a eficiência operacional e reduzir os custos de transporte.

Objetivos Específicos:

1. Reduzir significativamente o tempo necessário para a roteirização, passando de um processo manual de 3 horas para uma análise automatizada mais eficiente.
2. Automatizar o processo de roteirização, eliminando a necessidade de alocação exclusiva de um funcionário para essa tarefa.
3. Diminuir a distância total percorrida pelos veículos, aumentando a eficiência do uso da frota.
4. Alcançar uma economia substancial nos custos de transporte, contribuindo para uma gestão financeira mais eficiente.

3 PROCESSOS METODOLÓGICOS/MATERIAIS E MÉTODOS

Esta seção descreve os métodos, técnicas e ferramentas utilizados para abordar o problema de pesquisa relacionado à otimização de rotas para o setor de FLV. A escolha desses métodos foi fundamentada em considerações teóricas e práticas para garantir uma abordagem eficaz e robusta.

A decisão de utilizar algoritmos de otimização em Python no ambiente Jupyter Notebook para resolver o problema de roteirização foi motivada pela capacidade dessa linguagem de programação em lidar com grandes conjuntos de dados e pela ampla disponibilidade de bibliotecas especializadas em otimização. O ambiente Jupyter Notebook foi escolhido por sua facilidade de uso, interatividade e capacidade de documentar e visualizar o processo de desenvolvimento. Esses métodos foram considerados os mais adequados devido à sua flexibilidade, eficiência e capacidade de personalização para atender às necessidades específicas da Distribuidora Pomar.

Através da consideração de variáveis como diversos tipos de veículos, restrições de tempo, custos de entrega e outras particularidades do negócio, a escolha do Python se destaca por sua flexibilidade, escalabilidade e pela vasta gama de bibliotecas especializadas em otimização. Além disso, a linguagem é relativamente fácil de aprender, permitindo que a própria equipe da Distribuidora Pomar desenvolva e mantenha as soluções internamente. Ao investir em algoritmos de otimização em Python, a Distribuidora Pomar se posiciona na vanguarda da tecnologia, aprimorando sua logística, reduzindo custos e alcançando um novo patamar de eficiência e competitividade no mercado.

Para o desenvolvimento do código de roteirização foi utilizado o algoritmo do vizinho mais próximo (NNA) que se destaca como uma heurística eficiente para solucionar problemas de roteirização de veículos (VRP) em cenários de pequena a média escala. Sua simplicidade e rapidez de implementação o tornam uma ferramenta popular em diversas aplicações, como entregas, coletas e logística em geral (Arora & Chiang, 2015).

A jornada de roteirização de veículos começa em um ponto de partida aleatório ou no depósito, definindo o ponto inicial da rota. Em cada iteração, o ponto não visitado mais próximo do ponto atual é selecionado, guiando a rota de forma eficiente. O ponto selecionado é então incorporado à rota, atualizando a lista de pontos ainda a serem visitados. Essas etapas de seleção e adição se repetem até que todos os pontos sejam visitados, construindo assim a rota completa. Finalmente, a jornada é concluída com o retorno ao ponto de partida, ou depósito, completando o ciclo de entregas ou coletas.

Toth e Vigo (2014) destacam que o método do Vizinho Mais Próximo (NNA) é notável por sua simplicidade, sendo de fácil compreensão e implementação, mesmo para usuários com pouco conhecimento em programação. Bransetter e Eglese

(2005) apontam que a execução do algoritmo é rápida e eficiente, tornando-o ideal para cenários que exigem soluções velozes. Além disso, Arora e Chiang (2015) ressaltam que, em muitos casos, o NNA consegue encontrar soluções que estão próximas da solução ótima global, oferecendo uma boa aproximação. O NNA também apresenta um baixo custo computacional, o que o torna adequado para problemas em dispositivos com recursos limitados.

Arora e Chiang (2015) apontam que uma das principais desvantagens do método do Vizinho Mais Próximo (NNA) é a falta de otimização, pois o NNA não garante a solução ótima global, existindo a possibilidade de rotas melhores não serem exploradas. O algoritmo é sensível à ordem inicial dos pontos, o que pode influenciar significativamente na qualidade da solução final. Além disso, o NNA pode enfrentar dificuldades em lidar com restrições complexas, como tempo de viagem máximo, capacidade de carga dos veículos e horários de entrega, no entanto sua simplicidade, rapidez e baixo custo computacional o tornam uma opção vantajosa em diversos cenários.

Para o cálculo da distância entre dois pontos foi utilizado a fórmula de Haversine que calcula a distância entre dois pontos com base na extensão da linha reta entre os dois pontos em termos de longitude e latitude. A fórmula de Haversine é comumente usada em problemas de navegação porque pode fornecer a distância de grande círculo entre dois pontos na superfície do globo, independentemente da altura das colinas e da profundidade dos vales na superfície da Terra (Rezania e Febriyanti 2020).

Segundo Rofiq e Uzzy (2014) A fórmula é baseada na lei dos senos e é considerada uma das formas mais precisas de calcular a distância entre dois pontos na Terra. Ela utiliza a diferença ou magnitude das mudanças na latitude (Δlat) e longitude ($\Delta long$) de dois pontos coordenados em radianos.

$$\Delta lat = latitude2 - latitude1 \quad (1)$$

$$\Delta long = longitude2 - longitude1 \quad (2)$$

A partir das duas equações acima, calculamos a distância entre dois pontos usando a fórmula na Equação (3).

$$Distância = 2.R.arcsen\left(\sqrt{sen^2\left(\frac{\Delta lat}{2}\right) + \cos(lat1)\cos(lat2)sen^2\left(\frac{\Delta long}{2}\right)}\right) \quad (3)$$

Onde R na Equação (3) é o raio da Terra, que é de 6371 km.

Para o estudo em questão foi utilizado a Fórmula de Haversine em conjunto com o algoritmo do Vizinho Mais Próximo para resolver problemas de roteirização de veículos ou do Caixeiro Viajante de forma mais precisa, levando em consideração a distância real entre os pontos geográficos.

O algoritmo do Vizinho Mais Próximo é uma heurística popular para o problema do Caixeiro Viajante (Traveling Salesman Problem - TSP) e problemas

relacionados de roteamento de veículos. Ele é relativamente simples de implementar e pode produzir soluções razoáveis em um tempo computacional razoável, tornando-o uma escolha popular para problemas de roteirização. O Problema do Caixeiro Viajante (TSP) é, de longe, um dos problemas de otimização combinatória mais conhecidos e estudados extensivamente. Ele tem sido usado como um problema de referência para novos desenvolvimentos urbanos e de navegação há décadas. Ele pede para encontrar a rota mais curta (em termos de comprimento, tempo ou custo personalizado) que visita cada vértice em um determinado conjunto exatamente uma vez antes de retornar ao vértice inicial (Zia, Cakir e Seker, 2018).

4 COLETA E PREPARAÇÃO DE DADOS

Os dados utilizados para a simulação foram fornecidos pela Distribuidora Pomar e extraídos de um arquivo Excel contendo os endereços e demandas dos clientes da Região dos Lagos para o dia 1º de fevereiro de 2024. A coleta e preparação dos dados envolveram várias etapas importantes para garantir a qualidade e a integridade das informações utilizadas na implementação do modelo de roteirização.

A Distribuidora Pomar utiliza um sistema próprio, desenvolvido internamente por seus funcionários. Todos os clientes da distribuidora são cadastrados nesse sistema, que armazena detalhadamente os dados de endereço, coordenadas geográficas, dificuldades de acesso às lojas e condições de estoque. O sistema também registra um histórico de acessos, permitindo que os clientes realizem pedidos e especifiquem as quantidades desejadas de cada produto.

Para o setor de roteirização, o sistema gera automaticamente uma lista das lojas que fizeram pedidos naquele dia, juntamente com a quantidade de volumes solicitada por cada loja. Foi a partir desse sistema que extraímos um arquivo Excel com as informações dos clientes, incluindo latitude e longitude, endereço completo, nome da loja e nome fantasia, e a quantidade de volumes do pedido realizado no dia 1º de fevereiro de 2024.

Após a extração do arquivo Excel, foi necessário processar e transformar os dados para que pudessem ser utilizados no algoritmo de roteirização. Esse processo incluiu as seguintes etapas, primeiramente os dados foram lidos do arquivo Excel utilizando bibliotecas específicas para manipulação de planilhas, como o Pandas em Python. Após a leitura foi realizado a validação dos dados para verificar a presença de informações incompletas ou inconsistentes. Qualquer dado faltante ou erro identificado foi corrigido ou removido para assegurar a precisão das informações.

Os dados foram organizados em uma estrutura de dicionário, com cada cliente sendo uma chave e suas informações associadas (coordenadas e volumes) sendo os valores. Isso facilitou a manipulação e o processamento dos dados pelo algoritmo de roteirização.

Com essas etapas de coleta, processamento e transformação dos dados, asseguramos que as informações utilizadas pelo algoritmo de roteirização fossem precisas e completas. Essa preparação cuidadosa dos dados foi essencial para garantir a eficácia da otimização das rotas e a obtenção dos benefícios esperados em termos de redução de custos e aumento da eficiência operacional.

5 IMPLEMENTAÇÃO DO ALGORITMO

O objetivo do código é otimizar as rotas de entrega, levando em consideração a capacidade dos veículos e a localização geográfica dos clientes. O algoritmo implementado utiliza uma abordagem de vizinho mais próximo para determinar as rotas, e pode ser encontrado no apêndice deste trabalho.

1. Importação de Bibliotecas e Leitura de Dados:
   O código importa math para cálculos matemáticos e pandas para manipulação de dados. Os dados são carregados de um arquivo Excel que contém todas as informações necessárias.

2. Processamento dos Dados:
   Os dados dos clientes são extraídos linha a linha e armazenados em um dicionário, onde cada chave é o nome do cliente e o valor é um dicionário contendo coordenadas e volumes.

3. Definição de Parâmetros:
   É definido a capacidade máxima de carga de cada veículo e as coordenadas geográficas do depósito, que é o ponto inicial e final das rotas.

4. Função para Calcular Distância:
   A foi utilizado uma função que calcula a distância entre dois pontos geográficos usando a fórmula de Haversine, que é adequada para distâncias entre pontos na superfície da Terra.

5. Funções Auxiliares para Cálculo de Distância e Volume Total:
   Foi utilizado uma função que calcula a distância total percorrida em uma rota, incluindo a volta ao depósito e uma função que calcula o volume total dos transportado na rota criada.

   O cálculo da distância total começa com a inicialização de uma variável. As coordenadas do ponto inicial são definidas como sendo as latitudes e longitudes, que correspondem às coordenadas do depósito. Para cada cliente na rota, suas coordenadas são recuperadas. A distância entre o ponto atual e o cliente é calculada usando uma função para calcular essa distância, e essa distância é adicionada à distância total. As variáveis de latitude e longitude são

então atualizadas para as coordenadas do cliente atual. Após visitar todos os clientes na rota, a distância de volta ao depósito é calculada e adicionada à distância total.

6. Função para Encontrar Melhores Rotas:
   Utiliza o Algoritmo do Vizinho Mais Próximo para encontrar a melhor rota para cada veículo, considerando a capacidade de carga e a distância mínima entre os clientes.

   O processo de encontrar as melhores rotas para os veículos começa com a inicialização de algumas variáveis chave. A lista de rotas é criada para armazenar todas as rotas dos veículos, e uma lista de todos os clientes, obtida a partir das chaves do dicionário com as informações dos clientes.

   Enquanto houver clientes, o algoritmo continua a iterar. Para cada iteração, uma nova lista de rota é criada para armazenar a rota do veículo atual. A variável quer armazenar a soma dos volumes é inicializada em 0 para rastrear o volume total acumulado no veículo.

   O algoritmo então escolhe o próximo cliente para adicionar à rota. Duas variáveis auxiliares são usadas, uma armazena a menor distancia, que é usada para encontrar o cliente mais próximo, e outra, que armazena o próximo cliente a ser adicionado à rota.

7. Função para Dividir Clientes:
   A função divide clientes cujo volume de pedido excede a capacidade do veículo em partes menores, garantindo que cada parte esteja dentro da capacidade permitida.

6 RESULTADOS E DISCUSSÃO

Iniciando a etapa de teste do código implementado, foi selecionado um dia de carregamento que abrangeu um total de 15 lojas distribuídas pela região dos lagos no Rio de Janeiro, conforme ilustrado na Tabela 1. Durante a análise dos dados, observou-se uma notável variação na quantidade de volumes dos pedidos das lojas, não seguindo um padrão específico. Destaca-se que a loja Princesa de Búzios 02 registrou o maior pedido, totalizando 64,6 volumes. Essa diversidade de volumes apresenta um desafio significativo para o algoritmo de roteirização, exigindo uma capacidade de adaptação eficiente para otimizar as rotas de entrega e maximizar a utilização dos veículos disponíveis.

Tabela 1: Lojas e volumes carregados (autor)

LOJAS	VOLUMES
MAZA BOX	36,46
PRINCESA MARICA 02	24,31
PRINCESA MARICA 01	33,76
PRINCESA IGUABA GRANDE	58,30
PRINCESA BARRA DE SÃO JOÃO	9,60
PRINCESA RIO DAS OSTRAS	30,18
PRINCESA CABO FRIO 01	22,99
PRINCESA BUZIOS 03	37,56
PRINCESA BUZIOS 02	64,56
PRINCESA BUZIOS 01	29,89
PRINCESA CABO FRIO 02	16,50
PRINCESA CABO FRIO 03	32,13
PRINCESA ARRAIAL 03	57,60
PRINCESA ARRAIAL 02	30,38
PRINCESA ARRAIAL 01	31,00

A implementação do algoritmo de roteirização de entregas consistiu em uma série de etapas meticulosamente planejadas. Inicialmente, os dados dos clientes e a capacidade dos veículos foram preparados para análise. Cada cliente foi devidamente registrado com suas coordenadas geográficas e o volume dos pedidos, enquanto a capacidade de carga dos veículos foi estabelecida para orientar a distribuição eficiente das mercadorias.

A determinação da capacidade dos veículos foi realizada levando em consideração o volume dos produtos, uma escolha estratégica da empresa devido ao espaço ocupado pelas caixas dentro dos veículos. Em muitos casos, o limite máximo de peso do carregamento não é atingido, mas o espaço disponível no baú do veículo é totalmente ocupado. Essa abordagem reflete a realidade operacional da empresa e tem impacto direto na eficiência da distribuição das mercadorias. Ao considerar o volume como critério principal para a capacidade dos veículos, o algoritmo de roteirização foi capaz de otimizar as rotas levando em conta a utilização máxima do espaço disponível, mesmo quando a capacidade de peso não é completamente utilizada.

Uma etapa crucial do código foi a divisão de clientes cujos pedidos excediam a capacidade dos veículos (300,00 Volumes). Para lidar com essa situação, foi desenvolvida uma função específica que dividi esses clientes em partes menores, garantindo que o algoritmo de roteirização pudesse acomodá-los de maneira adequada durante a geração das rotas. Essa etapa do código desempenha um papel crucial na garantia de que nenhum pedido exceda a capacidade dos veículos, permitindo uma distribuição eficiente das mercadorias.

Para analisar a capacidade dos veículos, selecionamos aleatoriamente cinco clientes: PRINCESA BARRA DE SÃO JOÃO, PRINCESA RIO DAS OSTRAS, PRINCESA CABO FRIO 01, PRINCESA BUZIOS 03 e PRINCESA BUZIOS 02, cada um com seu volume inicial conforme ilustrado na Tabela 1. A roteirização resultou em um volume total de 164,88 volumes (Tabela 2), o que ficou abaixo da capacidade estipulada do veículo, ao analisar tais rotas o resultado gerado pelo código com a ordem de entrega pode ser visto na Tabela 3. No entanto, para demonstrar a capacidade do código em lidar com excessos de capacidade, supomos que a Loja PRINCESA RIO DAS OSTRAS tivesse um pedido de 350,00 volumes. Como mostrado em destaque na Tabela 4, o código automaticamente dividiu esse cliente em duas partes e alocou essas partes em diferentes veículos para garantir que a capacidade máxima dos veículos não fosse ultrapassada. Esse exemplo ilustra a flexibilidade do algoritmo em lidar com diferentes cenários logísticos, garantindo uma distribuição eficiente das mercadorias dentro das restrições de capacidade dos veículos.

Tabela 2: Rota com lojas fictícia para análise de capacidade (autor)

LOJAS	VOLUMES
PRINCESA BARRA DE SÃO JOÃO	9,60
PRINCESA RIO DAS OSTRAS	30,18
PRINCESA CABO FRIO 01	22,99
PRINCESA BUZIOS 03	37,56
PRINCESA BUZIOS 02	64,56
Volumes Total	**164,89**

Tabela 3 - Resultado do código para rota fictícia (autor)

VEÍCULO	LOJAS	ENTREGA	VOLUMES
Veículo 1	PRINCESA RIO DAS OSTRAS	1	30,18
Veículo 1	PRINCESA BARRA DE SÃO JOÃO	2	9,60
Veículo 1	PRINCESA BUZIOS 03	3	37,56
Veículo 1	PRINCESA BUZIOS 02	4	64,56
Veículo 1	PRINCESA CABO FRIO 01	5	22,99
KM Percorrido	**670,91**	**Total de Volumes**	**164,89**

Tabela 4 - Divisão do cliente para evitar excesso de capacidade (autor)

VEÍCULO	LOJAS	ENTREGA	VOLUMES
Veículo 1	PRINCESA RIO DAS OSTRAS PARTE 1	1	175,00
Veículo 1	PRINCESA BARRA DE SÃO JOÃO	2	9,60
Veículo 1	PRINCESA BUZIOS 03	3	37,56
Veículo 1	PRINCESA BUZIOS 02	4	64,56
KM Percorrido	**636,82**	**Total de Volumes**	**286,71**

Veículo 2	PRINCESA RIO DAS OSTRAS PARTE 2	1	175,00
Veículo 2	PRINCESA CABO FRIO 01	2	22,99
KM Percorrido	**663,81**	**Total de Volumes**	**197,99**

Outro aspecto importante considerado na geração de rotas é a priorização de entregas para clientes específicos, seja devido a acordos comerciais ou questões estratégicas da empresa. No contexto dos clientes de teste apresentados na Tabela 2, decidimos atribuir prioridade ao cliente PRINCESA BARRA DE SÃO JOÃO, determinando que ele receba a primeira entrega do veículo que transportará a carga daquele dia. Ao definir esse cliente como a primeira entrega, observamos que ele automaticamente se torna a primeira parada na rota, seguido pelas demais entregas do veículo, como pode ser visto na Tabela 5. Essa abordagem permite que a empresa atenda a compromissos comerciais importantes e mantenha a eficiência operacional ao mesmo tempo, garantindo que clientes prioritários recebam suas entregas de forma rápida e confiável, no entanto podemos notar que houve um aumento na distância total percorrida pelo veículo, conforme discriminado na Tabela 6.

Tabela 5 - Prioridade de entrega (autor)

VEÍCULO	LOJAS	ENTREGA	VOLUMES
Veículo 1	PRINCESA BARRA DE SÃO JOÃO	1	9,60
Veículo 1	PRINCESA RIO DAS OSTRAS	2	30,18
Veículo 1	PRINCESA BUZIOS 03	3	37,56
Veículo 1	PRINCESA BUZIOS 02	4	64,56
Veículo 1	PRINCESA CABO FRIO 01	5	22,99
KM Percorrido	**685,11**	**Total de Volumes**	**164,89**

Tabela 6: Diferença de KM percorrido devido a prioridade de entrega (autor)

Rota com Prioridade de Entrega		**Rota sem Prioridade de Entrega**	
LOJAS	**VOLUMES**	**LOJAS**	**VOLUMES**
PRINCESA BARRA DE SÃO JOÃO	9,60	PRINCESA RIO DAS OSTRAS	30,18
PRINCESA RIO DAS OSTRAS	30,18	PRINCESA BARRA DE SÃO JOÃO	9,60
PRINCESA BUZIOS 03	37,56	PRINCESA BUZIOS 03	37,56
PRINCESA BUZIOS 02	64,56	PRINCESA BUZIOS 02	64,56
PRINCESA CABO FRIO 01	22,99	PRINCESA CABO FRIO 01	22,99
KM Percorrido	**685,11**	**KM Percorrido**	**670,91**

Um dos pontos cruciais do código foi a aplicação do algoritmo do Vizinho Mais Próximo para determinar as melhores rotas para cada veículo. Esse processo envolveu uma iteração cuidadosa sobre os clientes não visitados, selecionando, em cada passo, o cliente mais próximo do atual até que todos os clientes fossem

visitados. Essa abordagem permitiu uma otimização eficiente das rotas, levando em consideração a proximidade geográfica entre os pontos de entrega.

No dia de carregamento selecionado para verificar o código, foram alocados três veículos para a região dos lagos, conforme ilustrado na Tabela 7. O veículo 1 partiu com um total de 192,61 volumes, o veículo 2 transportou um total de 155 volumes e o veículo 3 transportou um total de 167,61 volumes, podemos observar também na Tabela 7 a ordem de entrega de cada um dos veículos, onde o 1 representa a primeira entrega realizada e assim por diante.

Tabela 7: Rotas reais feitas para Região dos Lagos (autor)

VEÍCULO	LOJAS	ENTREGA	VOLUMES
Veículo 1	PRINCESA RIO DAS OSTRAS	1	30,18
Veículo 1	PRINCESA BARRA DE SÃO JOÃO	2	9,60
Veículo 1	PRINCESA IGUABA GRANDE	3	58,30
Veículo 1	PRINCESA MARICA 01	4	33,76
Veículo 1	PRINCESA MARICA 02	5	24,31
Veículo 1	MAZA BOX	6	36,46
		Total de Volumes	**192,61**
Veículo 2	PRINCESA BUZIOS 01	1	29,89
Veículo 2	PRINCESA BUZIOS 02	2	64,56
Veículo 2	PRINCESA BUZIOS 03	3	37,56
Veículo 2	PRINCESA CABO FRIO 01	4	22,99
		Total de Volumes	**155,00**
Veículo 3	PRINCESA ARRAIAL 01	1	31,00
Veículo 3	PRINCESA ARRAIAL 02	2	30,38
Veículo 3	PRINCESA ARRAIAL 03	3	57,60
Veículo 3	PRINCESA CABO FRIO 03	4	32,13
Veículo 3	PRINCESA CABO FRIO 02	5	16,50
		Total de Volumes	**167,61**

A análise minuciosa das rotas geradas pelo código em comparação com as rotas feitas manualmente revelou resultados interessantes. Para o veículo 1, o código manteve a mesma rota conforme a rota manual, como evidenciado na Tabela 8. No entanto, para o veículo 2 (Tabela 8), houve uma alteração na ordem de entrega, com o cliente PRINCESA DE BUZIOS 03 sendo atendido primeiro antes dos demais clientes, o que resultou em uma variação insignificante de 0,6 km na distância percorrida. Essa variação pode ser considerada insignificante devido à proximidade das lojas de PRINCESA DE BUZIOS. Já para o veículo 3, o código gerou a rota de maneira inversa à rota manual, mas mantendo a sequência entre as lojas, como demonstrado na Tabela 8.

Essas observações destacam a eficácia do código em replicar as rotas manualmente definidas, garantindo uma distribuição eficiente das entregas na região dos lagos a diferença entre o realizado e o gerado pelo algoritmo podem ser visualizadas na Tabela 8.

Tabela 8: Diferença de rotas entre realizado e simulado

Realizado		**Simulado**	
LOJAS	**ENTREGA**	**LOJAS**	**ENTREGA**
PRINCESA RIO DAS OSTRAS	1	PRINCESA RIO DAS OSTRAS	1
PRINCESA BARRA DE SÃO JOÃO	2	PRINCESA BARRA DE SÃO JOÃO	2
PRINCESA IGUABA GRANDE	3	PRINCESA IGUABA GRANDE	3
PRINCESA MARICA 01	4	PRINCESA MARICA 01	4
PRINCESA MARICA 02	5	PRINCESA MARICA 02	5
MAZA BOX	6	MAZA BOX	6
KM Percorrido	**795,60**	**KM Percorrido**	**795,60**
PRINCESA BUZIOS 01	1	PRINCESA BUZIOS 03	1
PRINCESA BUZIOS 02	2	PRINCESA BUZIOS 01	2
PRINCESA BUZIOS 03	3	PRINCESA BUZIOS 02	3
PRINCESA CABO FRIO 01	4	PRINCESA CABO FRIO 01	4
KM Percorrido	**661,81**	**KM Percorrido**	**661,21**
PRINCESA ARRAIAL 01	1	PRINCESA CABO FRIO 02	1
PRINCESA ARRAIAL 02	2	PRINCESA CABO FRIO 03	2
PRINCESA ARRAIAL 03	3	PRINCESA ARRAIAL 03	3
PRINCESA CABO FRIO 03	4	PRINCESA ARRAIAL 02	4
PRINCESA CABO FRIO 02	5	PRINCESA ARRAIAL 01	5
KM Percorrido	**682,85**	**KM Percorrido**	**682,85**

Após a geração das rotas para cada veículo, o código calculou meticulosamente a distância total percorrida e o peso total transportado em cada rota. Essas métricas desempenharam um papel fundamental na avaliação da eficácia das rotas geradas e sua capacidade de atender às demandas logísticas de maneira satisfatória.

Uma avaliação adicional foi realizada ao adicionar todas as lojas do dia no código e permitir que ele fizesse toda a roteirização automaticamente. Como mostrado na Tabela 10, o código criou apenas dois veículos, ambos satisfazendo as condições de capacidade máxima. Em comparação com as rotas manualmente definidas como mostrado na Tabela 9, que resultaram em uma rota com 3 veículos e percorrendo um total de aproximadamente 2139,66 km, as rotas geradas pelo código economizaram significativamente em distância, totalizando aproximadamente 1656,14 km (Tabela 10). Essa economia de 483,52 km equivale a uma economia financeira de R$ 2359,58, considerando o custo do km para a empresa de R$ 4,88. Esses resultados demonstram claramente os benefícios da utilização do algoritmo

de roteirização implementado, tanto em termos de eficiência operacional quanto de economia financeira para a empresa.

Tabela 9: Rotas Realizadas de Forma Manual

Realizado			
VEÍCULO	**LOJAS**	**ENTREGA**	**VOLUMES**
Veículo 1	PRINCESA RIO DAS OSTRAS	1	30,18
Veículo 1	PRINCESA BARRA DE SÃO JOÃO	2	9,60
Veículo 1	PRINCESA IGUABA GRANDE	3	58,30
Veículo 1	PRINCESA MARICA 01	4	33,76
Veículo 1	PRINCESA MARICA 02	5	24,31
Veículo 1	MAZA BOX	6	36,46
KM Percorrido	**795,6**	**Total de Volumes**	**192,61**
Veículo 2	PRINCESA BUZIOS 01	1	29,89
Veículo 2	PRINCESA BUZIOS 02	2	64,56
Veículo 2	PRINCESA BUZIOS 03	3	37,56
Veículo 2	PRINCESA CABO FRIO 01	4	22,99
KM Percorrido	**661,81**	**Total de Volumes**	**155,00**
Veículo 3	PRINCESA ARRAIAL 01	1	31,00
Veículo 3	PRINCESA ARRAIAL 02	2	30,38
Veículo 3	PRINCESA ARRAIAL 03	3	57,60
Veículo 3	PRINCESA CABO FRIO 03	4	32,13
Veículo 3	PRINCESA CABO FRIO 02	5	16,50
KM Percorrido	**682,85**	**Total de Volumes**	**167,61**
Total KM Percorrido	**2140,26**		

Tabela 10: Rotas Realizadas pelo Algoritmo

Simulado			
VEÍCULO	**LOJAS**	**ENTREGA**	**VOLUMES**
Veículo 1	PRINCESA RIO DAS OSTRAS	1	30,18
Veículo 1	PRINCESA BARRA DE SÃO JOÃO	2	9,60
Veículo 1	PRINCESA BUZIOS 03	3	37,56
Veículo 1	PRINCESA BUZIOS 01	4	29,89
Veículo 1	PRINCESA BUZIOS 02	5	64,56
Veículo 1	PRINCESA CABO FRIO 03	6	32,13
Veículo 1	PRINCESA CABO FRIO 01	7	22,99
Veículo 1	PRINCESA CABO FRIO 02	8	16,50
Veículo 1	PRINCESA ARRAIAL 01	9	31,00
Veículo 1	PRINCESA MARICA 02	10	24,31
KM Percorrido	**819,61**	**Total de Volumes**	**298,71**
Veículo 2	PRINCESA IGUABA GRANDE	1	58,30
Veículo 2	PRINCESA ARRAIAL 02	2	30,38
Veículo 2	PRINCESA ARRAIAL 03	3	57,60

Veículo 2	PRINCESA MARICA 01	4	33,76
Veículo 2	MAZA BOX	5	36,46
KM Percorrido	**836,53**	**Total de Volumes**	**216,50**
Total KM Percorrido	**1656,14**		

Apesar dos resultados obtidos devemos lembrar que o Algoritmo do Vizinho Mais Próximo não garante a solução ótima. Ele toma decisões que podem levar a rotas que não sejam ótimas, pois sempre escolhe o cliente mais próximo sem considerar o impacto das decisões futuras. O algoritmo também tem uma dependência da ordem inicial, onde o ponto de partida e a ordem de visita dos clientes podem influenciar significativamente o resultado. Além disso o algoritmo é simples e não leva em conta restrições complexas, como janelas de tempo, diferentes capacidades de veículos, preferências de clientes, entre outros.

O algoritmo usa volume como métrica de capacidade, mas em cenários reais, tanto volume quanto peso podem ser restrições importantes e desconsiderar uma dessas restrições pode levar a soluções impraticáveis. Embora o Algoritmo do Vizinho Mais Próximo seja mais eficiente do que métodos exaustivos (como permutações), ele ainda pode ser ineficiente para grandes conjuntos de dados.

Por fim, os resultados das rotas otimizadas foram apresentados de forma clara e concisa. Cada rota foi detalhadamente descrita, incluindo os clientes visitados, a distância total percorrida e o volume total transportado. Essa análise detalhada permitiu uma avaliação criteriosa das soluções encontradas pelo algoritmo, fornecendo insights valiosos para aprimoramentos futuros e aplicações práticas no contexto da logística de entregas.

7 CONCLUSÕES E CONSIDERAÇÕES FINAIS

Após uma análise abrangente do código implementado para a roteirização de entregas, é possível concluir que o sistema apresenta uma série de vantagens e benefícios significativos para a empresa. Primeiramente, destacou-se a eficiência operacional do código, capaz de gerar rotas otimizadas para os veículos de entrega, considerando não apenas a capacidade de carga de cada veículo, mas também as prioridades de entrega dos clientes e as restrições logísticas pertinentes. Além disso, observou-se uma economia substancial de recursos, uma vez que a utilização do algoritmo resultou em uma redução significativa na distância total percorrida pelos veículos, consequentemente reduzindo os custos operacionais associados ao transporte.

A flexibilidade e adaptabilidade do código também foram pontos destacados, especialmente na capacidade de lidar com cenários logísticos complexos, como a divisão de clientes com pedidos que excedem a capacidade dos veículos e a priorização de entregas para clientes específicos.

A análise baseada em dados, por meio das métricas calculadas pelo código, proporcionou uma base sólida para avaliar a eficácia das rotas geradas e para embasar decisões estratégicas. Por fim, a comparação entre as rotas geradas pelo código e as rotas manualmente definidas permitiu identificar áreas de melhoria e refinamento, proporcionando insights valiosos para futuras otimizações do sistema.

O algoritmo do vizinho mais próximo (NNA) se consolida como uma ferramenta valiosa para solucionar problemas de roteirização de veículos (VRP) em pequena a média escala. Sua simplicidade, rapidez e baixo custo computacional o tornam uma opção vantajosa em diversos cenários. É importante ressaltar que, para problemas mais complexos ou que exigem otimização precisa, outras técnicas como programação linear ou metaheurísticas podem ser mais adequadas. A escolha do método ideal depende das características específicas do problema, como tamanho, complexidade, recursos computacionais disponíveis e a precisão desejada.

Em suma, o código de roteirização demonstrou ser uma ferramenta poderosa para melhorar a eficiência operacional, reduzir custos e atender às demandas logísticas de forma mais eficaz. Mediante um contínuo processo de refinamento e aprimoramento, a empresa poderá maximizar os benefícios da otimização de rotas no longo prazo.

AGRADECIMENTOS

Gostaria de expressar meus sinceros agradecimentos a todos que contribuíram para a realização deste trabalho. Primeiramente, agradeço ao professor Denilton Macário de Paula pela orientação e apoio ao longo do desenvolvimento deste projeto. Agradeço também à instituição IFES, que proporcionou o ambiente e os recursos necessários para minha formação e ao Alexandre Gegeski pela ajuda na implementação do código.

Aos meus pais, Jandira e Almir, pelo apoio incondicional e incentivo constante. Ao meu irmão Daniel e minha cunhada Aline, pelo companheirismo e suporte em todos os momentos.

Um agradecimento mais que especial à minha esposa Jinse. Sua paciência, compreensão, amor e apoio foram fundamentais para a conclusão deste trabalho. Jinse, você esteve ao meu lado em todos os desafios, oferecendo encorajamento nos momentos difíceis e celebrando cada pequena conquista comigo. Este trabalho não seria possível sem você, e sou eternamente grato por tudo o que fez por mim.

E, finalmente, agradeço a Deus, pela força e sabedoria concedidas ao longo desta jornada.

A todos, meu muito obrigado.

DECLARAÇÃO DE USO DE TECNOLOGIAS AUXILIADAS POR INTELIGÊNCIA ARTIFICIAL

Eu, Alan Victor Ferreira Modolo, aluno do curso de Engenharia de Produção com ênfase em Ciência de Dados do Instituto Federal do Espírito Santo (IFES), declaro que utilizei tecnologias assistidas por inteligência artificial (ChatGPT), para auxiliar na realização deste trabalho de conclusão de curso.

A ferramenta de inteligência artificial foi empregada de forma ética e responsável, seguindo as diretrizes e regulamentos estabelecidos pela instituição, para revisar o texto, garantindo coesão e correção gramatical.

REFERÊNCIAS

AGRAMANISTI AZDY, R.; DARNIS, F. Use of Haversine Formula in Finding Distance Between Temporary Shelter and Waste End Processing Sites. **Journal of Physics: Conference Series**, v. 1500, n. 1, p. 012104, 1 abr. 2020.

Arora, S., & Chiang, M. **Transportation planning and optimization**. Academic Press, 2015.

Bransetter, S., & Eglese, P. **Routing problems in transportation**. Springer, 2005.

CIENCIADEDADOSBRASIL.COM.BR. **NumPy e Pandas: Guia Prático**. Disponível em: <https://cienciadedadosbrasil.com.br/numpy-e-pandas-guia-pratico/#google_vignette>. Acesso em: 10 jan. 2024.

CUNHA, C. B. DA. Aspectos práticos da aplicação de modelos de roteirização de veículos a problemas reais. **TRANSPORTES**, v. 8, n. 2, 2 jul. 2000.

Distribuidora Pomar. Disponível em: <https://distribuidorapomar.com>. Acesso em: 02 fev. 2024.

Instituto Brasileiro de Geografia e Estatística (IBGE). Pesquisa de Orçamentos Familiares 2008-2009: Mais de 90% da população comem poucas frutas, legumes e verduras. Rio de Janeiro: **IBGE**; 2010.

M. Rofiq and R. F. Uzzy, “Penentuan Jalur Terpendek Menuju Cafe di Kota Malang Menggunakan Metode Bellman-Ford dengan Location Based Service Berbasis Android**,” J. Ilm. Teknol. dan Inf. ASIA**, vol. 8, no. 2, pp. 49–64, 2014.

MENDIS, S.; DAVIS, S.; NORRVING, B. Organizational Update: The World Health Organization Global Status Report on Noncommunicable Diseases 2014; One More Landmark Step in the Combat Against Stroke and Vascular Disease. **Stroke**, v. 46, n. 5, p. e121–e122, 14 abr. 2015.

TOTH, P.; VIGO, D. (EDS.). **Vehicle Routing**. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2014.

ZIA, M.; CAKIR, Z.; SEKER, D. Spatial Transformation of Equality – Generalized Travelling Salesman Problem to Travelling Salesman Problem. **ISPRS International Journal of Geo-Information**, v. 7, n. 3, p. 115, 15 mar. 2018.

APÊNDICE

```
from itertools import permutations
import math
import pandas as pd

# Carregar os dados do arquivo Excel
arquivo = 'C:\\Users\\Alan\\Documents\\Alan\\Produção\\TFC\\Codigo\\dados_sistema.xlsx'
df = pd.read_excel(arquivo)

# Processar os dados
dados_clientes = {}
for index, row in df.iterrows():
    cliente = row['Cliente']
    volumes = row['Volumes']
    lat = row['latitudecliente']
    lon = row['longitudecliente']
    dados_clientes[cliente] = {'coordenadas': (lat, lon), 'volumes': volumes}
print (dados_clientes)

# Capacidade de carga de cada veículo em Volume
capacidade_veiculos = 300

# Coordenadas do depósito (ponto inicial e final das rotas)
deposito = (-20.378388, -40.309096)

# Função para calcular a distância entre dois pontos geográficos usando a fórmula de Haversine
def calcular_distancia(lat1, lon1, lat2, lon2):
```

```
raio_terra = 6371.0 # Raio médio da Terra em km
      # Converter graus para radianos
      lat1_rad = math.radians(lat1)
      lon1_rad = math.radians(lon1)
      lat2_rad = math.radians(lat2)
      lon2_rad = math.radians(lon2)
      # Diferença das longitudes e das latitudes
      delta_lon = lon2_rad - lon1_rad
      delta_lat = lat2_rad - lat1_rad
      # Fórmula de Haversine
      a = math.sin(delta_lat / 2)**2 + math.cos(lat1_rad) *
math.cos(lat2_rad) * math.sin(delta_lon / 2)**2
      c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
      distancia = raio_terra * c
return distancia

# Função para calcular a distância total de uma rota
def calcular_distancia_total(rota):
distancia_total = 0
lat1, lon1 = deposito # Começar do depósito
for cliente in rota:
      lat2, lon2 = dados_clientes[cliente]['coordenadas']
      distancia_total += calcular_distancia(lat1, lon1, lat2, lon2)
      lat1, lon1 = lat2, lon2
distancia_total += calcular_distancia(lat1, lon1, deposito[0],
deposito[1]) # Voltar ao depósito
return distancia_total

# Função para calcular o peso total de uma rota
def calcular_peso_total(rota):
```

```
    peso_total = sum(dados_clientes[cliente]['volumes'] for cliente
in rota)
return peso_total


# Função para encontrar a melhor rota para cada veículo usando o
Algoritmo do Vizinho Mais Próximo
def encontrar_melhores_rotas(dados_clientes, capacidade_veiculo):
    rotas_veiculos = []
    clientes = list(dados_clientes.keys())
    while clientes:
        rota_veiculo = []
        peso_acumulado = 0
        cliente_atual = None
        while clientes:
            menor_distancia = float('inf')
            proximo_cliente = None
            for cliente in clientes:
                if cliente_atual is None:
                    lat1, lon1 = deposito
                else:
                    lat1,               lon1               =
                dados_clientes[cliente_atual]['coordenadas']
                    lat2,                   lon2                   =
                dados_clientes[cliente]['coordenadas']
                    distancia   =   calcular_distancia(lat1,   lon1,
                lat2, lon2)
                if distancia < menor_distancia and (peso_acumulado +
                dados_clientes[cliente]['volumes'])                <=
                capacidade_veiculo:
                    menor_distancia = distancia
                    proximo_cliente = cliente
            if proximo_cliente:
```

```
                    rota_veiculo.append(proximo_cliente)
                    clientes.remove(proximo_cliente)
                    peso_acumulado += dados_clientes[proximo_cliente]['volumes']
                    cliente_atual = proximo_cliente
               else:
                    break
          if rota_veiculo:
               rotas_veiculos.append(rota_veiculo)
          else:
          break
     return rotas_veiculos

# Função para lidar com clientes que têm peso maior que a capacidade do veículo
def dividir_clientes(dados_clientes, capacidade_veiculo):
     novos_dados_clientes = {}
     for cliente, dados in dados_clientes.items():
          if dados['volumes'] > capacidade_veiculo:
               partes = math.ceil(dados['volumes'] / capacidade_veiculo)
               peso_parte = dados['volumes'] / partes
               for i in range(partes):
                    novos_dados_clientes[f"{cliente}_parte_{i+1}"] = {'coordenadas': dados['coordenadas'], 'volumes': peso_parte}
          else:
               novos_dados_clientes[cliente] = dados
     return novos_dados_clientes

# Dividir clientes que têm peso maior que a capacidade do veículo
```

```
dados_clientes = dividir_clientes(dados_clientes, capacidade_veiculos)

# Encontrar a melhor rota para cada veículo
rotas_veiculos = encontrar_melhores_rotas(dados_clientes, capacidade_veiculos)

# Ordenar os clientes de acordo com a prioridade
clientes_prioritarios = []
for veiculo in rotas_veiculos:
    veiculo.sort(key=lambda cliente: clientes_prioritarios.index(cliente)
    if cliente in clientes_prioritarios
    else float('inf'))

# Imprimir as rotas para cada veículo
if rotas_veiculos:
print("Rotas para cada veículo:")
for idx, rota_veiculo in enumerate(rotas_veiculos):
print(f"Veículo {idx + 1}:")
distancia_total = calcular_distancia_total(rota_veiculo)
peso_total = calcular_peso_total(rota_veiculo)
print(" - Clientes:")
for cliente in rota_veiculo:
print(f" - {cliente}: {dados_clientes[cliente]['coordenadas']} (volumes: {dados_clientes[cliente]['volumes']}
Volume)")
print(f" - Distância Total: {distancia_total:.2f} km | Volume Total: {peso_total:.2f} Volume\n")
else:
print("Não foi possível encontrar rotas.")
```

ALAN VICTOR FERREIRA MODOLO

OTIMIZAÇÃO DE ROTAS DE ENTREGA NO SETOR DE FLV COM PYTHON: UM ESTUDO DE CASO NA REGIÃO DO LAGOS RIO DE JANEIRO

Trabalho Final de Curso apresentado ao Curso de Pós-Graduação Lato Sensu em Engenharia de Produção com ênfase em Ciência de Dados do Instituto Federal do Espírito Santo, como requisito parcial para obtenção de título de Especialista em Engenharia de Produção com ênfase em Ciência de Dados.

Aprovado em 11 de julho de 2024

COMISSÃO EXAMINADORA

Prof. Me. Denilton Macário de Paula
Instituto Federal do Espírito
Santo Orientador

Prof. Dr. Tiago José Menezes Gonçalves
Instituto Federal do Espírito Santo

Prof. Me. Wilson Carminati Benaquio
Instituto Federal do Espírito Santo

FOLHA DE APROVAÇÃO-TCC Nº Folha de Aprovação TFC - Alan Victor Ferreira Modo/2024 - VIA-CTLG (11.02.19.02.01.04.04)
(Nº do Documento: 1)

(Nº do Protocolo: NÃO PROTOCOLADO)

(Assinado digitalmente em 26/07/2024 10:39)
DENILTON MACARIO DE PAULA
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
VIA-CTLG (11.02.19.02.01.04.04)
Matrícula: 2413270

(Assinado digitalmente em 26/07/2024 18:43)
TIAGO JOSE MENEZES GONCALVES
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAR-CCEP (11.02.19.01.08.03.10)
Matrícula: 2073974

(Assinado digitalmente em 29/07/2024 09:30)
WILSON CARMINATTI BENAQUIO
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
VIA-CCTL (11.02.19.02.01.04.05)
Matrícula: 2177619

*Visualize o documento original em https://sipac.ifes.edu.br/documentos/ informando seu número: **1**, ano: **2024**, tipo: **FOLHA DE APROVAÇÃO-TCC**, data de emissão: **26/07/2024** e o código de verificação: **a2dbf1b021***