

**UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"**

**FACULDADE DE CIÊNCIAS - CAMPUS BAURU**

**DEPARTAMENTO DE COMPUTAÇÃO**

**BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**LUIZ FERNANDO MERLI DE OLIVEIRA SEMENTILLE**

**UMA APLICAÇÃO WEB PARA ANÁLISE COMPARATIVA DE  
SELEÇÃO DE CARACTERÍSTICAS BASEADAS EM  
META-HEURÍSTICAS**

**BAURU**

**Janeiro/2023**

LUIZ FERNANDO MERLI DE OLIVEIRA SEMENTILLE

**UMA APLICAÇÃO WEB PARA ANÁLISE COMPARATIVA DE  
SELEÇÃO DE CARACTERÍSTICAS BASEADAS EM  
META-HEURÍSTICAS**

Trabalho de Conclusão de Curso do Curso de  
Bacharelado em Ciência da Computação da Uni-  
versidade Estadual Paulista “Júlio de Mesquita  
Filho”, Faculdade de Ciências, Campus Bauru.  
Orientador: Prof. Associado João Paulo Papa  
Coorientador: Prof. Dr. Douglas Rodrigues

BAURU  
Janeiro/2023

S471a	<p>Sementille, Luiz Fernando Merli de Oliveira</p> <p>Uma aplicação web para análise comparativa de seleção de características baseadas em meta-heurísticas / Luiz Fernando Merli de Oliveira Sementille. -- Bauru, 2023</p> <p>77 p. : il.</p> <p>Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências, Bauru</p> <p>Orientador: João Paulo Papa</p> <p>Coorientador: Douglas Rodrigues</p> <p>1. Aplicação Web. 2. Seleção de Características. 3. Meta-heurística. I. Título.</p>
-------	---

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências, Bauru.

Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Luiz Fernando Merli de Oliveira Sementille

## **Uma aplicação web para análise comparativa de seleção de características baseadas em meta-heurísticas**

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Faculdade de Ciências, Campus Bauru.

Banca Examinadora

---

**Prof. Associado João Paulo Papa**

Orientador

Universidade Estadual Paulista “Júlio de Mesquita Filho”

Faculdade de Ciências

Bacharelado em Ciência da Computação

---

**Profa. Dra. Simone das Graças Domingues Prado**

Universidade Estadual Paulista “Júlio de Mesquita Filho”

Faculdade de Ciências

Bacharelado em Ciência da Computação

---

**Prof. Dr. Clayton Reginaldo Pereira**

Universidade Estadual Paulista “Júlio de Mesquita Filho”

Faculdade de Ciências

Bacharelado em Ciência da Computação

Bauru, \_\_\_\_\_ de \_\_\_\_\_ de \_\_\_\_\_.

*Dedico este trabalho a toda a minha família, a minha cachorrinha Bibi, aos meus amigos e a todos que estiveram presentes durante esta longa jornada.*

# Agradecimentos

Primeiramente, gostaria de agradecer a Deus, por ser a fonte de todas as minhas forças.

Agradeço, em segundo lugar, a minha família, por todo amor, carinho, educação e por toda a motivação durante todas as etapas da minha vida.

Agradeço também a minha namorada, Mariana, por toda a sua compreensão, incentivo e carinho durante este momento tão desafiador.

Agradeço profundamente aos meus amigos Giovani Candido, Luis Henrique Morelli e Davi Augusto Neves Leite, os quais fizeram os meus dias mais leves e proporcionaram os momentos mais engraçados da graduação.

Por fim, meus sinceros agradecimentos ao Prof. Associado João Paulo Papa, meu orientador, por todas as oportunidades de bolsa de estudos oferecidas e por ter acreditado em meu potencial, e também ao Prof. Dr. Douglas Rodrigues, que se dispôs a compartilhar todos os conhecimentos necessários durante o desenvolvimento dos meus projetos.

*"Vitória, afinal das contas, acho eu."*

Bilbo Bolseiro - O Hobbit

# Resumo

Nos dias atuais, é notório o crescimento da importância que as técnicas de aprendizado de máquina têm tido em virtude da massiva quantidade de dados presentes na Internet. Dentre as situações em que o aprendizado de máquina pode ser empregado, pode-se citar a detecção de tumores em exames médicos, a identificação de perfis de consumo e a detecção de intrusões em redes de computadores. Diante desse contexto, uma das etapas mais importantes para que um sistema de aprendizado tenha desempenhos satisfatórios é a **seleção de características**. Esta etapa envolve aplicar algoritmos ao vetor de características, com a finalidade de encontrar um subconjunto deste vetor tal que aumente a acurácia na classificação e reduza a complexidade do modelo de aprendizado, podendo assim ser compreendida como um problema de otimização NP-Difícil. Deste modo, a utilização de métodos determinísticos não apresenta bom desempenho, tornando as meta-heurísticas, técnicas que se baseiam em comportamentos ótimos encontrados na natureza, excelentes candidatas para esse tipo de problema. Assim sendo, o presente projeto visa o desenvolvimento e implementação de uma aplicação web cujo objetivo é permitir a comparação de tarefas de seleção de características baseadas em técnicas meta-heurísticas.

**Palavras-chave:** Aplicação Web. Seleção de Características. Meta-Heurística.



# Abstract

Nowadays, it is notorious the growth of importance that machine learning techniques have had due to the massive amount of data present on the Internet. Among the situations in which machine learning can be employed, one can cite the detection of tumors in medical exams, the identification of consumption profiles, and the detection of intrusions in computer networks. Given this context, one of the most important steps for a learning system to perform satisfactorily is the **feature selection**. This step involves applying algorithms to the feature vector, in order to find a subset of this vector such that it increases the classification accuracy and reduces the complexity of the learning model, and can thus be understood as an NP-hard optimization problem. Thus, the use of deterministic methods does not perform well, making metaheuristics, techniques that are based on optimal behavior found in nature, excellent candidates for this type of problem. Thus, the present project aims at developing and implementing a web application whose objective is to allow the comparison of feature selection tasks based on metaheuristic techniques.

**Keywords:** Web application. Feature Selection. Metaheuristic.

# Lista de figuras

Figura 1 – Características irrelevantes . . . . .	22
Figura 2 – Características redundantes . . . . .	23
Figura 3 – Esquema da abordagem Wrapper . . . . .	24
Figura 4 – Matriz de Confusão e as Métricas Derivadas . . . . .	32
Figura 5 – Página de Login. . . . .	40
Figura 6 – Resumo das tabelas do sistema. . . . .	41
Figura 7 – Tabela de Usuários. . . . .	42
Figura 8 – Tabela de Datasets. . . . .	43
Figura 9 – Tabela de Funções de Transferência. . . . .	44
Figura 10 – Tabela de Otimizadores. . . . .	45
Figura 11 – Tabela de Tarefas. . . . .	45
Figura 12 – Tabela de Tarefas de Usuário. . . . .	46
Figura 13 – Página de Login. . . . .	47
Figura 14 – Mensagem de Erro: Campos Vazios. . . . .	48
Figura 15 – Mensagem de Erro: Usuário Inválido. . . . .	48
Figura 16 – Página de Cadastro. . . . .	49
Figura 17 – Tooltip. . . . .	50
Figura 18 – Mensagem de Erro: Campos Vazios. . . . .	51
Figura 19 – Mensagem de Erro: Usuário Inválido. . . . .	52
Figura 20 – Mensagem de Erro: Senhas diferentes. . . . .	53
Figura 21 – Mensagem: Usuário criado com sucesso. . . . .	54
Figura 22 – Página de Recuperação de Senha. . . . .	55
Figura 23 – Mensagem de Erro: E-mail não encontrado. . . . .	55
Figura 24 – Mensagem: E-mail para redefinição de senha enviado. . . . .	56
Figura 25 – E-mail recebido. . . . .	56
Figura 26 – Página de Redefinição de Senha. . . . .	57
Figura 27 – Mensagem: Senha Alterada com Sucesso. . . . .	57
Figura 28 – Página de <i>Dashboard</i> . . . . .	58
Figura 29 – <i>Dashboard: Status</i> de Tarefa . . . . .	59
Figura 30 – Página da Tarefa de Seleção de Características. . . . .	60
Figura 31 – Otimizadores Seleccionados. . . . .	61
Figura 32 – Bases de Dados Seleccionadas. . . . .	62
Figura 33 – Funções de Transferências Seleccionadas. . . . .	63
Figura 34 – Expressão LaTeX: Função de Transferência. . . . .	64
Figura 35 – Mensagem de Erro: Campo Vazio. . . . .	65
Figura 36 – Página de Detalhes da Tarefa - Parte 1. . . . .	66

Figura 37 – Página de Detalhes da Tarefa - Parte 2. . . . .	67
Figura 38 – Apresentação do resultado quando a tarefa termina com sucesso. . . . .	67
Figura 39 – Barra de Progresso: Cancelado. . . . .	68
Figura 40 – Barra de Progresso: Sucesso. . . . .	68
Figura 41 – Resultados da Otimização. . . . .	69
Figura 42 – Valores das Métricas de Classificação. . . . .	69
Figura 43 – Gráfico de Distribuição de Frequência por Característica. . . . .	70
Figura 44 – Gráfico de Convergência. . . . .	70
Figura 45 – Resultados da Otimização. . . . .	71
Figura 46 – Valores das Métricas de Classificação. . . . .	71
Figura 47 – Valores das Métricas de Classificação. . . . .	72
Figura 48 – Gráfico de Distribuição Média de Frequências por Característica. . . . .	72
Figura 49 – Gráfico de Convergência Média. . . . .	73
Figura 50 – Página de Erro 404 . . . . .	73

# Lista de abreviaturas e siglas

ABC	<i>Artificial Bee Colony</i>
CS	<i>Cucko Search</i>
GA	<i>Genetic Algorithm</i>
KNN	<i>K-Nearest Neighbor</i>
OPF	<i>Optimum Path-Forest</i>
PSO	<i>Particle Swarm Optimization</i>
SA	<i>Simulated Annealing</i>

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>15</b>
<b>1.1</b>	<b>Problemática</b>	<b>16</b>
<b>1.2</b>	<b>Justificativa</b>	<b>16</b>
<b>1.3</b>	<b>Objetivos</b>	<b>16</b>
1.3.1	Objetivo Geral	16
1.3.2	Objetivos Específicos	17
<b>1.4</b>	<b>Organização do Trabalho</b>	<b>17</b>
<b>2</b>	<b>LEVANTAMENTO BIBLIOGRÁFICO</b>	<b>18</b>
<b>2.1</b>	<b>Otimização</b>	<b>18</b>
<b>2.2</b>	<b>Meta-Heurística</b>	<b>19</b>
2.2.1	Algoritmo Genético (GA)	20
2.2.2	Algoritmo do Vaga-lume (FA)	20
2.2.3	Busca Cuco (CS)	20
2.2.4	Colônia Artificial de Abelhas (ABC)	21
2.2.5	Otimização por Enxame de Partículas (PSO)	21
2.2.6	Recozimento Simulado (SA)	22
<b>2.3</b>	<b>Seleção de Características</b>	<b>22</b>
2.3.1	Técnicas	23
2.3.1.1	<i>Filter</i>	24
2.3.1.2	<i>Wrapper</i>	24
2.3.1.3	<i>Embedded</i>	25
2.3.1.4	<i>Hybrid</i>	25
<b>2.4</b>	<b>Funções de Transferência</b>	<b>25</b>
2.4.1	Família S	25
2.4.2	Família V	26
<b>2.5</b>	<b>Classificador <i>OPF (Optimum Path-Forest)</i></b>	<b>26</b>
2.5.1	Treinamento	26
2.5.2	Teste	28
<b>3</b>	<b>METODOLOGIA</b>	<b>30</b>
<b>3.1</b>	<b>Meta-Heurísticas</b>	<b>30</b>
<b>3.2</b>	<b>Método de Seleção de Características</b>	<b>31</b>
<b>3.3</b>	<b>Funções de Transferência</b>	<b>31</b>
<b>3.4</b>	<b>Métricas de Classificação</b>	<b>31</b>
3.4.1	Acurácia	32

3.4.2	Precisão . . . . .	33
3.4.3	Revocação ou Sensibilidade . . . . .	33
3.4.4	<i>F1-Score</i> . . . . .	33
<b>3.5</b>	<b>Bases de Dados . . . . .</b>	<b>34</b>
3.5.1	Heart Statlog . . . . .	34
3.5.2	Hill Valley . . . . .	34
3.5.3	Ionosphere . . . . .	35
3.5.4	Vehicle . . . . .	35
3.5.5	Sonar . . . . .	35
3.5.6	Spambase . . . . .	35
3.5.7	Waveform . . . . .	35
<b>3.6</b>	<b>Ferramentas . . . . .</b>	<b>36</b>
3.6.1	Opytimizer . . . . .	36
3.6.2	OPFython . . . . .	36
3.6.3	Django . . . . .	36
3.6.4	Celery . . . . .	37
3.6.5	Plotly . . . . .	37
3.6.6	MathJax . . . . .	37
3.6.7	Bootstrap . . . . .	38
<b>4</b>	<b>DESENVOLVIMENTO . . . . .</b>	<b>39</b>
<b>4.1</b>	<b>Painel Administrativo . . . . .</b>	<b>39</b>
4.1.1	Login . . . . .	39
4.1.2	Painel Inicial . . . . .	40
4.1.2.1	Tabela de Usuários . . . . .	41
4.1.2.2	Tabela de Datasets . . . . .	42
4.1.2.3	Tabela de Funções de Transferência . . . . .	43
4.1.2.4	Tabela de Otimizadores . . . . .	44
4.1.2.5	Tabela de Tarefas . . . . .	45
4.1.2.6	Tabela de Tarefas de Usuário . . . . .	46
<b>4.2</b>	<b>MetaOPT . . . . .</b>	<b>46</b>
4.2.1	Login . . . . .	46
4.2.2	Cadastro . . . . .	49
4.2.3	Redefinição de Senha . . . . .	54
4.2.4	<i>Dashboard</i> . . . . .	58
4.2.5	Tarefa de Seleção de Características . . . . .	59
4.2.6	Detalhes da Tarefa . . . . .	65
4.2.7	Resultados . . . . .	68
4.2.7.1	Única Execução . . . . .	68
4.2.7.2	Múltiplas Execuções . . . . .	70

4.2.8	<i>Erro 404</i> . . . . .	73
5	<b>CONCLUSÃO</b> . . . . .	74
5.1	<b>Trabalhos Futuros</b> . . . . .	74
	<b>REFERÊNCIAS</b> . . . . .	75

# 1 Introdução

Os classificadores de padrões são algoritmos projetados para encontrar uma superfície de separação tal que consiga promover uma separabilidade das amostras. Essa separabilidade deverá fazer com que os dados de um lado da superfície pertençam à uma classe e os do outro lado, à outra classe.

Atualmente, as técnicas de classificação de padrões têm sido sendo amplamente aplicadas em diversas áreas do conhecimento, como no auxílio ao diagnóstico médico em sistemas especialistas (REMESEIRO; BOLON-CANEDO, 2019), na identificação de intrusões em redes de computadores (BOUZOUBAA; TAHER; NSIRI, 2021) ou até no reconhecimento de superfícies em imagens obtidas por sensoriamento remoto (SERPICO; BRUZZONE, 2001).

Considerando as aplicações acima mencionadas, é possível inferir que as técnicas de classificação de padrões são utilizadas para resolver problemas constituídos por bases de dados com muitas amostras e com uma grande dimensionalidade (muitas características), o que implica maiores gastos computacionais para processar esses dados. Felizmente, na maior parte das vezes, nem todas as características extraídas dos dados são realmente úteis para a tarefa de classificação, sendo estas consideradas irrelevantes ou redundantes.

A fim de reduzir os custos computacionais e preservar a precisão da classificação das amostras foi desenvolvida uma técnica de redução de dimensionalidade, denominada **seleção de características**. A tarefa de seleção de características é uma das mais importantes em um sistema de aprendizado e pode ser modelada como um problema de otimização, pois o objetivo é escolher o subconjunto de características que maximiza (ou minimiza) a função objetivo. Neste caso, esta técnica é aplicada com a finalidade de encontrar o subconjunto de características que maximizam a separabilidade das amostras, refletindo diretamente na taxa de acerto do classificador.

Dentre as técnicas de otimização, uma considerável atenção por parte dos pesquisadores tem sido dada às metodologias baseadas em meta-heurísticas. As meta-heurísticas compõem um subcampo muito importante da otimização estocástica, cuja é constituída por algoritmos e técnicas que empregam aleatoriedade, conhecimento de resultados anteriores e interação social entre os agentes que irão percorrer o espaço de busca para encontrar as soluções. Embora também possam ser usados para resolver problemas simples e com poucas dimensões, sua principal aplicação é em problemas NP-Difícil.

Vale mencionar que, para esse tipo de problema, os métodos exatos (lógica e programação matemática) não são eficientes, pois frequentemente ficam presos em ótimos locais. De forma semelhante, os algoritmos de força bruta também não são eficientes, haja vista que o espaço de busca é muito grande, sendo extremamente custosa, em termos computacionais, a



enumeração de todas as soluções candidatas. Por fim, embora não haja garantias de convergência para o ponto ótimo, a ideia desse tipo de técnica é encontrar soluções aceitáveis, com custo e tempo computacionais relativamente baixos (YANG, 2014).

## 1.1 Problemática

Atualmente, existem alguns módulos especializados em seleção de características como, por exemplo, o módulo *feature selection* da *API Scikit Learn* (BUTINCK et al., 2013), que possui implementado algumas das técnicas mais famosas da abordagem **wrapper** (BUTCHER; SMITH, 2020, pp. 229–231). Outro exemplo de aplicação foi desenvolvida por Masoudi-Sobhanzadeh, Motieghader e Masoudi-Nejad (2019) e se trata de uma aplicação *desktop* denominada *FeatureSelection*, que utiliza a mesma abordagem do módulo citado anteriormente.

## 1.2 Justificativa

Diante do contexto e problemática apresentados, não foi possível elencar muitas aplicações com *GUI* para realizar a seleção de características baseadas em meta-heurísticas e tampouco aplicações que proporcionem ao usuário ferramentas estatísticas com a finalidade de comparar o desempenho das meta-heurísticas. O desenvolvimento dessa aplicação proporcionará uma facilidade aos usuários, pois agilizará o processo de testes uma vez que estes, sendo um pesquisador ou não, não precisará saber nada sobre programação.

## 1.3 Objetivos

Nesta seção detalha-se o objetivo geral e os objetivos específicos do trabalho.

### 1.3.1 Objetivo Geral

Desenvolver e validar uma aplicação web que permita a realização de análises comparativas entre tarefas de seleção de características baseadas em meta-heurística por meio de dados estatísticos, os quais serão exibidos no formato de tabelas e de gráficos. Os principais resultados envolvem os valores das métricas mais comumente utilizadas na literatura, tais como acurácia, precisão, revocação e *f1-score*. Além disso, gráficos de convergência, resultantes do processo de otimização, e gráficos de frequência média de cada uma das características, também serão exibidos.

### 1.3.2 Objetivos Específicos

Considerando o objetivo geral proposto, os objetivos específicos podem ser elencados como segue:

- Listar as técnicas de seleção de características;
- Identificar as meta-heurísticas mais usadas pela literatura nesse tipo de tarefa;
- Enumerar as principais bases de dados utilizadas na literatura;
- Definir as tecnologias a serem empregadas, tais como: linguagem de programação, *frameworks* e bibliotecas;
- Determinar os parâmetros de execução da tarefa;
- Deliberar como os resultados obtidos serão exibidos;
- Desenvolver a parte lógica da aplicação, bem como integrar as tecnologias;
- Executar a aplicação final e corrigir os eventuais erros.

## 1.4 Organização do Trabalho

O presente trabalho está organizado conforme a estrutura abaixo.

**Capítulo 2:** Apresenta e define os conceitos relacionados à otimização matemática, meta-heurísticas, seleção de características e seus métodos, funções de transferência e classificador OPF (*Optimum Path-Forest*, do inglês);

**Capítulo 3:** Introduce o processo decisório no tocante aos módulos e ferramentas utilizados no projeto e exibe o contexto no qual se deu a integração destes;

**Capítulo 4:** Detalha o desenvolvimento da aplicação e como isso refletiu nas páginas da interface;

**Capítulo 5:** Realiza as considerações finais e sugere potenciais futuros trabalhos.

## 2 Levantamento Bibliográfico

A compreensão deste trabalho implica no conhecimento à priori dos conceitos relacionados à otimização matemática, em específico, à otimização meta-heurística, à seleção de características e à classificação de padrões, os quais são apresentados neste capítulo.

### 2.1 Otimização

Na matemática, a **otimização** é a área que estuda os problemas de maximização e minimização de funções e cujo objetivo é encontrar a solução ótima do problema, ou seja, o conjunto de valores que as variáveis devem assumir de modo a obter o resultado máximo ou mínimo da função. Esse objetivo pode ser atingido pela aplicação de modelos matemáticos e levando-se em conta as restrições do problema.

O formato geral de um problema de otimização pode ser visto abaixo, na Equação 2.1

$$\begin{aligned} &\underset{x \in \mathbb{R}^n}{\text{minimizar}} && f_i(x), \quad (i = 1, 2, \dots, M), \\ &\text{sujeito a} && h_j(x) = 0, \quad (j = 1, 2, \dots, J), \\ &&& g_k(x) \leq 0, \quad (k = 1, 2, \dots, K) \end{aligned} \tag{2.1}$$

onde,  $f_i(x)$  é a função a ser otimizada, também conhecida como **função de custo** ou **função objetivo**, e  $h_j(x)$  e  $g_k(x)$  são as **restrições** do problema. As variáveis  $x_i$  são chamadas de **variáveis de decisão** e podem assumir valores reais contínuos, discretos ou ambos. É importante ressaltar que, embora o formato genérico esteja escrito como um problema de minimização, também é possível escrevê-lo como um problema de maximização. Outrossim, as inequações das restrições também podem ser escritas como  $\geq 0$ .

Um problema de otimização pode ser classificado de acordo com a quantidade de variáveis de decisão. Se houver apenas uma variável de decisão ( $M = 1$ ), é chamado de **objetivo único**; se houverem mais de uma variável de decisão ( $M > 1$ ), de **multiobjetivo**; ou de **viável**, caso só existirem as restrições, mas nenhuma variável de decisão ( $M = 0$ ), neste caso toda solução viável é uma solução ótima. Outro critério de classificação possível está relacionado ao número de restrições  $J + K$ . Se não houver nenhuma restrição ( $J + K = 0$ ), é chamado de problema **sem restrições**; se  $K = 0$  e  $J \geq 1$ , de **restrito à igualdade**; e se  $K \geq 1$  e  $J = 0$ , de **restrito à desigualdades**.

Em geral, os algoritmos para resolução deste tipo de problema são categorizados em **determinísticos** e em **estocásticos**. Os métodos determinísticos recebem essa nomenclatura devido a sua previsibilidade, isto é, para uma determinada entrada, uma mesma saída será obtida.

Uma característica importante dessas técnicas é a necessidade, à priori, de uma informação, que na maioria dos casos trata-se do gradiente (1ª derivada), como é o caso do algoritmo Newton-Raphson, contudo, uma pequena parte das técnicas utiliza uma solução básica viável, como é o caso do método Simplex.

Como o cálculo do gradiente depende do ponto fornecido, há uma grande chance de convergir para um ponto de ótimo local. Isso se torna um problema quando a função a ser otimizada é multimodal, ou seja, que possui dois ou mais pontos de mínimo ou máximo locais, uma vez que a probabilidade de convergência para um ótimo local é maior ainda. Neste cenário, a aplicação de técnicas determinísticas não apresenta um bom desempenho.

## 2.2 Meta-Heurística

Nesse contexto, desenvolveram-se os métodos estocásticos. A otimização estocástica é formada por algoritmos e técnicas que empregam aleatoriedade para tentar encontrar a solução ótima para problemas NP-Difícil. De acordo com Yang (2014), esses métodos são divididos em **heurísticos** e **meta-heurísticos**, entretanto a maior parte dos autores na literatura consideram esses conceitos intercambiáveis, haja vista que sua diferença é significativamente pequena.

As técnicas meta-heurísticas utilizam aleatoriedade, conhecimento de resultados anteriores e interação social entre os agentes que irão percorrer o espaço de busca para encontrar soluções para problemas os quais não se tem muita informação (como o gradiente). De maneira análoga às técnicas determinísticas, não há garantias de convergência para a otimalidade, mas a ideia é que o algoritmo seja prático e funcione na maior parte das vezes, além de ser capaz de produzir boas soluções, isto é, mais próximas do ótimo global.

Para que um algoritmo meta-heurístico seja capaz de atingir o ótimo global é de suma importância que haja o balanço entre dois componentes essenciais, a **intensificação** e **diversificação** (também conhecidos como *exploitation and exploration*, em inglês). A diversificação diz respeito à procura das soluções no espaço de busca como um todo (busca global), objetivo que pode ser atingido por meio da randomização das posições dos agentes. Já a intensificação significa intensificar a busca local na região em que a melhor solução atual foi encontrada.

Por fim, essas técnicas seguem o teorema *no free lunch* (ou sem almoço grátis, em português), que afirma que nenhum algoritmo consegue alcançar a solução ótima de todos os problemas que existem. Apesar disso, podem ser modificadas para ter desempenho melhorado em diversos problemas.

Abaixo, nas seções a seguir, discorre-se brevemente acerca das meta-heurísticas adotadas nesse trabalho, bem como seu mecanismo de funcionamento.

### 2.2.1 Algoritmo Genético (GA)

O algoritmo genético foi concebido por Holland (1992) e se trata de uma modelagem da teoria Darwiniana da seleção natural. O funcionamento dessa meta-heurística considera cada iteração como uma nova geração de indivíduos.

Primeiramente, a função objetivo é codificada para fornecer soluções em termos de vetores binários que representam os cromossomos. Durante sua execução, operadores genéticos são aplicados para garantir a diversificação e intensificação do método. Os operadores são: mutação, *crossover* e seleção.

- **Mutação:** Troca partes de uma solução de forma aleatória;
- **Crossover:** Realiza a troca de partes do cromossomo (solução) com outros cromossomos;
- **Seleção:** Escolha das soluções com melhor *fitness* para levá-los à próxima geração.

Ao final de todas as gerações, o cromossomo com as melhores características, ou seja, a solução que retornar o melhor valor de *fitness* será dada como a solução ótima.

### 2.2.2 Algoritmo do Vaga-lume (FA)

Proposto por Yang (2009), o algoritmo do Vaga-lume ilustra o funcionamento do mecanismo de atração dos parceiros de acasalamento utilizado pelos vaga-lumes, o qual está diretamente ligado ao uso da luz produzida por eles a partir do processo de bioluminescência. Os machos dos vaga-lumes piscam suas luzes em uma determinada frequência e em um determinado intervalo de tempo para atrair as fêmeas que, por sua vez, respondem ao padrão único dos machos. Dependendo da espécie, as piscadas podem ter outras funções além da sexual, como a atração de presas e alertar predadores sobre seu gosto amargo.

Diante o exposto, esse comportamento foi modelado matematicamente levando em consideração dois parâmetros correlacionados: a intensidade da luz e a distância entre os indivíduos. Quanto maior a distância entre os vaga-lumes, menos intensa a luz parecerá e vice-versa, dessa forma. Além disso, indivíduos com luz mais intensa tendem a atrair indivíduos com iluminação menos intensas.

### 2.2.3 Busca Cuco (CS)

Este algoritmo desenvolvido por Yang e Suash Deb (2009) abstrai o comportamento parasitismo de ninhada praticado pela maioria das espécies de aves Cuco. Trata de uma tática evolutiva adotada por algumas espécies de aves, que colocam seus ovos em ninhos de outras

aves, geralmente de espécies diferentes, também conhecidos como ninhos hospedeiros. Os cucos não somente fazem isso, como também, na maioria das vezes, retiram os ovos da outra ave para aumentar a chance de incubação dos seus ovos. Esse comportamento também é replicado pelos filhotes de Cuco que intencionalmente, ao nascerem, também derrubam os outros ovos do ninho.

O modelo considera cada ninho como um agente e cada ovo presente no ninho, uma solução candidata. Neste sentido, os melhores ninhos com ovos de melhor qualidade serão levados as próximas gerações. Assim sendo, não há diferença entre um ovo, um ninho ou um cuco, uma vez que cada ninho contém apenas um ovo, que também representa apenas um cuco.

#### 2.2.4 Colônia Artificial de Abelhas (ABC)

O presente algoritmo foi proposto por Karaboga e Basturk (2007) e pode ser entendido como uma especialização do algoritmo Colônia de Abelhas (ou Bee Colony, em inglês), cuja inspiração se deu no comportamento de forrageamento das abelhas produtoras de mel. Nesta abordagem, cada abelha da colônia é categorizada de acordo com a sua função na procura por fontes de néctar, as quais podem ser empregadas, observadoras e escoteiras, sendo esta última, o que difere os dois algoritmos mencionados.

As abelhas empregadas visitam as fontes de comida para calcular a quantidade de néctar e depois voltam à colmeia para avisar as observadoras que, por sua vez, devem escolher, dentre fontes disponíveis, a que possui a maior quantidade de néctar. Assim que a escolha ocorre, as fontes encontradas pelas outras abelhas são descartadas e as mesmas assumem a função de escoteira, devendo procurar, aleatoriamente, novas fontes de comida. Logo, a função objetivo  $f(x)$  fornecida pode ser codificada como  $F(x)$  para representar a quantidade de néctar na posição  $x$ .

#### 2.2.5 Otimização por Enxame de Partículas (PSO)

Foi desenvolvido por Kennedy, Eberhart e Shi (2001) baseado no comportamento de enxame observado em peixes e pássaros. Nesta modelagem, a procura das soluções dentro do espaço de busca é realizada por agentes individuais, denominados partículas.

Cada partícula é atraída em direção a posição da atual partícula mais bem posicionada, ao mesmo tempo que tende a se mover aleatoriamente. Quando uma partícula encontra uma solução melhor do que as anteriores, ela altera a posição da partícula como a nova melhor. De forma resumida, o objetivo é, portanto, encontrar a melhor solução dentre todas as soluções candidatas até que a mesma não se altere por um determinado número de iterações.

### 2.2.6 Recozimento Simulado (SA)

Esta técnica foi elaborada por Khachaturyan, Semenovsovskaya e Vainshtein (1981), e simula o processo de recozimento, tratamento térmico em que o metal sofre um aquecimento controlado até atingir determinada temperatura, permanecendo nesta até esfriar e congelar num estado cristalino com o mínimo de energia e cristais de tamanhos maiores, a fim de reduzir defeitos em estruturas metálicas.

Considerando a explicação do método, é possível modelar a função objetivo como a função de recozimento, a qual representa a temperatura em que as condições acima mencionadas são atingidas.

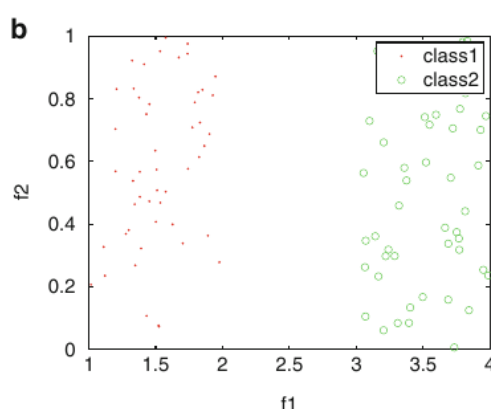
## 2.3 Seleção de Características

O problema principal do Aprendizado de Máquina é aproximar uma função que relacione os dados de entrada  $X = x_1, x_2, \dots, x_M$ , em que  $M$  é o número de características da base, e os dados de saída  $Y$ . Essa relação é estabelecida por meio dos pontos  $X_i, Y_i, i = 1, \dots, N$ , em que  $N$  é o número total de pontos. Usualmente,  $X_i$  são vetores formados por números reais e  $Y_i$  são valores reais.

Todavia, nem sempre uma saída  $Y$  é determinada pelo conjunto completo de características de entrada, isto é, a saída pode ser definida por um subconjunto do conjunto de entrada. As características que não são úteis para determinar o dado  $Y$  podem ser categorizadas como irrelevantes ou redundantes.

Na Figura 1, a característica f2 é irrelevante, pois com ela não é possível dividir as classes corretamente e, além disso, a sua remoção não afeta a capacidade de f1 em distinguir os exemplos da classe1 e da classe2.

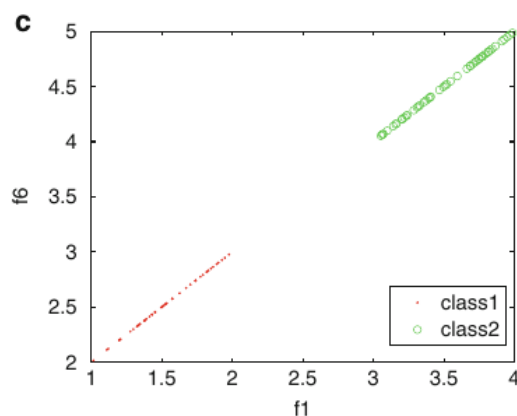
Figura 1 – Características irrelevantes



Fonte: Wang, Tang e Liu (2016)

As características consideradas redundantes implicam na copresença de outra característica, por exemplo, na Figura 2 as características  $f_6$  e  $f_1$  são características relevantes uma vez que, quando analisadas separadamente, conseguem distinguir os exemplos da classe1 e da classe2. No entanto, quando analisadas em conjunto, não há aumento na performance de aprendizado.

Figura 2 – Características redundantes



Fonte: Wang, Tang e Liu (2016)

Quando essas características são inseridas no processo de aprendizado, elas podem causar o aumento de forma polinomial do custo computacional (por cada característica a mais) e o *overfitting*, fenômeno que ocorre quando o modelo aprende demais sobre os dados. Nesse fenômeno, a função de relação entre os dados de entrada e a saída se ajusta sobre todos os pontos  $\{X_i, Y_i\}$  das amostras de treinamento, gerando altas taxas de acerto do classificador nas amostras de treinamento. Contudo, quando o classificador é submetido a novos dados, nunca antes vistos, a performance cai, pois o modelo "decorou" os dados de treino.

A seleção de características, portanto, é considerada uma técnica de **redução de dimensionalidade**, pois o objetivo é escolher, dentre todas as características da base, apenas as relevantes, de forma a manter o modelo de aprendizado pequeno, com um baixo custo computacional e com menores chances de *overfitting*.

### 2.3.1 Técnicas

As técnicas mais comuns segundo Jovic, Brkic e Bogunovic (2015) para executar a tarefa de seleção de características são: *filter*, *wrapper*, *embedded* e *hybrid*.



### 2.3.1.1 Filter

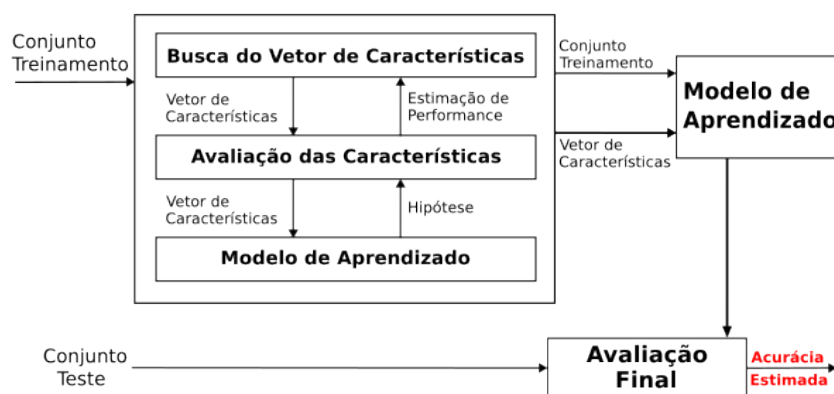
Os métodos *filter* selecionam as características baseadas em métricas de desempenho independentemente do modelo de aprendizado de máquina empregado. Essas métricas podem ser classificadas de acordo com as propriedades que cada uma avalia, podendo ser informação, distância, consistência, similaridade ou estatística.

Para mais, a escolha do método de filtragem depende da tarefa que o conjunto de dados será submetido, sendo assim, eles são estruturados de modo que seja específico para, no máximo, duas tarefas, as quais podem ser classificação, regressão ou agrupamento.

### 2.3.1.2 Wrapper

Os métodos *wrapper* avaliam o conjunto de características a partir do seu desempenho no modelo de aprendizado. Por exemplo, se o objetivo do modelo é a classificação de padrões, então o desempenho será mensurado conforme a performance do classificador. De forma análoga, se a abordagem é o agrupamento (ou *clustering*), a avaliação usará a performance do algoritmo utilizado. Ainda, a avaliação será repetida para cada um dos conjuntos de características e a geração do conjunto ótimo dependerá da estratégia de busca escolhida. Um esquema dessa abordagem pode ser observado na Figura 3.

Figura 3 – Esquema da abordagem Wrapper



Fonte: Adaptado de Kohavi e John (1997)

Essa abordagem é consideravelmente mais lenta do que a *filter*, já que eles avaliam a qualidade do conjunto de características encontrado diretamente no modelo, cuja estrutura, em geral, é grande. Não obstante, Jovic, Brkic e Bogunovic (2015) afirmam que esses métodos são superiores aos de filtragem no tocante a escolha do conjunto de características ótimo.

### 2.3.1.3 *Embedded*

A abordagem *embedded* executa a seleção de características durante a execução do algoritmo do modelo, deste modo ela pode estar incorporada diretamente ao algoritmo do modelo ou ser uma funcionalidade estendida. Os algoritmos mais comumente usados com esse objetivo são CART, C4.5 e florestas aleatórias.

### 2.3.1.4 *Hybrid*

Os métodos *hybrid* (híbrido, em português), como o próprio nome já expõe, preocupam-se em combinar os benefícios das abordagens *filter* e *wrapper*, uma vez que qualquer combinação delas pode ser usada para este propósito. Usualmente, executa-se um algoritmo do tipo *filter* para sobre o espaço de características visando sua redução e, em seguida, um algoritmo *wrapper* é empregado para eleger o melhor subconjunto candidato.

## 2.4 Funções de Transferência

Tratam-se de funções matemática utilizadas com a finalidade de transferir a solução, que pertence ao espaço real e contínuo, para o espaço binário  $[0,1]$ . Via de regra, essas funções pertencem às famílias **S** e a **V**.

### 2.4.1 Família S

Essa família é definida pelo conjunto de funções que possuem o formato de S que, usualmente, são a função Sigmoid e algumas de suas variações. As funções utilizadas nesse trabalho foram a  $S_1$ ,  $S_2$ ,  $S_3$  e  $S_4$ , as quais são representadas, respectivamente, pela Equação 2.2, Equação 2.3, Equação 2.4 e Equação 2.5, conforme listado abaixo.

$$S_1(x_i) = \frac{1}{1 + e^{-x_i}} \quad (2.2)$$

$$S_2(x_i) = \frac{1}{1 + e^{-2x_i}} \quad (2.3)$$

$$S_3(x_i) = \frac{1}{1 + e^{\frac{-x_i}{2}}} \quad (2.4)$$

$$S_4(x_i) = \frac{1}{1 + e^{\frac{-x_i}{3}}} \quad (2.5)$$

### 2.4.2 Família V

Da mesma forma, essa família é definida pelas funções cuja curva é semelhante a letra V. As funções utilizadas nesse trabalho foram a  $V_1$ ,  $V_2$ ,  $V_3$  e  $V_4$ , as quais são representadas, respectivamente, pela Equação 2.6, Equação 2.7, Equação 2.8 e Equação 2.9, conforme apresentado abaixo.

$$V_1(x_i) = \left| \operatorname{erf} \left( \frac{\sqrt{\pi}}{-2x_i} \right) \right| \quad (2.6)$$

$$V_2(x_i) = |\tanh(-x_i)| \quad (2.7)$$

$$V_3(x_i) = \left| \frac{-x_i}{\sqrt{1-x_i^2}} \right| \quad (2.8)$$

$$V_4(x_i) = \left| \frac{2}{\pi \arctan \left( \frac{\pi}{-2x_i} \right)} \right| \quad (2.9)$$

## 2.5 Classificador *OPF (Optimum Path-Forest)*

O classificador OPF, desenvolvido por Papa, Falcão e Suzuki (2009), possui, como a maioria dos algoritmos de classificação de padrões encontrados na literatura, essencialmente, duas fases: **treinamento** e **teste**. Para tanto, é necessário supor que a base de dados que se deseja classificar esteja particionada em dois conjuntos disjuntos,  $Z_1$  e  $Z_2$ , que correspondem, respectivamente, ao conjunto de treinamento e ao conjunto de teste.

### 2.5.1 Treinamento

Na etapa de treinamento, cada amostra pertencente ao conjunto de treinamento  $Z_1$  é representada como nó de um grafo, cujos arcos são definidos por uma relação de adjacência e os seus pesos, por uma função de distância. Usualmente, a função de distância é a Euclidiana, enquanto para a relação de adjacência, duas abordagens podem ser utilizadas, o grafo KNN (*K-Nearest Neighbor*, em inglês), que considera os  $K$  nós mais próximos, e o grafo completamente conectado, em que todos os nós estão conectados entre si por meio das arestas.

Posteriormente, definem-se os **protótipos**, nós escolhidos como representante de cada classe. Para essa tarefa, Ponti e Papa (2011) constroem uma árvore geradora mínima e, a partir dela, os protótipos são definidos como sendo os nós de classes distintas, cuja aresta que

os conecte tenha peso mínimo. Ainda, é relevante mencionar que por classe, ao menos um protótipo deve ser escolhido.

Após isso, são calculados os caminhos ótimos entre cada protótipo e cada uma das amostras de  $Z_1$ , de tal modo que cada protótipo se torne uma raiz de uma árvore de caminhos ótimos (conhecida como *Optimum Path-Tree*, em inglês) composta pelas amostras mais fortemente conectadas. A definição de conectividade, neste contexto, depende da função de custo utilizada na implementação do OPF. Por exemplo, caso a função  $f_{max}$  tenha sido escolhida, uma amostra é considerada fortemente conectada à um protótipo  $X$  quando seu custo é menor do que a aresta de maior custo presente na árvore liderada por  $X$ . Ademais, quando uma amostra decide de qual árvore deseja fazer parte, diz-se que a amostra foi conquistada pelo protótipo representante desta árvore.

Diante o exposto no parágrafo anterior, explica-se a denominação do modelo pelo fato de duas ou mais árvores de caminhos ótimos serem geradas durante o processo de treinamento, constituindo uma floresta de caminhos ótimos.

Destarte, de forma resumida, na fase de treinamento serão encontrados os protótipos e determinadas as árvores de caminho mínimo encabeçadas por eles. O pseudocódigo dessa etapa pode ser observado no algoritmo 1.

**Algoritmo 1:** Pseudocódigo - Etapa de Treinamento do Classificador OPF**Entrada:** Um conjunto de treinamento  $T$  e seus rótulos  $\lambda$ .**Saída:** Floresta de Caminhos Ótimos  $P_1$ , Vetor de Custos  $C_1$ , Vetor de Rótulos  $L_1$ , Conjunto ordenado  $T'$ .**Auxiliar:** Fila de Prioridades  $Q$ , Conjunto de Protótipos  $S$ , Variável de Custo  $cst$ .

```

1  Início
2  | Configure  $T' \leftarrow 0$  e compute por MST o conjunto de protótipos  $S \subset T$ .
3  | Para cada  $s \in T \setminus S$  , faça
4  |   | Configure  $C_1(s) \leftarrow +\infty$ .
5  | fim
6  | Para cada  $s \in S$  , faça
7  |   |  $C_1(s) \leftarrow 0$ ,  $P_1(s) \leftarrow nil$ ,  $L_1(s) \leftarrow \lambda(s)$  e insira  $s$  em  $Q$ .
8  | fim
9  | Enquanto  $Q$  não é vazia , faça
10 |   | Remova de  $Q$  uma amostra  $s$  tal que  $C_1(s)$  é mínimo.
11 |   | Insira  $s$  em  $T$ .
12 |   | Para cada  $t \in T$  tal que  $t \neq s$  e  $C_1(t) > C_1(s)$  , faça
13 |   |   | Compute  $cst \leftarrow \max\{C_1(s), d(s, t)\}$ .
14 |   |   | Se  $cst < C_1(t)$  , então
15 |   |   |   | Se  $C_1(t) \neq +\infty$  , então
16 |   |   |   |   | Remova  $t$  de  $Q$ .
17 |   |   |   | fim
18 |   |   |   |  $P_1 \leftarrow s$ ,  $L_1(t) \leftarrow L_1(s)$ ,  $C_1(t) \leftarrow cst$ 
19 |   |   |   | Insira  $t$  em  $Q$ .
20 |   |   | fim
21 |   | fim
22 | fim
23 fim
24 Retorna Classificador  $[P_1, C_1, L_1, T']$ 

```

## 2.5.2 Teste

Na etapa de testes ou classificação, o conjunto de testes  $Z_2$  será submetido ao classificador que, então, calculará a distância entre cada amostra (nó) de  $Z_2$  e todos os nós das árvores geradas pelo algoritmo descrito na subseção 2.5.1. O nó cuja distância for a menor, definirá o rótulo a amostra de teste. Abaixo, no algoritmo 2, o pseudocódigo correspondente à etapa de teste é mostrado.

**Algoritmo 2:** Pseudocódigo - Etapa de Classificação do Classificador OPF

**Entrada:** Classificador  $[P_1, C_1, L_1, T']$ , conjunto de avaliação  $E$  (ou conjunto de teste  $O$ ), o par  $(v, d)$  para o vetor de características e cálculo das distâncias.

**Saída:** Rótulo  $L_2$  e o mapa de predecessores  $P_2$  definidos por  $E$ .

**Auxiliar:** Variáveis de custo  $tmp$  e  $mincost$ .

```

1  Início
2  |   Para cada  $t \in E$  , faça
3  |   |    $i \leftarrow 1$ ,  $mincost \leftarrow \max\{C_1(k_i), d(k_i, t)\}$ .
4  |   |    $L_2 \leftarrow L_1(k_i)$  e  $P_2(t) \leftarrow k_i$ .
5  |   |   Enquanto  $i < |T'|$  e  $mincost > C_1(k_{i+1})$  , faça
6  |   |   |   Compute  $tmp \leftarrow \max\{C_1(k_{i+1}, d(k_{i+1}, t))\}$ .
7  |   |   fim
8  |   |   Se  $tmp < mincost$  , então
9  |   |   |    $mincost \leftarrow tmp$ .
10 |   |   |    $L_2 \leftarrow L_{k_{i+1}}$  e  $P_2(t) \leftarrow k_{i+1}$ .
11 |   |   fim
12 |   |    $i \leftarrow i + 1$ .
13 |   fim
14 fim
15 Retorna  $[L_2, P_2]$ 

```

## 3 Metodologia

Neste capítulo, será detalhado todo o processo de decisão envolvido na escolha das meta-heurísticas, funções de transferência e bases de dados, bem como das tecnologias empregadas no desenvolvimento da aplicação, além de expor as condições experimentais de hardware e a organização das etapas da aplicação.

### 3.1 Meta-Heurísticas

A escolha da quantidade de meta-heurísticas norteou-se na limitação da biblioteca Opytimizer, no tocante a quantas e quais meta-heurísticas estão implementadas. Em adição, era sabido que a incorporação de uma grande quantidade de meta-heurísticas consumiria um tempo significativo no desenvolvimento da aplicação. Neste contexto, deliberou-se por apenas 6 algoritmos.

A apuração foi embasada no levantamento realizado por Dokeroglu, Deniz e Kiziloz (2022), o qual listou as técnicas meta-heurísticas mais populares para a tarefa de seleção de características, de acordo com a quantidade de resultados obtidos nos repositórios Google Scholar e Scopus. Em razão do Google Scholar possuir um montante de artigos, periódicos e livros substancialmente superior ao Scopus, foram aceitos apenas os resultados fornecidos pelo repositório do Google.

O estudo contemplou 22 técnicas desenvolvidas no intervalo de duas décadas (2000 a 2020), consideradas pelos autores como recentes. Ainda, os autores compararam a popularidade das técnicas recentes com 8 tidas como clássicas, isto é, concebidas antes do ano 2000.

Os resultados obtidos mostraram que no Google Scholar, dentre os algoritmos clássicos, os três mais expressivos em termos de número de estudos publicados, foram: GA (81.900), PSO (43.200) e, por fim, o SA (17.600). Enquanto os resultados obtidos com relação aos algoritmos recentes foram bem menos expressivos, o que pode ser explicado pelo fato de terem sido elaborados a menos tempo do que as clássicas. Neste caso, a técnica ABC se destacou, com 10.300 estudos, seguida por FA (6.440) e, ao final, CS (6.300).

As meta-heurísticas selecionadas para fazerem parte da aplicação foram, por conseguinte, GA, PSO, SA, ABC, FA e CS, as quais tem o funcionamento descrito brevemente na seção 2.2.

## 3.2 Método de Seleção de Características

Uma vez que a aplicação proposta é baseada no uso de meta-heurísticas para selecionar as características, é possível concluir que o método empregado para executar essa tarefa foi a *wrapper*. Esta abordagem, como detalhado na subseção 2.3.1.2, aplica um algoritmo de busca para definir cada subconjunto de características candidato, que neste caso entende-se como sendo as meta-heurísticas.

## 3.3 Funções de Transferência

Dado que a aplicação utiliza a abordagem *wrapper* (veja subseção 2.3.1.2) e as meta-heurísticas selecionadas retornam resultados no espaço real, fez-se necessário o uso de funções de transferência. Essas funções foram escolhidas fundamentando-se no *survey* escrito por Dokeroglu, Deniz e Kiziloğlu (2022), que analisaram 82 estudos acerca do uso das meta-heurísticas na tarefa de seleção de características.

Deste total, 46 não utilizam funções de transferência, 25 aplicam as funções da família S (leia a subseção 2.4.1), 18 implementam a família V (vide subseção 2.4.2) e 2 estudos aplicam um limiar de representação para o espaço binário. Sendo assim, foram adotadas para aplicação todas as funções da família S e todas da família V.

## 3.4 Métricas de Classificação

Em consonância com Sammut e Webb (2017), a matriz de confusão pode ser definida como uma estrutura que resume toda a performance de um classificador de padrões. Dessa estrutura, derivam as métricas mais populares de desempenho de classificadores, tais como a **acurácia**, **precisão**, **revocação** e ***f1-score***. Uma representação da matriz de confusão  $2 \times 2$ , utilizada para problemas dicotômicos, é exibida abaixo, na Figura 4.



Figura 4 – Matriz de Confusão e as Métricas Derivadas

		Classe Predita		
		Positivo	Negativo	
Classe Real	Positivo	Verdadeiro Positivo (VP)	Falso Negativo (FN)	<b>Revocação</b> $\frac{VP}{(VP + FN)}$
	Negativo	Falso Positivo (FP)	Verdadeiro Falso (VF)	<b>Especificidade</b> $\frac{VF}{(VF + FP)}$
		<b>Precisão</b> $\frac{VP}{(VP + FP)}$	<b>Predição de Valores Negativos</b> $\frac{VF}{(VF + FN)}$	<b>Acurácia</b> $\frac{VP + VF}{(VF + VF + FP + FN)}$

Fonte: Adaptado de An (2020).

Normalmente, a altura da matriz corresponde a cada um dos rótulos reais das amostras, enquanto a largura representa os rótulos preditos. Como, em problemas dicotômicos, existem apenas dois rótulos (ou classes), então, por exclusão, se uma amostra não pertence a classe 1, pertencerá a classe 2. Logo, uma forma possível de se abordar a matriz de confusão é considerar os rótulos como sendo positivos ou negativos, ou usando os próprios nomes das classes.

Além disso, cada quadrante possui um valor associado, que será aplicado no cálculo das métricas detalhadas subseções a seguir. Os valores são:

- **Verdadeiro Positivo (VP)**: Quantidade de amostras foram preditas como positivo e o rótulo real é, de fato, positivo;
- **Falso Negativo (FN)**: Quantidade de amostras com rótulos verdadeiros e que foram preditos como falso;
- **Falso Positivo (FP)**: Quantidade de amostras preditas como verdadeiro, mas que, na realidade, eram falsos;
- **Verdadeiro Negativo (VN)**: Quantidade de amostras previstas como negativo e o rótulo real é, de fato, negativo.

### 3.4.1 Acurácia

Expressa a quantidade de predições corretas em relação a todas as predições realizadas. Ainda que essa métrica seja particularmente útil quando a base de dados é balanceada, ou seja, quando há, aproximadamente, a mesma quantidade de amostras por classe, ela raramente é ponderada como métrica única para se avaliar um classificador.

A Equação 3.1 ilustra a fórmula da acurácia.

$$Acurácia = \frac{n^o \text{ predições corretas}}{n^o \text{ total de predições}} = \frac{VP + VN}{VP + VN + FP + FN} \quad (3.1)$$

### 3.4.2 Precisão

Calcula a proporção de verdadeiros positivos em relação a todas as predições positivas, incluindo as falsas positivas e os reais acertos, avaliando, assim, a assertividade do modelo quando ele decide fazer uma classificação positiva. Numericamente, a precisão pode assumir valores no intervalo entre 0 e 1, em que 0 indica que o modelo apenas irá predizer falsos positivos e 1, indica que nenhuma amostra foi erroneamente classificada como positiva.

Um modelo preciso não indica, necessariamente, que todos os rótulos positivos serão preditos como positivos, mas que possui uma quantidade de verdadeiros positivos significativamente superior ao de falsos positivos.

A fórmula da precisão é dada pela Equação 3.2, apresentada abaixo.

$$Precisão = \frac{VP}{VP + FP} \quad (3.2)$$

### 3.4.3 Revocação ou Sensibilidade

Indica a proporção de amostras positivas em relação a quantidade de amostras classificadas corretamente pelo modelo. De forma análoga à precisão, a revocação também pode assumir valores dentro do intervalo de 0 e 1.

A Equação 3.3 exibe a fórmula da revocação:

$$Revocação = \frac{VP}{VP + FN} \quad (3.3)$$

Para mais, essa métrica possui grande importância em modelos desenvolvidos para lidar com problemas médicos, como, por exemplo, modelos para classificar existência ou não de câncer. Pois, um valor de revocação baixo indica que o modelo está admitindo muitos falsos negativos, fato que pode implicar diretamente num possível tratamento precoce dos pacientes, ocasionando, inclusive, em fatalidades.

### 3.4.4 F1-Score

É definida pela **média harmônica** entre a precisão e a revocação, deste modo, seu valor dependerá dos resultados dessas duas métricas, como é possível observar na Equação 3.4.

$$F1-Score = 2 * \frac{Precisão * Revocação}{Precisão + Revocação} \quad (3.4)$$

Logo, infere-se que altas taxas de precisão e revocação implicam em um alto valor F1. Do mesmo modo, baixas taxas de precisão e revocação implicam em baixo valor de F1. E, caso uma das métricas seja menor que a outra, a F1 lidará com um valor médio.

### 3.5 Bases de Dados

Optaram-se pelas bases de dados pertencentes ao repositório UCI Machine Learning (DUA; GRAFF, 2017), o qual é muito utilizado na literatura no que tange à seleção de características, como pode ser observado nos artigos de Mohamed et al. (2020) e Too e Abdullah (2020).

Ademais, como não há um número definido de bases para essa tarefa, foram selecionadas apenas 7, sendo elas: Heart Statlog, Hill Valley, Ionosphere, Vehicle, Sonar, Spambase e Waveform. Essa escolha feita de tal modo que houvesse um equilíbrio entre o que se considerou bases pequenas, médias e grandes, em termos de quantidade de características. Foram consideradas bases pequenas aquelas constituídas de 10 a 30 características; bases médias, as constituídas de 31 a 50 características e bases grandes, as com mais de 50 características.

#### 3.5.1 Heart Statlog

Essa base de dados é constituída de **270 amostras**, em que cada amostra é composta por **13 características** extraídas de imagens de exames cardiológicos. O objetivo é, por meio da análise das amostras, prever se o indivíduo possui ou não alguma doença do coração.

#### 3.5.2 Hill Valley

Se trata de uma base de dados representativa de um problema dicotômico (2 classes), contendo **606 amostras**, em que cada amostra é composta por **100 características**. Na prática, cada amostra representa 100 pontos de um gráfico bidimensional que, quando plotado em ordem, pode formar ou desenho de uma colina (*hill*, em inglês), ou de um vale (*valley*, em inglês). Portanto, o objetivo é classificar corretamente as amostras do conjunto de teste, em colina ou vale.

### 3.5.3 Ionosphere

Essa base de dados é constituída de **351 amostras**, em que cada amostra é composta por **34 características** provenientes do monitoramento da ionosfera, camada da Terra que se localiza entre 80km e 1000km de altitude, responsável por refletir ondas de rádio e desintegrar corpos celestes que invadem a atmosfera. O objetivo é, por meio da análise das amostras, prever se a região de estudo apresenta alguma estrutura ionosférica ou não.

### 3.5.4 Vehicle

Essa base de dados é constituída de **846 amostras**, em que cada amostra é composta por **18 características** extraídas de silhuetas de 4 diferentes tipos de veículos: **opel, saab, bus, van**. O objetivo é, por meio da análise das amostras, determinar a qual dos tipos de veículos a silhueta em questão pertence.

### 3.5.5 Sonar

Essa base de dados é constituída de **208 amostras** de padrões obtidos pela reflexão de sinais de sonar sobre cilindros de metal e cilindros de rocha, em vários ângulos e sob várias condições; cada amostra é composta por **60 características**, que representam informações relacionadas à reflexão do sinal, como por exemplo, a energia pertencente a uma frequência de banda particular. O objetivo é, por meio da análise das amostras, prever se o cilindro estudado é de metal ou feito de rocha.

### 3.5.6 Spambase

Essa base de dados é constituída de **4601 amostras**, em que cada amostra é referente a um e-mail e cada uma das **48 características** denota letras, palavras e a frequência de ocorrência delas no corpo do e-mail. Além disso, a base representa um problema dicotômico, dessa forma, deseja-se saber se cada amostra é um e-mail de spam ou não.

### 3.5.7 Waveform

Essa base de dados é constituída de **5000 amostras** referentes a ondas as quais foram introduzidos ruídos. Cada amostra é composta por **40 características** que representam valores de entre 0 e 6. O objetivo é, por meio da análise das características, identificar qual das 3 ondas base formaram a onda em questão. Diferentemente das bases anteriores, esta trata-se de um problema com 3 classes.

## 3.6 Ferramentas

Para o desenvolvimento da aplicação, foi utilizado um computador pessoal com as seguintes especificações: processador modelo Intel® Core™ i7-4790 CPU @ 3.60GHz x 8; placa de vídeo NVIDIA GeForce GTX 1050 TI de 4GB; 16GB de memória RAM e SSD de 480GB. Com relação ao software, o sistema operacional contido neste dispositivo foi o Zorin OS 16 de 64 bits, além do editor de texto Visual Studio Code (1.74.2).

Ademais, a linguagem de programação escolhida foi Python devido a sua documentação abrangente e a vasta quantidade de bibliotecas e *frameworks* existentes, as quais destacam-se as direcionadas a aplicações que envolvem manipulação de uma grande quantidade de dados.

### 3.6.1 Opytimizer

A Opytimizer é um *framework* de código aberto, em Python, desenvolvido por Rosa, Rodrigues e Papa (2019), que contém uma grande quantidade de meta-heurísticas implementadas, abrangendo desde as evolucionárias até as baseadas em enxame de partículas. De acordo com os autores, a ideia principal por trás desta biblioteca era permitir tanto o desenvolvimento de novos algoritmos meta-heurísticos por parte dos usuários quanto facilitar a integração com outras bibliotecas de grande prestígio na computação, como Scikit-Learn, PyTorch, entre outros. A versão utilizada neste projeto foi a 3.1.1, a mais recente até a finalização deste trabalho.

### 3.6.2 OPFython

Rosa e Papa (2021) elaboraram e implementaram um *framework* em Python que contém algoritmos de Floresta de Caminhos Ótimos (vide seção 2.5). Para este projeto, foi aplicada a instância do classificador OPF para aprendizado supervisionado, ou seja, a abordagem para problemas cujas classes (ou rótulos) das amostras são já conhecidos. A versão utilizada neste projeto foi a 1.0.11.

### 3.6.3 Django

O Django é um *framework* web, *open-source* que utiliza a linguagem Python para o desenvolvimento de aplicações web. Para mais, é de suma importância informar que o padrão de arquitetura de software utilizado no seu desenvolvimento foi o MVT, uma variação do padrão MVC comumente utilizado no desenvolvimento web. A adoção do MVT implica na existência das camadas *view*, *model* e *template*. A camada *view* é responsável pela lógica da aplicação, bem como pela renderização das *templates*, páginas HTML que irão formatar e exibir os dados

ao usuário. Ademais, a *model* é responsável pela instância do banco de dados e por todos os métodos relacionados a ele, como, por exemplo, às consultas.

Em síntese, a anteposição deste *framework* ocorreu em virtude da sua compatibilidade e suporte às bibliotecas selecionadas para a funcionalidade proposta, as quais foram programadas em Python.

### 3.6.4 Celery

O Celery (v5.2) é uma ferramenta desenvolvida para escalonamento de tarefas. Na prática, as tarefas a serem executadas são enviadas pela aplicação cliente (*host*) ao Celery, iniciando um processo de troca de mensagens entre eles. Esta comunicação é realizada através de uma instância Redis<sup>1</sup> e tem a finalidade de permitir que o agente Celery informe a aplicação sobre os progressos de execução.

É importante ressaltar que em virtude da execução assíncrona das tarefas, permite-se ao usuário iniciar uma ou mais tarefas e continuar navegando pela aplicação, ou até mesmo fechar a aba sem que as execuções sejam comprometidas. Essa característica da aplicação foi viabilizada pelo uso dos módulos Celery Progress e Django Celery Results, em que o primeiro é responsável pela barra de progresso e o segundo, pelo armazenamento das informações e resultados da tarefa no banco de dados, garantindo a persistência dos dados. Portanto, o Celery como um todo, representa uma funcionalidade crucial da aplicação.

### 3.6.5 Plotly

Para a geração de gráficos em HTML que poderiam ser integrados em *templates* Django, empregou-se a biblioteca gráfica e de código aberto, Plotly<sup>2</sup>. Este módulo suporta aproximadamente 40 tipos de gráficos, dentre os quais foram usados dois, o de barras e o de dispersão. A versão utilizada neste projeto foi a 5.11.0.

### 3.6.6 MathJax

No decorrer do projeto, fez-se necessária a exibição de fórmulas referentes às funções de transferências. A solução encontrada foi a utilização do MathJax<sup>3</sup> (v3.2.2), ferramenta *open source* desenvolvida em JavaScript com o propósito de renderizar expressões LaTeX, MathML e notações AsciiMath. Outrossim, este módulo é compatível com todos os navegadores modernos.

---

<sup>1</sup> O Redis é um banco de dados não relacional muito utilizado para troca de mensagens em tempo real.

<sup>2</sup> Disponível em: <<https://plotly.com/graphing-libraries/>>. Acesso em: 02 de jan. 2023.

<sup>3</sup> Disponível em: <<https://www.mathjax.org/>>. Acesso em 03 jan. 2023

### 3.6.7 Bootstrap

O Bootstrap<sup>4</sup> trata-se de um *framework* para desenvolvimento de componentes de interface para aplicações web, que integra HTML, CSS e JavaScript. Essa ferramenta, em sua versão 5.3.0, foi incorporada à aplicação e utilizada durante todas as etapas de *front-end*, com a finalidade de acelerar o desenvolvimento e deixar a interface gráfica com uma aparência mais amigável e moderna ao usuário.

---

<sup>4</sup> Disponível em: <<https://getbootstrap.com/>>. Acesso em: 20 de dez. 2022

## 4 Desenvolvimento

Neste capítulo, esmiúça-se as etapas de desenvolvimento do projeto, dentre as quais incluem-se o painel administrativo e o sistema MetaOPT, dando enfoque na aplicação de seleção de características.

### 4.1 Painel Administrativo

Além de todas as vantagens provenientes do modelo MVT, o Django também gera um site administrativo, totalmente customizável, com a finalidade de permitir que os administradores da aplicação consigam lidar com todas as questões relacionadas ao banco de dados tais como, criar tabelas, inserir registros, adicionar atributos e entre outros, de forma gráfica. Para acessá-lo foi necessário, primeiramente, executar um comando que inicializou o banco de dados e criou uma tabela referente aos usuários, e criar uma conta de acesso especial, denominada superusuário.

Ainda, é válido mencionar que o banco de dados nativo do Django é o SQLite, entretanto, bancos de dados como o PostgreSQL e MySQL, também podem ser integrados a ele. Em vista da facilidade do banco nativo, optou-se, deste modo, pelo uso banco de dados padrão do *framework*.

#### 4.1.1 Login

Como descrito no início desta seção, o acesso ao painel administrativo é restrito aos gerentes do sistema, os quais possuem uma conta de superusuário. Normalmente, o endereço adotado pelo Django para entrar na página de login implica na adição do comando `'/admin'` imediatamente após o endereço raiz da aplicação, como, por exemplo, **`http://localhost:8000/admin`**, no caso de aplicações executadas de maneira local. Porém, como a maioria da estrutura do Django, é possível customizar o caminho desta funcionalidade também.

Ao acessar o endereço referente ao painel administrativo, a página mostrada na Figura 5 será renderizada .



Figura 5 – Página de Login.

The image shows a web form for logging into the Django administration interface. At the top, there is a dark teal header bar with the text "Administração do Django" in white. Below this, the form is white and contains two input fields. The first field is labeled "Usuário:" and has a single vertical line inside, indicating it is active. The second field is labeled "Senha:" and is empty. Below these fields is a teal button with the text "Acessar" in white. The entire form is centered on a light gray background.

Fonte: Elaborada pelo autor.

#### 4.1.2 Painel Inicial

Efetuada o login, será exibido um painel contendo uma visão geral das tabelas do sistema, conforme a Figura 6. Ainda, as tabelas exibidas são divididas de acordo com as funcionalidades do sistema que elas fazem parte. Neste sentido, as tabelas utilizadas pelo módulo de seleção de características do MetaOPT são: Usuários, Datasets, TransferFunctions e Opytimizer, Task Result e User Task.

Figura 6 – Resumo das tabelas do sistema.

Administração do Django	
Administração do Site	
AUTENTICAÇÃO E AUTORIZAÇÃO	
Usuários	+ Adicionar    ✎ Modificar
CELERY RESULTS	
Task results	👁 Visualizar
DASHBOARD	
Datasets	+ Adicionar    ✎ Modificar
Functions	+ Adicionar    ✎ Modificar
Optimizers	+ Adicionar    ✎ Modificar
Transfer functions	+ Adicionar    ✎ Modificar
User tasks	👁 Visualizar

Fonte: Elaborada pelo autor.

#### 4.1.2.1 Tabela de Usuários

Na Figura 7, detalhe-se a tabela de usuários, que contém todas as informações fornecidas pelo usuário no momento no cadastro. Nela, encontram-se os seguintes campos: identificador (id), nome de usuário, endereço de e-mail, primeiro e último nomes e, por fim, um campo booleano que indica se o usuário é um administrador do sistema. Apesar de não ser exibido na imagem, um botão no canto superior direito da página pode ser encontrado, o qual permite a criação de novos usuários manualmente.

Figura 7 – Tabela de Usuários.

Selecione usuário para modificar

ADICIONAR USUÁRIO +

Ação:   0 de 2 selecionados

<input type="checkbox"/>	ID	USUÁRIO	ENDEREÇO DE EMAIL	PRIMEIRO NOME	ÚLTIMO NOME	MEMBRO DA EQUIPE
<input type="checkbox"/>	1	admin	admin@gmail.com			✓
<input type="checkbox"/>	3	usuario.tcc	usuario@gmail.com	usuario	fulano	✗

2 usuários

Fonte: Elaborada pelo autor.

#### 4.1.2.2 Tabela de Datasets

A tabela de Datasets apresentada na Figura 8, armazena os dados relacionados às bases de dados escolhidas (vide seção 3.5). Dentre as informações armazenadas pode-se citar o seu nome, a quantidade de características e a o número de amostras. Além disso, embora não esteja visualmente explícito, outro campo relacionado às bases é o caminho para seu arquivo, utilizado para encontrá-las dentro das pastas da aplicação.

Figura 8 – Tabela de Datasets.

Selecione dataset para modificar

ADICIONAR DATASET +

Ação:  Ir 0 de 7 selecionados

<input type="checkbox"/>	NAME	FEATURES	SAMPLES
<input type="checkbox"/>	Heart Statlog	13	270
<input type="checkbox"/>	Hill Valley	100	606
<input type="checkbox"/>	Ionosphere	34	351
<input type="checkbox"/>	Sonar	60	208
<input type="checkbox"/>	Spambase	48	4601
<input type="checkbox"/>	Vehicle	18	846
<input type="checkbox"/>	Waveform	40	5000

7 datasets

Fonte: Elaborada pelo autor.

#### 4.1.2.3 Tabela de Funções de Transferência

De forma análoga à tabela de Datasets, a Figura 9 apresenta a tabela de funções de transferência onde as são armazenadas informações sobre as mesmas. Neste caso, o seu nome e sua expressão LaTeX, o qual é renderizada na página do módulo de seleção de características, detalhado na subseção 4.2.5.

Figura 9 – Tabela de Funções de Transferência.

Selecione transfer function para modificar

ADICIONAR TRANSFER FUNCTION +

Ação:  Ir 0 de 8 selecionados

<input type="checkbox"/>	NAME	LATEX EXPRESSION
<input type="checkbox"/>	S1	$f(x_i) = \frac{1}{1 + e^{-x_i}}$
<input type="checkbox"/>	S2	$f(x_i) = \frac{1}{1 + e^{-2x_i}}$
<input type="checkbox"/>	S3	$f(x_i) = \frac{1}{1 + e^{\frac{-x_i}{2}}}$
<input type="checkbox"/>	S4	$f(x_i) = \frac{1}{1 + e^{\frac{-x_i}{3}}}$
<input type="checkbox"/>	V1	$f(x_i) = \left  \operatorname{erf} \left( \frac{\sqrt{\pi}}{2} x_i \right) \right $
<input type="checkbox"/>	V2	$f(x_i) = \left  \tanh(x_i) \right $
<input type="checkbox"/>	V3	$f(x_i) = \left  \frac{x_i}{\sqrt{1 + x_i^2}} \right $
<input type="checkbox"/>	V4	$f(x_i) = \left  \frac{2}{\pi} \arctan(x_i) \right $

8 transfer functions

Fonte: Elaborada pelo autor.

4.1.2.4 Tabela de Otimizadores

A Figura 10 mostra a tabela das meta-heurísticas escolhidas, os únicos campos que ela contém são o nome da técnica e o seu respectivo acrônimo. O acrônimo, dentro do código da aplicação, se faz muito importante, pois através dele é possível instanciar a classe do otimizador diretamente da biblioteca Opytimizer.

Figura 10 – Tabela de Otimizadores.

Selecione optimizer para modificar

ADICIONAR OPTIMIZER +

Ação:  Ir 0 de 6 selecionados

- ☐ OPTIMIZER
- ☐ Algoritmo Genético (GA)
- ☐ Algoritmo do Vaga-lume (FA)
- ☐ Busca Cuco (CS)
- ☐ Colônia Artificial de Abelhas (ABC)
- ☐ Otimização por Enxame de Partículas (PSO)
- ☐ Recozimento Simulado (SA)

6 optimizers

Fonte: Elaborada pelo autor.

#### 4.1.2.5 Tabela de Tarefas

Esta tabela contém o registro de todas as tarefas que foram executadas no sistema. O registro conta com as informações sobre o status de execução da tarefa, o seu tipo, seus parâmetros de execução, os resultados obtidos, além das datas de criação e término. A imagem da tabela pode ser observada abaixo, na Figura 11.

Figura 11 – Tabela de Tarefas.

Selecione task result para visualizar

Q

Pesquisar

2023

16 de Janeiro

Ação:

Ir

0 de 2 selecionados

<input type="checkbox"/>	TASK ID	TASK STATE	TASK NAME	TASK ARGUMENTS	RESULT DATA	CREATED DATETIME	COMPLETED DATETIME
<input type="checkbox"/>	865706c8-914f-4a24-8453-c5faa225c055	REVOKED	feature_selection	{ "user_id": 1, "optimizer": "FA", "function": "Classificador OPF", "dataset": ...	{ "exc_type": "TaskRevokedError", "exc_message": ["terminated"], "exc_module": ...	16 de Janeiro de 2023 às 23:24	16 de Janeiro de 2023 às 23:24
<input type="checkbox"/>	3cec8ca7-4a2b-4bb0-b593-d0c902c8ff81	SUCCESS	feature_selection	{ "user_id": 1, "optimizer": "PSO", "function": "Classificador OPF", "dataset": ...	{ "best_solution": [0.7022642408197249, 0.501216451687087, 0.5060868335126388, ...	16 de Janeiro de 2023 às 23:22	16 de Janeiro de 2023 às 23:23

2 task results

Fonte: Elaborada pelo autor.

#### 4.1.2.6 Tabela de Tarefas de Usuário

Por fim, esta tabela armazena todas as tarefas executadas e qual o usuário responsável por ela. Dessa forma, ela é usada para consultar as informações sobre cada tarefa de um determinado usuário e preencher a *dashboard* com elas. A Figura 12 ilustra a tabela de tarefas do usuário.

Figura 12 – Tabela de Tarefas de Usuário.

Selecione user task para visualizar

<

Fonte: Elaborada pelo autor.

## 4.2 MetaOPT

Esta seção introduz e detalha o funcionamento das páginas principais do projeto MetaOPT. O sistema é composto de dois módulos, o de otimização e o de seleção de características, contudo, o escopo deste trabalho contempla apenas o segundo módulo. Ademais, a explicação da aplicação levará em consideração o fluxo de navegação esperado de um novo usuário.

### 4.2.1 Login

Ao acessar o endereço da aplicação, o usuário será direcionado para a página de login para que insira suas credenciais e, deste modo, tenha acesso as suas tarefas.

Como a Figura 13 mostra, esta página é constituída por uma estrutura central, denominada *card*, cujo objetivo é servir como um contêiner para o formulário de login. Este, por sua vez, é composto de dois campos de texto, designados para o nome do usuário e a senha; por um botão de acesso à aplicação, que validará as credenciais; por dois links, um para redefinição

de senha e o outro, para usuários que ainda não possuem cadastro; e um botão para exibição dos conteúdos dos campos das senhas.

Para mais, é notória a existência de um menu de navegação (ou *navbar*, em inglês) na parte superior da página, com links para acesso rápido às funcionalidades do sistema. Quando o usuário não está conectado à aplicação, os links, da esquerda para a direita, direcionam para a página de login, para a página de cadastro e para página de redefinição de senha, respectivamente. Na parte inferior da página, encontra-se o rodapé, que possui o slogan do MetaOPT. Ambos, menu de navegação e rodapé, compõem a identidade visual do sistema e, por consequência, serão observados em todas as páginas da aplicação.

Figura 13 – Página de Login.

MetaOPT

Login Cadastro Redefinição de Senha

Login Bem-vindo, visitante!

Entre em sua conta

Digite seu usuário

Digite sua senha

[Esqueceu sua senha?](#) [Mostrar senha](#)

[Crie uma conta!](#) [Continuar](#)

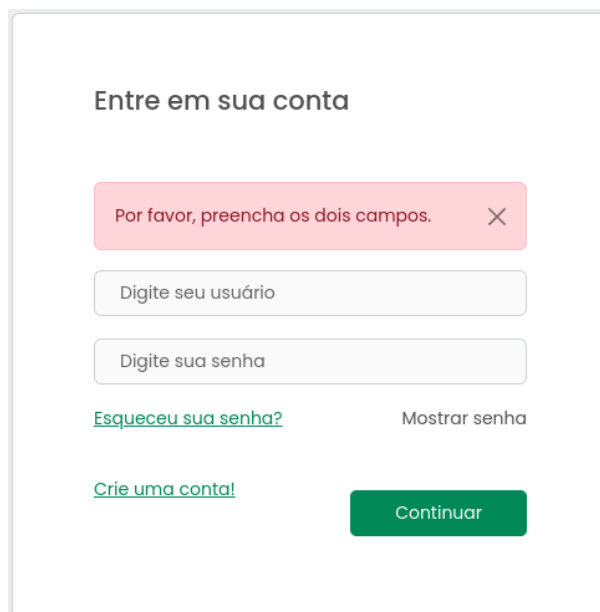
MetaOPT - Meta-Heurísticas em Tarefas de Otimização e de Seleção de Características

Fonte: Elaborada pelo autor.

A Figura 14 expõe a mensagem de erro que é apresentada ao usuário caso ele acione o botão Continuar, mas, não tenha preenchido pelo menos um dos campos.



Figura 14 – Mensagem de Erro: Campos Vazios.



Entre em sua conta

Por favor, preencha os dois campos. ✕

Digite seu usuário

Digite sua senha

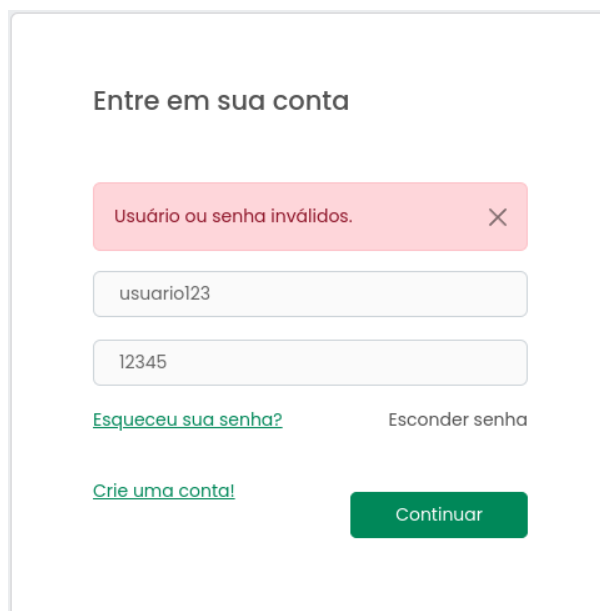
[Esqueceu sua senha?](#) [Mostrar senha](#)

[Crie uma conta!](#) [Continuar](#)

Fonte: Elaborada pelo autor.

Enquanto a Figura 15 introduz a mensagem de erro exibida na hipótese de não haver nenhum usuário cadastrado com essas credenciais digitadas.

Figura 15 – Mensagem de Erro: Usuário Inválido.



Entre em sua conta

Usuário ou senha inválidos. ✕

usuario123

12345

[Esqueceu sua senha?](#) [Esconder senha](#)

[Crie uma conta!](#) [Continuar](#)

Fonte: Elaborada pelo autor.

### 4.2.2 Cadastro

Se, eventualmente, for primeira vez do usuário na aplicação, então ele deverá fazer um cadastro para obter suas credenciais de acesso. A página de cadastro pode ser acessada, como descrito na subseção anterior, pelo link "Cadastro" no menu de navegação ou pelo link "Crie uma conta!" exibido no formulário de login.

De todo modo, o formulário de cadastro desta página, em conformidade com a Figura 16, contém 6 campos, todos de texto, designados para nome e sobrenome, nome de usuário, endereço de e-mail, senha e confirmação de senha. Em adição, há um link para o usuário retornar à página de login e dois outros botões, um para mostrar o conteúdo das senhas e outro, para confirmar o cadastro.

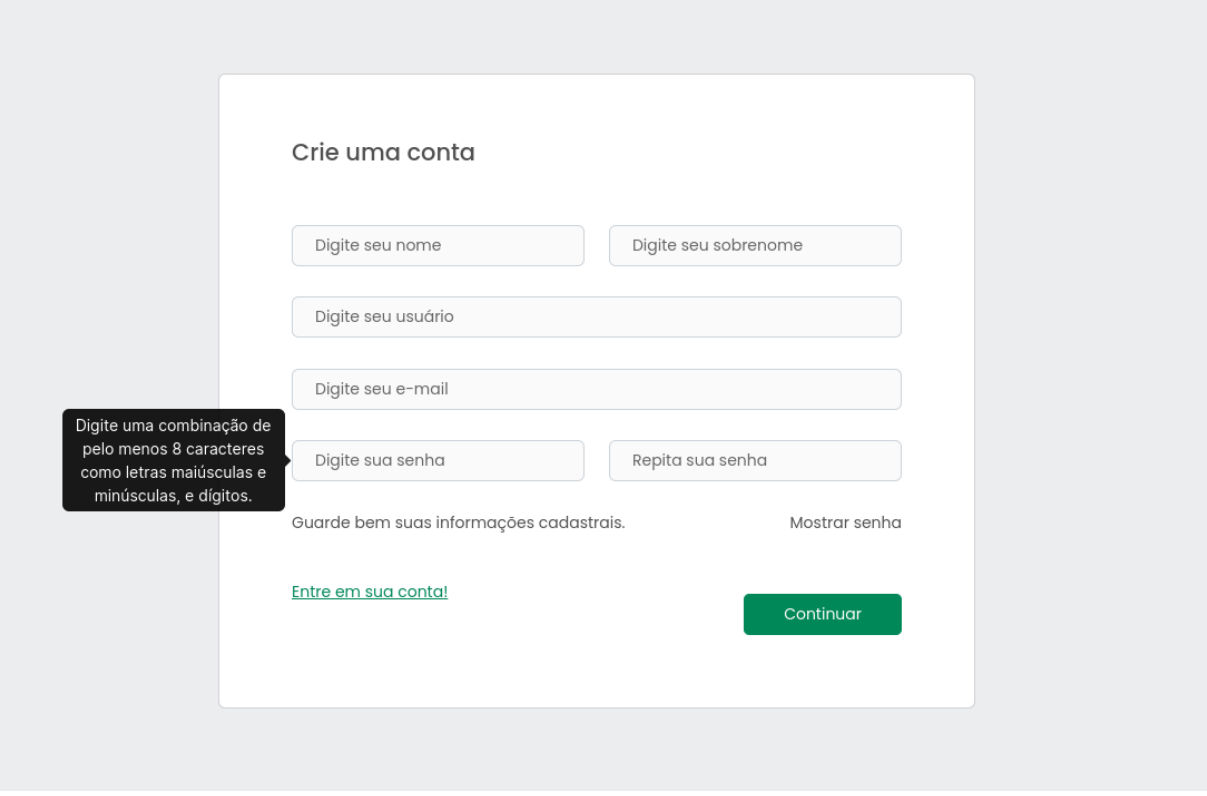
Figura 16 – Página de Cadastro.

A imagem mostra a interface de usuário para a criação de uma conta no sistema MetaOPT. O cabeçalho da página é dividido em duas partes: uma barra superior azul com o nome 'MetaOPT' e links para 'Login', 'Cadastro' e 'Redefinição de Senha'; e uma barra inferior cinza com o link 'Retornar' e a mensagem 'Bem-vindo, visitante!'. O formulário principal, intitulado 'Crie uma conta', é branco e centralizado. Ele contém seis campos de texto: 'Digite seu nome', 'Digite seu sobrenome', 'Digite seu usuário', 'Digite seu e-mail', 'Digite sua senha' e 'Repita sua senha'. Abaixo dos campos de senha, há o texto 'Guarde bem suas informações cadastrais.' e um link 'Mostrar senha'. No rodapé do formulário, há um link 'Entre em sua conta!' e um botão verde 'Continuar'. A barra de rodapé da página é azul e contém o texto 'MetaOPT - Meta-Heurísticas em Tarefas de Otimização e de Seleção de Características'.

Fonte: Elaborada pelo autor.

Dentre os atributos incorporados ao formulário, está a existência de balões de ajuda/dicas, também conhecidos como *tooltips*. Estes balões são exibidos sempre que o usuário estiver com o mouse posicionado sobre algum dos campos do formulário, informando qual é o tipo de entrada aceita. Veja a Figura 17.

Figura 17 – Tooltip.



O formulário, intitulado "Crie uma conta", contém os seguintes campos de entrada:

- Dois campos para "Nome" e "Sobrenome".
- Um campo para "Usuário".
- Um campo para "E-mail".
- Dois campos para "Senha" e "Repita sua senha".

Um tooltip preto com texto branco aponta para o campo de senha, contendo a mensagem: "Digite uma combinação de pelo menos 8 caracteres como letras maiúsculas e minúsculas, e dígitos."

Abaixo dos campos, há o texto "Guarde bem suas informações cadastrais." e um link "Mostrar senha".

No rodapé do formulário, há um link "Entre em sua conta!" e um botão verde "Continuar".

Fonte: Elaborada pelo autor.

De modo semelhante ao formulário de login, também são verificadas se as entradas deste formulário estão sendo corretas. A mensagem de erro mostrada na Figura 18 aparece sempre que um ou mais campos deixarem de ser preenchidos.

Figura 18 – Mensagem de Erro: Campos Vazios.

O formulário, intitulado "Crie uma conta", apresenta uma mensagem de erro global no topo: "Por favor, preencha todos os campos corretamente." com um ícone de fechar. Abaixo, há seis campos de entrada, cada um com uma mensagem de erro específica: "Digite seu nome" (erro: "Por favor, digite seu nome."), "Digite seu sobrenome" (erro: "Por favor, digite seu sobrenome."), "Digite seu usuário" (erro: "Por favor, digite seu usuário."), "Digite seu e-mail" (erro: "Por favor, digite seu e-mail."), "Digite sua senha" (erro: "Por favor, digite sua senha.") e "Repita sua senha" (erro: "Por favor, repita sua senha."). Cada campo possui um ícone de erro (círculo com ponto de exclamação). No rodapé do formulário, há o texto "Guarde bem suas informações cadastrais.", um link "Mostrar senha" e um link "Entre em sua conta!". Um botão verde "Continuar" está posicionado no canto inferior direito.

Fonte: Elaborada pelo autor.

Ainda que a mensagem de erro na parte superior seja a mesma da situação anterior, deve-se atentar na mensagem de erro que aparece abaixo do campo inválido, pois seu objetivo é informar o real motivo da inconsistência na entrada do mesmo. Neste caso, se o usuário inserir um nome de usuário que fere as orientações passadas pelo *tooltip* quanto ao tipo de entrada, uma mensagem como a da Figura 19 aparecerá.

Figura 19 – Mensagem de Erro: Usuário Inválido.

O formulário, intitulado "Crie uma conta", contém os seguintes elementos:

- Uma barra de mensagem de erro em um fundo rosa claro: "Por favor, preencha todos os campos corretamente." com um ícone de fechar (X) à direita.
- Dois campos de texto para nome e sobrenome: "Fulano" e "De tal", ambos com uma marca de verificação verde e o feedback "Parece certo!" abaixo.
- Um campo de texto para o nome de usuário contendo "usuario/tcc". Este campo está destacado com uma borda vermelha e um ícone de erro (círculo com ponto) à direita. Abaixo dele, uma mensagem de erro em vermelho diz: "Informe um nome de usuário válido. Este valor pode conter apenas letras, números e os seguintes caracteres @/./+/\_/-.".
- Um campo de texto para o e-mail contendo "usuario.tcc@gmail.com", com uma marca de verificação verde e o feedback "Parece certo!" abaixo.
- Dois campos de senha: "Digite sua senha" e "Repita sua senha", ambos com uma marca de verificação verde e o feedback "Parece certo!" abaixo.
- Links de navegação: "Guarde bem suas informações cadastrais." e "Mostrar senha" (desativado).
- Links de ação: "Entre em sua conta!" (link azul) e "Continuar" (botão verde).

Fonte: Elaborada pelo autor.

Outros campos que a exibição de mensagens de erro pode ser comum são os relacionados com senha, haja vista que o usuário deve inserir uma senha e confirmá-la no campo seguinte. Neste sentido, caso elas não sejam idênticas, a mensagem ilustrada na Figura 20 será mostrada. Outrossim, há uma série de requisitos que as senhas devem cumprir, visando o aumento da segurança da conta do usuário, como, por exemplo, as senhas devem conter, no mínimo, uma letra maiúscula, uma minúscula e números.

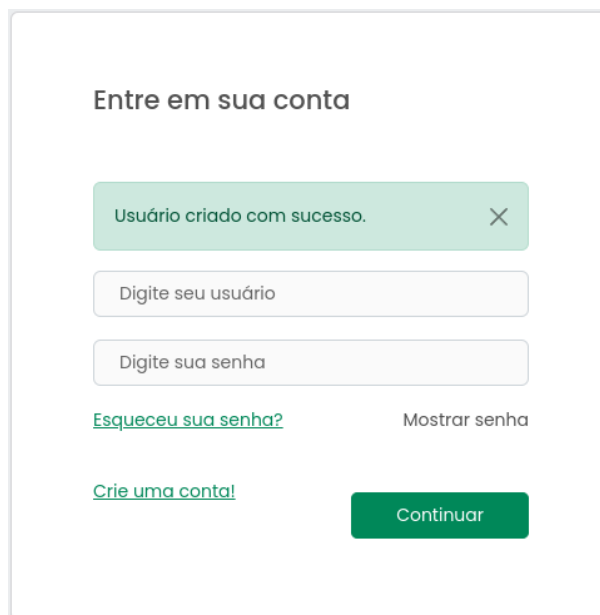
Figura 20 – Mensagem de Erro: Senhas diferentes.

O formulário, intitulado "Crie uma conta", apresenta um cabeçalho de mensagem de erro em uma caixa rosa: "Por favor, preencha todos os campos corretamente." com um ícone de fechar. Abaixo, os campos de entrada são validados individualmente com mensagens de feedback em verde: "Fulano" (Parece certo!), "De tal" (Parece certo!), "usuario.tcc" (Parece certo!) e "usuario.tcc@gmail.com" (Parece certo!). Os campos de senha, ambos com o valor "Tcc12345", mostram uma inconsistência: o primeiro campo tem uma marca de verificação verde, enquanto o segundo possui um ícone de alerta vermelho e uma mensagem de erro em vermelho: "O campo 'Confirmar senha' deve ser igual ao 'Senha'". Na base do formulário, há o texto "Guarde bem suas informações cadastrais.", o link "Esconder senha", o link "Entre em sua conta!" e um botão verde "Continuar".

Fonte: Elaborada pelo autor.

Finalmente, quando todos os campos do formulário de cadastro tiverem sido preenchidos de maneira correta e nenhuma inconsistência verificada, o usuário é direcionado novamente à página de login para preencher as suas credenciais e acessar o sistema, como observado na Figura 21.

Figura 21 – Mensagem: Usuário criado com sucesso.



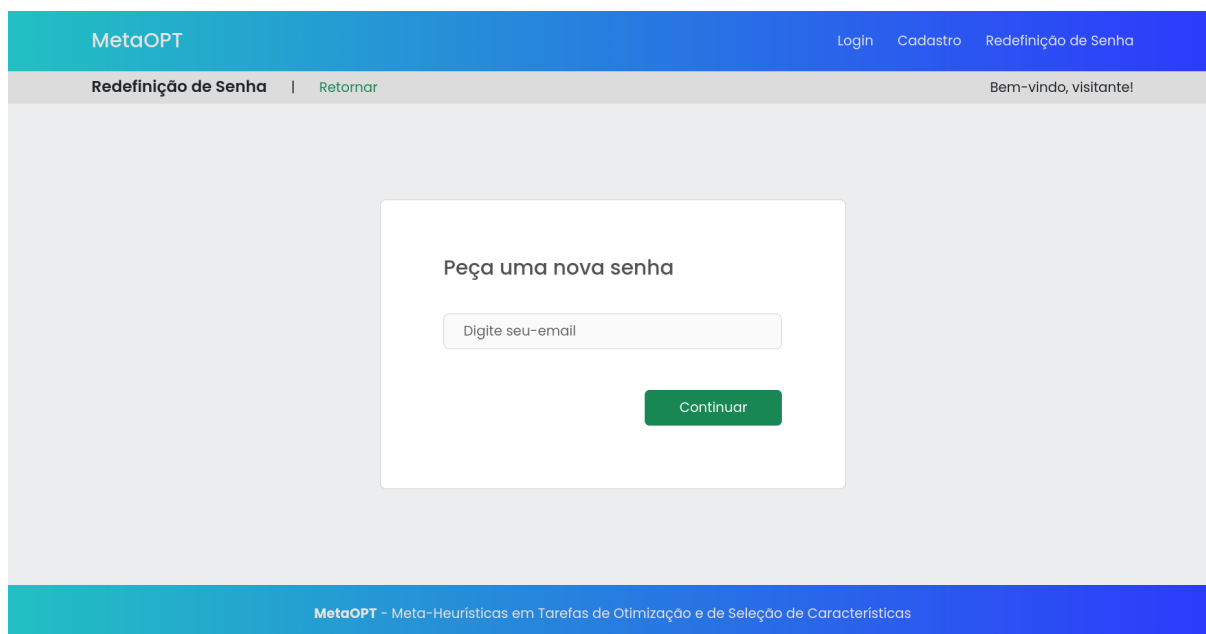
The image shows a login interface titled "Entre em sua conta". At the top, a green success message box states "Usuário criado com sucesso." with a close icon. Below this are two input fields: "Digite seu usuário" and "Digite sua senha". Under the password field, there is a link "Esqueceu sua senha?" and a "Mostrar senha" option. At the bottom left is a link "Crie uma conta!" and at the bottom right is a green "Continuar" button.

Fonte: Elaborada pelo autor.

#### 4.2.3 Redefinição de Senha

Pensando no fato de que o esquecimento da senha possa ser uma situação recorrente para alguns usuários, foi criada a página de redefinição de senha, apresentada na Figura 22. Em síntese, o usuário precisa preencher apenas um campo do formulário, cuja informação a ser fornecida é o e-mail atrelado a conta que se deseja recuperar o acesso.

Figura 22 – Página de Recuperação de Senha.

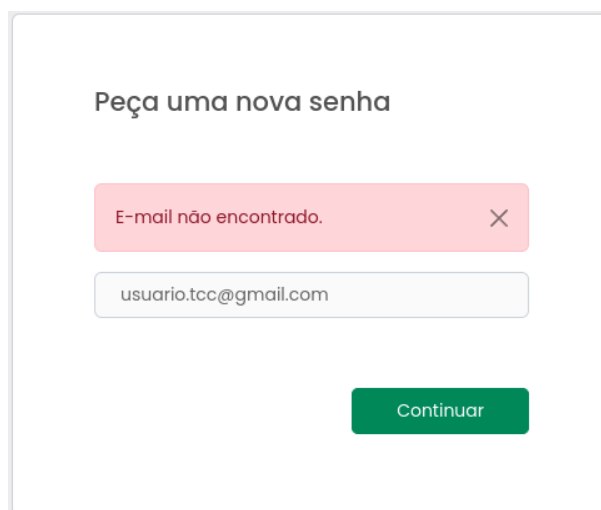


The screenshot shows the MetaOPT website's password recovery interface. At the top, a blue header bar contains the 'MetaOPT' logo on the left and 'Login', 'Cadastro', and 'Redefinição de Senha' links on the right. Below this, a grey navigation bar features 'Redefinição de Senha' as the active page, a 'Retornar' link, and a 'Bem-vindo, visitante!' message. The main content area is light grey and contains a white card with the heading 'Peça uma nova senha'. Inside the card is a text input field with the placeholder 'Digite seu-email' and a green 'Continuar' button. A footer bar at the bottom of the page reads 'MetaOPT - Meta-Heurísticas em Tarefas de Otimização e de Seleção de Características'.

Fonte: Elaborada pelo autor.

Caso verifique-se que o e-mail digitado não esteja atrelado a nenhuma conta, o erro da Figura 23 aparecerá. Entretanto, caso se trate de um endereço válido, uma mensagem, como a da Figura 24, será exibida informando que um e-mail foi enviado ao usuário.

Figura 23 – Mensagem de Erro: E-mail não encontrado.

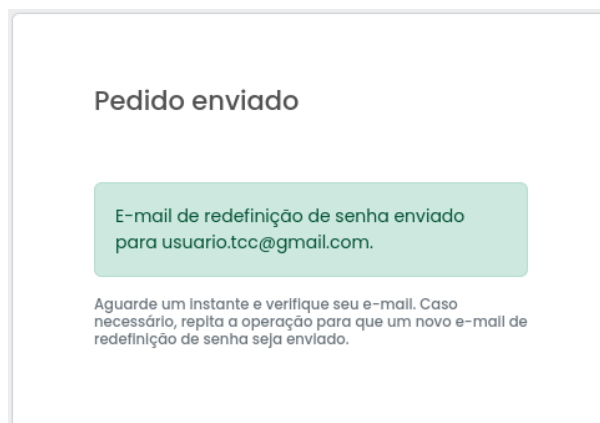


This screenshot shows the same password recovery form as Figure 22, but with an error state. A red rectangular message box at the top of the form contains the text 'E-mail não encontrado.' and a close icon (X). Below the message box, the email input field now contains the text 'usuario.tcc@gmail.com'. The green 'Continuar' button remains at the bottom of the form.

Fonte: Elaborada pelo autor.



Figura 24 – Mensagem: E-mail para redefinição de senha enviado.



Fonte: Elaborada pelo autor.

O e-mail recebido é similar ao que aparece na Figura 25, descrevendo o passo a passo para redefinição da senha. Neste e-mail, encontra-se o link para a página de alteração de senha, o qual estará disponível para acesso por, no máximo, 24 horas a partir do momento do envio.

Figura 25 – E-mail recebido.



Fonte: Elaborada pelo autor.

Ao clicar no link fornecido no e-mail, depara-se com a página de alteração da senha (Figura 26), na qual dois campos serão disponibilizados: o de nova senha e o de confirmação da nova senha. Novamente, há verificação quanto a igualdade das duas senhas informadas.

Figura 26 – Página de Redefinição de Senha.

A imagem mostra a interface de redefinição de senha do sistema MetaOPT. No topo, há uma barra azul com o nome 'MetaOPT' à esquerda e os links 'Login', 'Cadastro' e 'Redefinição de Senha' à direita. Abaixo, uma barra cinza contém 'Redefinição de Senha' e 'Bem-vindo, visitante!'. O formulário centralizado, intitulado 'Redefina sua senha', possui dois campos de entrada: 'Digite a nova senha' e 'Repita sua senha'. Abaixo dos campos, há um link 'Mostrar senha' e um botão verde 'Continuar'. Na base da página, uma barra azul contém o texto 'MetaOPT - Meta-Heurísticas em Tarefas de Otimização e de Seleção de Características'.

Fonte: Elaborada pelo autor.

Uma vez que as senhas tenham sido inseridas corretamente, o usuário será direcionado à página de login e a mensagem de sucesso da Figura 27 é exibida acima dos campos.

Figura 27 – Mensagem: Senha Alterada com Sucesso.

A imagem mostra a interface de login do sistema MetaOPT após uma alteração bem-sucedida de senha. O formulário, intitulado 'Entre em sua conta', apresenta uma mensagem de sucesso em uma caixa verde: 'Senha alterada com sucesso.' com um ícone de fechar (X). Abaixo, há dois campos de entrada: 'Digite seu usuário' e 'Digite sua senha'. À esquerda dos campos, há dois links: 'Esqueceu sua senha?' e 'Crie uma conta!'. À direita, há um link 'Mostrar senha' e um botão verde 'Continuar'.

Fonte: Elaborada pelo autor.

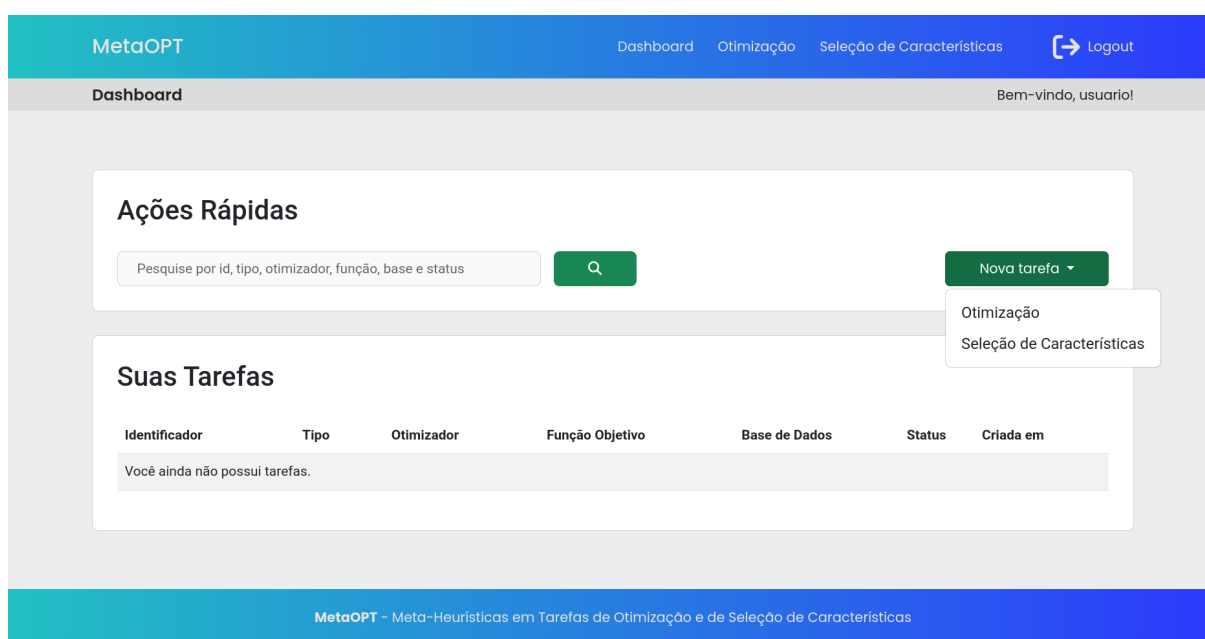
#### 4.2.4 Dashboard

Quando o login é efetuado, o usuário é levado à página da *dashboard*, o qual é estruturada conforme a Figura 28. Trata-se da página principal da aplicação, onde concentram-se todas as informações principais sobre cada uma das tarefas executadas pelo usuário e que interliga todos os módulos. Também, a partir dela, é possível acessar uma tarefa específica e ver todos os detalhes acerca de seu progresso e os resultados preliminares, além de permitir a criação novas tarefas e a consulta de tarefas já iniciadas.

Da página da *dashboard* em diante, houve a modificação do menu de navegação superior quanto aos links nela presentes. Agora, os links de acesso rápido direcionam o usuário para as páginas da dashboard, de criação de novas tarefas de otimização e de criação de novas tarefas de seleção de características, respectivamente, da esquerda para a direita.

A criação de novas tarefas, tanto de otimização quanto de seleção de características, pode ser feita de duas maneiras. A primeira é por meio do link de acesso rápido mencionado no parágrafo anterior e a segunda, pelo botão "Nova Tarefa" presente dentro do primeiro *card*, à direita.

Figura 28 – Página de *Dashboard*



Fonte: Elaborada pelo autor.

Na Figura 29, são mostradas quatro tarefas, cada qual em um estado de execução diferente. Em resumo, uma tarefa pode assumir 4 estados: cancelado, pendente, progresso, sucesso.

- **Cancelado:** Ocorre quando o usuário pressiona o botão "Cancelar" na página de detalhes

da tarefa (vide subseção 4.2.6), parando sua execução;

- **Pendente:** Ocorre quando todos os núcleos do processador estão indisponíveis, indicando que a tarefa está na fila de execução. Também pode ocorrer quando o Celery para de executar por algum motivo;
- **Progresso:** Indica que a tarefa está executando normalmente e ainda não terminou;
- **Sucesso:** Indica que a tarefa terminou de executar e os resultados estão prontos para serem analisados.

Figura 29 – *Dashboard: Status de Tarefa*

Suas Tarefas						
Identificador	Tipo	Otimizador	Função Objetivo	Base de Dados	Status	Criada em
82fb8d0b-06b2-4eff-80b3-c6869799c96d	Seleção	FA	Classificador OPF	Hill Valley	Pendente	24 de Janeiro de 2023 às 15:44
4b67ab85-bf9d-46c2-87d0-c2bc9c3b9ff2	Seleção	ABC	Classificador OPF	Ionosphere	Progresso	24 de Janeiro de 2023 às 15:44
8f4c8cd4-4680-4384-b427-aab810c0c4c1	Seleção	ABC	Classificador OPF	Ionosphere	Sucesso	24 de Janeiro de 2023 às 15:38
f19aa504-5d18-4fc1-89ff-58af5c62b203	Seleção	ABC	Classificador OPF	Ionosphere	Cancelada	24 de Janeiro de 2023 às 15:14

Fonte: Elaborada pelo autor.

#### 4.2.5 Tarefa de Seleção de Características

Nesta subseção é introduzida a página referente à criação de novas tarefas de seleção de características. De acordo com a Figura 30, nesta página encontra-se um formulário para configuração dos parâmetros experimentais da tarefa. Dentre os parâmetros configuráveis, pode-se citar o otimizador, a base de dados, a função de transferência, o número de agentes e de iterações que o otimizador executará, e a quantidade de vezes que a tarefa, com esses parâmetros, irá executar. Ademais, o botão "Iniciar Tarefa", como o próprio nome informa, serve para inicializar a tarefa.

Figura 30 – Página da Tarefa de Seleção de Características.

## Configure sua tarefa

A funcionalidade de seleção de características do MetaOPT emprega as meta-heurísticas mais populares da literatura para encontrar o vetor de características que minimiza o erro de classificação. O classificador de padrões utilizado foi o OPF (Optimum Path-Forest) e as bases de dados pertencem ao repositório UCI Machine Learning que foram escolhidas devido à sua relevância na literatura. Ademais, como nenhuma versão binária das técnicas de otimização foi implementada, o uso de funções de transferências se fez necessário.

Otimizador:

Selecione um otimizador

Base de Dados:

Selecione uma base

Função de Transferência:

Selecione uma função

Número de Agentes:

10

Número de Iterações:

50

Número de Execuções:

1

Iniciar tarefa

Fonte: Elaborada pelo autor.

Abaixo, na Figura 31 exibe-se a lista de otimizadores escolhidos segundo os critérios explicados na seção 3.1.

Figura 31 – Otimizadores Seleccionados.

## Configure sua tarefa

A funcionalidade de seleção de características do MetaOPT emprega as meta-heurísticas mais populares da literatura para encontrar o vetor de características que minimiza o erro de classificação. O classificador de padrões utilizado foi o OPF (Optimum Path-Forest) e as bases de dados pertencem ao repositório UCI Machine Learning que foram escolhidas devido à sua relevância na literatura. Ademais, como nenhuma versão binária das técnicas de otimização foi implementada, o uso de funções de transferências se fez necessário.

Otimizador:

Selecione um otimizador

Selecione um otimizador

Algoritmo Genético (GA)  
Algoritmo do Vaga-lume (FA)  
Busca Cuco (CS)  
Colônia Artificial de Abelhas (ABC)  
Otimização por Enxame de Partículas (PSO)  
Recozimento Simulado (SA)

Escolha uma das meta-heurísticas de otimização.

Número de Agentes:

10

Número de Iterações:

50

Número de Execuções:

1

Iniciar tarefa

Fonte: Elaborada pelo autor.

Abaixo, na Figura 32 exibe-se a lista de bases de dados escolhidas conforme os critérios explicados na seção 3.5. É cabível informar que, no momento em que o usuário seleciona uma das bases disponíveis, o número de características que ela possui aparece abaixo do campo. A escolha das funções de transferência, sugerida pela Figura 33, funciona de modo semelhante, contudo, a informação exibida abaixo do campo é a sua respectiva expressão LaTeX, conforme pode ser visto na Figura 34.

Figura 32 – Bases de Dados Seleccionadas.

## Configure sua tarefa

A funcionalidade de seleção de características do MetaOPT emprega as meta-heurísticas mais populares da literatura para encontrar o vetor de características que minimiza o erro de classificação. O classificador de padrões utilizado foi o OPF (Optimum Path-Forest) e as bases de dados pertencem ao repositório UCI Machine Learning que foram escolhidas devido à sua relevância na literatura. Ademais, como nenhuma versão binária das técnicas de otimização foi implementada, o uso de funções de transferências se fez necessário.

Otimizador:

Selecione um otimizador

Base de Dados:

Selecione uma base

- Selecione uma base
- Heart Statlog
- Hill Valley
- Ionosphere
- Sonar
- Spambase
- Vehicle
- Waveform

Número de Iterações:

50

Número de Execuções:

1

Iniciar tarefa

Escolha uma das bases para a tarefa de seleção de características.

Fonte: Elaborada pelo autor.

Figura 33 – Funções de Transferências Seleccionadas.

## Configure sua tarefa

A funcionalidade de seleção de características do MetaOPT emprega as meta-heurísticas mais populares da literatura para encontrar o vetor de características que minimiza o erro de classificação. O classificador de padrões utilizado foi o OPF (Optimum Path-Forest) e as bases de dados pertencem ao repositório UCI Machine Learning que foram escolhidas devido à sua relevância na literatura. Ademais, como nenhuma versão binária das técnicas de otimização foi implementada, o uso de funções de transferências se fez necessário.

Otimizador:

Selecione um otimizador

Base de Dados:

Selecione uma base

Função de Transferência:

Selecione uma função

- Selecione uma função
- S1
- S2
- S3
- S4
- V1
- V2
- V3
- V4

Número de Execuções:

1

Iniciar tarefa

Fonte: Elaborada pelo autor.



Figura 34 – Expressão LaTeX: Função de Transferência.

### Configure sua tarefa

A funcionalidade de seleção de características do MetaOPT emprega as meta-heurísticas mais populares da literatura para encontrar o vetor de características que minimiza o erro de classificação. O classificador de padrões utilizado foi o OPF (Optimum Path-Forest) e as bases de dados pertencem ao repositório UCI Machine Learning que foram escolhidas devido à sua relevância na literatura. Ademais, como nenhuma versão binária das técnicas de otimização foi implementada, o uso de funções de transferências se fez necessário.

Otimizador:

Algoritmo do Vaga-lume (FA) ▾

Base de Dados:

Hill Valley ▾

100 características

Função de Transferência:

S1 ▾

$$f(x_i) = \frac{1}{1+e^{-x_i}}$$

Número de Agentes:

30

Número de Iterações:

500

Número de Execuções:

25

Iniciar tarefa

Fonte: Elaborada pelo autor.

Para mais, a verificação para campos não preenchidos também foi aplicada neste formulário. Na Figura 35, é mostrada a mensagem de erro decorrente de campos vazios.

Figura 35 – Mensagem de Erro: Campo Vazio.

## Configure sua tarefa

A funcionalidade de seleção de características do MetaOPT emprega as meta-heurísticas mais populares da literatura para encontrar o vetor de características que minimiza o erro de classificação. O classificador de padrões utilizado foi o OPF (Optimum Path-Forest) e as bases de dados pertencem ao repositório UCI Machine Learning que foram escolhidas devido à sua relevância na literatura. Ademais, como nenhuma versão binária das técnicas de otimização foi implementada, o uso de funções de transferências se fez necessário.

Por favor, selecione todas as opções e insira valores válidos. X

Otimizador:  
Selecione um otimizador

Base de Dados:  
Selecione uma base

Função de Transferência:  
Selecione uma função

Número de Agentes:  
10

Número de Iterações:  
50

Número de Execuções:  
1

Iniciar tarefa

Fonte: Elaborada pelo autor.

#### 4.2.6 Detalhes da Tarefa

Ao iniciar uma tarefa, o usuário é levado à página de detalhes da tarefa, cuja estrutura é formada por 4 seções/áreas: descrição, progresso, parâmetros de execução e resultados.

Na Figura 36, encontra-se a seção de descrição, que apresenta algumas informações da tarefa, tais como seu identificador e o seu tipo. Na próxima seção, ilustrada na Figura 37,

encontram-se posicionados a barra de progresso e o botão "Cancelar", para interromper a execução da tarefa. Mais abaixo, localiza-se a área designada para exibir um resumo dos parâmetros de execução da tarefa, os quais foram definidos na página de nova tarefa. Inicialmente, esta se encontra fechada, mas pode ser aberta por meio de um clique. Ao final, está localizada a seção de resultados, onde serão exibidos os resultados preliminares da tarefa assim que ela termina e o botão "Ver Mais", que ao ser clicado direcionará o usuário para a página de resultados detalhados, para ver todos os resultados numéricos e gráficos, conforme mostrado na Figura 38.

Figura 36 – Página de Detalhes da Tarefa - Parte 1.

### Descrição

Identificador	8f4c8cd4-4680-4384-b427-aab810c0c4c1
Tipo	Seleção de Características

### Progresso

A execução acontece no servidor, logo você pode continuar navegando ou desconectar-se.

6 de 50 iterações. Tarefa em execução...

Cancelar

Fonte: Elaborada pelo autor.

Figura 37 – Página de Detalhes da Tarefa - Parte 2.

### Parâmetros de Execução

Otimizador	Colônia Artificial de Abelhas (ABC)
Função objetivo	Classificador OPF
Espaço de busca	<b>Dimensão:</b> 34 <b>Limite inferior:</b> 0 para todas as variáveis <b>Limite superior:</b> 1 para todas as variáveis
Base de dados	Ionosphere
Função de transferência	V1
Número de agentes	10
Número de iterações	50
Número de execuções	1

### Resultados

Se tudo ocorrer bem, os resultados aparecerão aqui após a execução.

Fonte: Elaborada pelo autor.

Figura 38 – Apresentação do resultado quando a tarefa termina com sucesso.

### Resultados

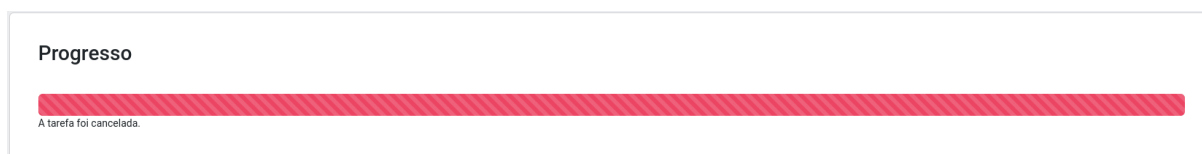
Melhor vetor de características	[0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1]
Quantidade selecionada	10
Melhor acurácia	0,873

Ver mais

Fonte: Elaborada pelo autor.

Enquanto uma tarefa estiver executando normalmente, a barra de progresso estará na cor azul. No entanto, se a tarefa for cancelada, a cor dela é alterada para vermelho, conforme a Figura 39, para indicar o fim da execução. Do mesmo modo, quando a tarefa termina de executar com sucesso, a cor da barra se torna verde, como ilustrado na Figura 40.

Figura 39 – Barra de Progresso: Cancelado.



Fonte: Elaborada pelo autor.

Figura 40 – Barra de Progresso: Sucesso.



Fonte: Elaborada pelo autor.

#### 4.2.7 Resultados

Como mencionado na seção anterior, o botão "Ver Mais" leva o usuário até a página de resultados. As informações contidas nesta página dependem do número de execuções independentes, definida pelo usuário no momento da criação da tarefa.

Se o valor escolhido para esse parâmetro for 1, então as informações que serão exibidas seguirão em consonância a subseção 4.2.7.1. Caso forem escolhidas 2 ou mais execuções, então a página mostrará também resultados estatísticos, como melhores e piores valores, bem como o valor médio para todas as métricas, assim como detalhado na subseção 4.2.7.2. Ademais, como o valor médio, quando considerado de forma isolada, não figura como uma métrica confiável, decidiu-se por apresentar também uma medida de dispersão, o desvio padrão.

Via de regra, a estrutura da página de resultados mantém-se a mesma independentemente do número de execução, sendo composta pelas seções de otimização, de seleção de características, de gráfico de distribuição e de gráfico de convergência.

##### 4.2.7.1 Única Execução

Na Figura 41, mostra-se os resultados do processo de otimização meta-heurística. Estes contemplam o melhor vetor solução real encontrado, isto é, o vetor antes de ser transportado para o espaço binário pela função de transferência, e o valor da função objetivo quando aplicado o este vetor.

Figura 41 – Resultados da Otimização.

Otimização	
Solução	
Melhor solução	[0,086; 0,108; 0,969; 0,918; 0,119; 0,082; 0,914; 0,086; 0,086; 0,886; 0,889; 0,902; 0,978]
Função	
Melhor valor	1,480e-01

Fonte: Elaborada pelo autor.

Logo abaixo, na Figura 42, estão dispostos os resultados referentes às métricas de classificação. Primeiramente, é exibido o vetor de características ótimo e a quantidade de características selecionadas, depois os valores da acurácia, *precision*, *recall* e *f1-score*. No caso das três últimas métricas citadas, os resultados são gerados em termos de cada classe presente no problema.

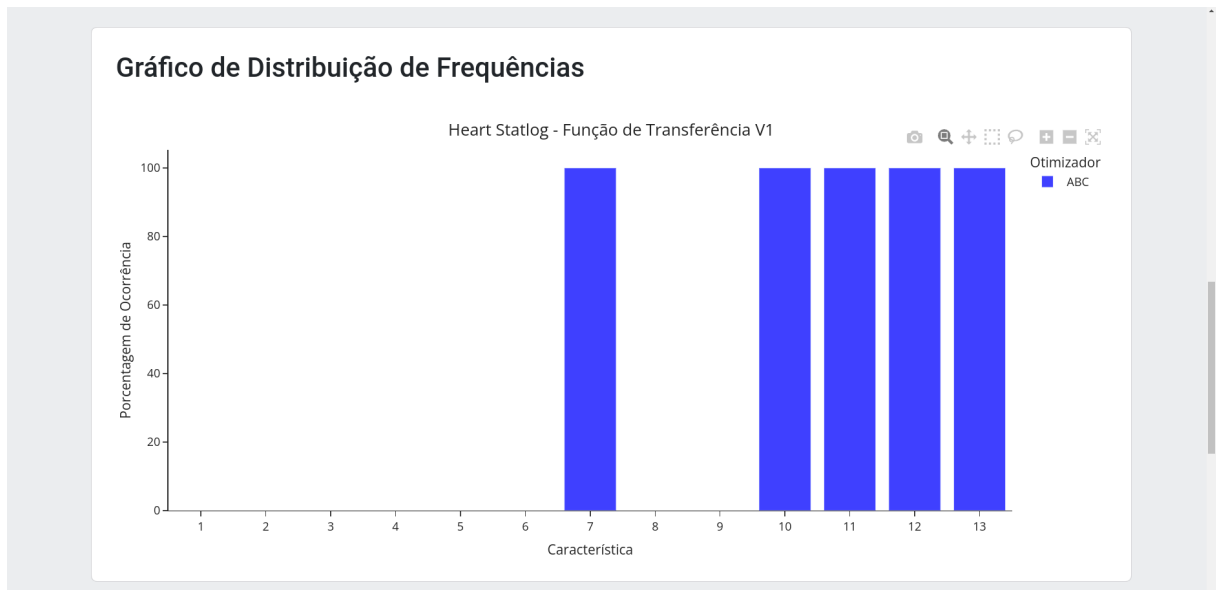
Figura 42 – Valores das Métricas de Classificação.

Seleção de Características	
Vetor de Características	
Melhor vetor	[0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1]
Quantidade selecionada	5
Acurácia	
Melhor valor	0,856
Precision	
Valor	Classe 1: 0,756 Classe 2: 0,857
Recall	
Valor	Classe 1: 0,903 Classe 2: 0,667
F1-Score	
Valor	Classe 1: 0,823 Classe 2: 0,750

Fonte: Elaborada pelo autor.

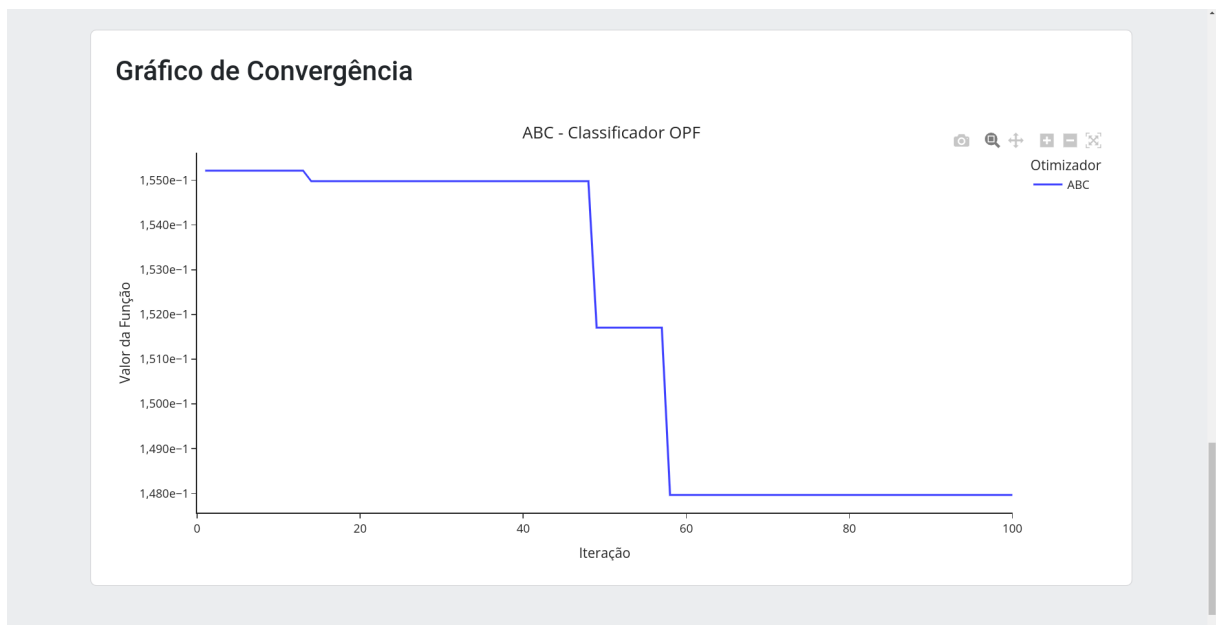
Findados os resultados numéricos, iniciam-se as seções de gráficos. Inicialmente, é gerado o gráfico de distribuição de frequências, cujo objetivo é representar a porcentagem de ocorrências de cada característica do problema, conforme a Figura 43. Após, na seção seguinte ilustrada na Figura 44, é exibido o gráfico de convergência decorrente do processo de otimização meta-heurística, aplicado para encontrar o vetor de características ótimo.

Figura 43 – Gráfico de Distribuição de Frequência por Característica.



Fonte: Elaborada pelo autor.

Figura 44 – Gráfico de Convergência.



Fonte: Elaborada pelo autor.

#### 4.2.7.2 Múltiplas Execuções

Para múltiplas execuções, acrescenta-se aos resultados os valores médios de cada métrica, bem os melhores e piores valores das execuções, além do desvio padrão. Os resultados

referentes à parte de otimização do módulo estão presentes na Figura 45, enquanto os das métricas de classificação encontram-se na Figura 46 e Figura 47.

Figura 45 – Resultados da Otimização.

Otimização	
Solução	
Melhor solução	[0,915; 0,379; 0,686; 0,000; 0,840; 0,000; 1,000; 0,000; 1,000; 0,000; 0,551; 0,387; 0,000; 0,486; 0,080; 0,011; 0,169; 0,126]
Função	
Melhor valor	1,481e-01
Pior valor	2,210e-01
Valor médio	1,824e-01
Desvio padrão	3,666e-02

Fonte: Elaborada pelo autor.

Figura 46 – Valores das Métricas de Classificação.

Seleção de Características	
Vetor de Características	
Melhor vetor	[0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0]
Quantidade selecionada	9
Acurácia	
Melhor valor	0,832
Pior valor	0,789
Valor médio	0,805
Desvio padrão	0,023
Precision	
Valor médio	Classe 1: 0,441 Classe 2: 0,459 Classe 3: 0,838 Classe 4: 0,890
Desvio Padrão	Classe 1: 0,037 Classe 2: 0,019 Classe 3: 0,042 Classe 4: 0,052

Fonte: Elaborada pelo autor.



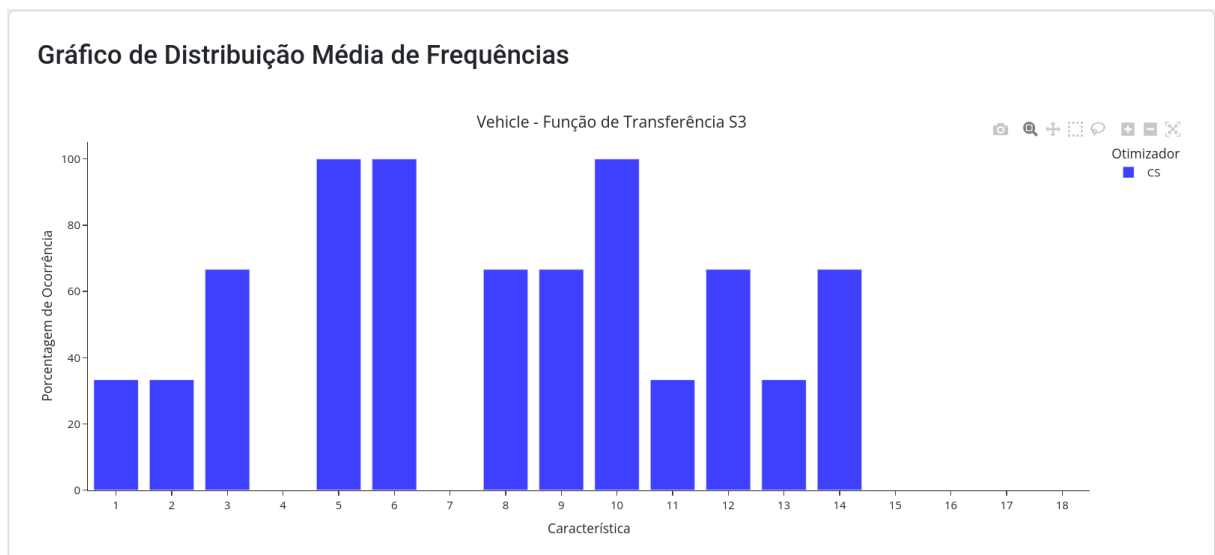
Figura 47 – Valores das Métricas de Classificação.

<b>Recall</b>	
<b>Valor médio</b>	Classe 1: 0,475 Classe 2: 0,513 Classe 3: 0,779 Classe 4: 0,773
<b>Desvio Padrão</b>	Classe 1: 0,055 Classe 2: 0,073 Classe 3: 0,101 Classe 4: 0,049
<b>F1-Score</b>	
<b>Valor médio</b>	Classe 1: 0,453 Classe 2: 0,483 Classe 3: 0,805 Classe 4: 0,826
<b>Desvio Padrão</b>	Classe 1: 0,019 Classe 2: 0,044 Classe 3: 0,066 Classe 4: 0,039

Fonte: Elaborada pelo autor.

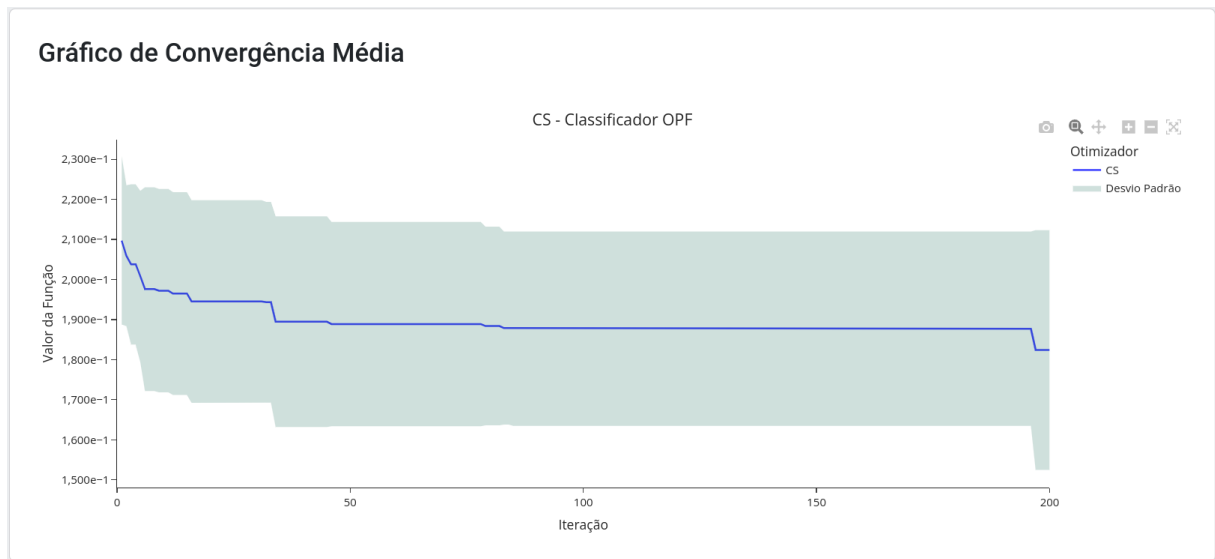
Com relação aos gráficos, a alteração está apenas no fato de que os pontos plotados são representativos dos valores médios. Diante disso, o gráfico de distribuição da Figura 48 agora utiliza a frequência média de ocorrência de cada uma das características, e o gráfico de convergência da Figura 49 representa o valor da função médio de cada iteração. Ainda, neste gráfico é possível observar a plotagem do desvio padrão, ilustrada pelo sombreamento atrás da curva do gráfico.

Figura 48 – Gráfico de Distribuição Média de Frequências por Característica.



Fonte: Elaborada pelo autor.

Figura 49 – Gráfico de Convergência Média.

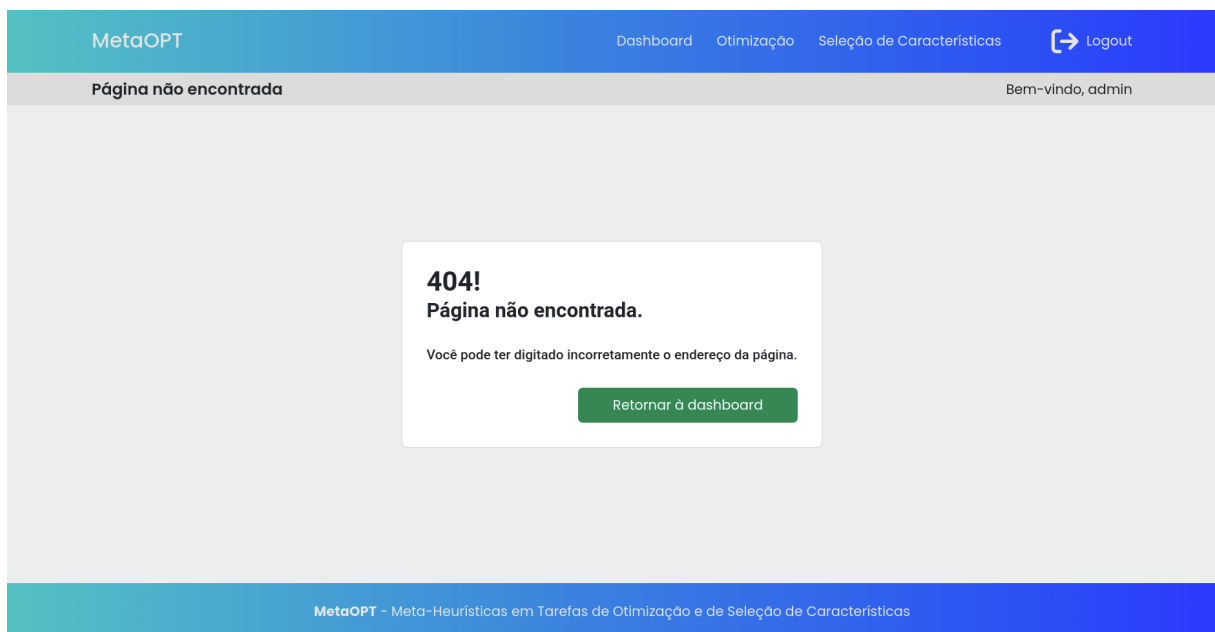


Fonte: Elaborada pelo autor.

#### 4.2.8 Erro 404

Por fim, nesta subseção, apresenta-se a página de erro 404, para qual o usuário é direcionado quando tenta acessar alguma página que não existe dentro do domínio de aplicação. Observe na Figura 50.

Figura 50 – Página de Erro 404



Fonte: Elaborada pelo autor.

## 5 Conclusão

A ampla exploração dos métodos de aprendizado de máquina tem proporcionado o surgimento de novas técnicas e abordagens para todas as etapas do sistema de visão computacional. Neste trabalho em questão, explorou-se a etapa de seleção de características, em especial aplicando a abordagem *wrapper*. Embora muitos estudos já tenham sido realizados neste sentido, o presente trabalho difere-se pela utilização de otimizadores meta-heurísticos, aliados ao classificador de padrões baseado em Floresta de Caminhos Ótimos.

Entretanto, aplicações para cumprir essa tarefa são escassas na literatura, as quais destacam-se apenas aplicações *desktop* ou *frameworks*. No caso desta última, há ainda a desvantagem de que apenas usuários com conhecimentos em programação possam utilizá-los. Dessa necessidade, o projeto MetaOPT foi proposto, visando facilitar os experimentos realizados por pesquisadores e entusiastas, excluindo a necessidade de programar ou instalar dependências.

Para o desenvolvimento da aplicação, estudou-se as técnicas de seleção de características presentes na literatura, bem como definiu-se em qual destas o projeto proposto faz parte. Ainda, foram identificadas e implementadas as bases e técnicas meta-heurísticas mais populares para este tipo de tarefa. A partir dessas informações, foi desenvolvida uma aplicação web que permite a execução das tarefas de maneira assíncrona, possibilitando o acompanhamento em tempo real quanto do progresso de execução e a geração de resultados, os quais foram alicerçados nas métricas mais importantes de classificação de padrões, além da construção de gráficos de distribuição de frequências e de convergência.

### 5.1 Trabalhos Futuros

Uma vez constatada a eficiência e as facilidades proporcionadas pelos módulos do projeto MetaOPT, deseja-se realizar a hospedagem da aplicação em servidor remoto, bem como sua divulgação, à priori, para os integrantes do grupo de pesquisa Recogna, o qual o autor deste trabalho faz parte e, posteriormente, à comunidade científica potencialmente por meio de artigos.

Tomando-se em conta que um projeto de conclusão de curso é limitado em 1 ano, muitas funcionalidades úteis não puderam ser implementadas. Dessa forma, pretende-se deliberar sobre a adição de uma ferramenta de comparação automática de tarefas, o qual o usuário selecionaria as tarefas com parâmetros de execução idênticos e todas os resultados seriam exibidos de na forma de tabelas e de gráficos. No caso deste último, para cada tipo, os resultados seriam unidos e apresentados em único gráfico.

## Referências

- AN, J. *How to Remember all these Classification Concepts Forever*. 2020. Disponível em: <<https://medium.com/swlh/how-to-remember-all-these-classification-concepts-forever-761c065be33>>. Acesso em: 04 jan. 2023.
- BOUZOUBAA, K.; TAHER, Y.; NSIRI, B. Predicting DOS-DDOS Attacks: Review and Evaluation Study of Feature Selection Methods based on Wrapper Process. *International Journal of Advanced Computer Science and Applications*, v. 12, n. 5, p. 132–145, 2021. ISSN 21565570.
- BUITINCK, L.; LOUPPE, G.; BLONDEL, M.; PEDREGOSA, F.; MUELLER, A.; GRISEL, O.; NICULAE, V.; PRETTENHOFER, P.; GRAMFORT, A.; GROBLER, J.; LAYTON, R.; VANDERPLAS, J.; JOLY, A.; HOLT, B.; VAROQUAUX, G. API design for machine learning software: experiences from the scikit-learn project. In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. [S.l.: s.n.], 2013. p. 108–122.
- BUTCHER, B.; SMITH, B. J. Feature Engineering and Selection: A Practical Approach for Predictive Models: by Max Kuhn and Kjell Johnson. Boca Raton, FL: Chapman & Hall/CRC Press, 2019, xv + 297 pp., \$79.95(H), ISBN: 978-1-13-807922-9. *The American Statistician*, v. 74, n. 3, p. 229–231, jul. 2020. ISSN 0003-1305, 1537-2731. Disponível em: <<https://www.tandfonline.com/doi/full/10.1080/00031305.2020.1790217>>. Acesso em: 04 jan. 2023.
- DOKEROGLU, T.; DENIZ, A.; KIZILOZ, H. E. A comprehensive survey on recent metaheuristics for feature selection. *Neurocomputing*, v. 494, p. 269–296, jul. 2022. ISSN 09252312. Disponível em: <<https://linkinghub.elsevier.com/retrieve/pii/S092523122200474X>>. Acesso em: 15 dez. 2022.
- DUA, D.; GRAFF, C. *UCI Machine Learning Repository*. 2017. Disponível em: <<http://archive.ics.uci.edu/ml>>. Acesso em: 04 jan. 2023.
- HOLLAND, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. [S.l.]: MIT press, 1992.
- JOVIC, A.; BRKIC, K.; BOGUNOVIC, N. A review of feature selection methods with applications. In: *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. Opatija, Croatia: IEEE, 2015. p. 1200–1205. ISBN 978-953-233-082-3. Disponível em: <<http://ieeexplore.ieee.org/document/7160458/>>. Acesso em: 02 jan. 2023.
- KARABOGA, D.; BASTURK, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, v. 39, n. 3, p. 459–471, out. 2007. ISSN 0925-5001, 1573-2916. Disponível em: <<http://link.springer.com/10.1007/s10898-007-9149-x>>. Acesso em: 02 jan. 2023.
- KENNEDY, J. F.; EBERHART, R. C.; SHI, Y. *Swarm intelligence*. San Francisco: Morgan Kaufmann Publishers, 2001. (The Morgan Kaufmann series in evolutionary computation). ISBN 978-1-55860-595-4.

KHACHATURYAN, A.; SEMENOVSOVSKAYA, S.; VAINSHTEIN, B. The thermodynamic approach to the structure analysis of crystals. *Acta Crystallographica Section A*, v. 37, n. 5, p. 742–754, set. 1981. ISSN 0567-7394. Disponível em: <<https://scripts.iucr.org/cgi-bin/paper?S0567739481001630>>. Acesso em: 02 jan. 2023.

KOHAVI, R.; JOHN, G. H. Wrappers for feature subset selection. *Artificial Intelligence*, v. 97, n. 1, p. 273–324, 1997. ISSN 0004-3702. Relevance. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S000437029700043X>>. Acesso em: 02 jan. 2023.

MASOUDI-SOBHANZADEH, Y.; MOTIEGHADER, H.; MASOUDI-NEJAD, A. FeatureSelect: a software for feature selection based on machine learning approaches. *BMC Bioinformatics*, v. 20, n. 1, p. 170, abr. 2019. ISSN 1471-2105. Disponível em: <<https://doi.org/10.1186/s12859-019-2754-0>>. Acesso em: 04 jan. 2023.

MOHAMED, A.-A. A.; HASSAN, S.; HEMEIDA, A.; ALKHALAF, S.; MAHMOUD, M.; ELDIN, A. M. B. Parasitism – Predation algorithm (PPA): A novel approach for feature selection. *Ain Shams Engineering Journal*, v. 11, n. 2, p. 293–308, jun. 2020. ISSN 20904479. Disponível em: <<https://linkinghub.elsevier.com/retrieve/pii/S2090447919301406>>. Acesso em: 04 jan. 2023.

PAPA, J. P.; FALCÃO, A. X.; SUZUKI, C. T. N. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, v. 19, n. 2, p. 120–131, jun. 2009. ISSN 08999457, 10981098. Disponível em: <<https://onlinelibrary.wiley.com/doi/10.1002/ima.20188>>. Acesso em: 02 jan. 2023.

PONTI, M. P.; PAPA, J. P. Improving Accuracy and Speed of Optimum-Path Forest Classifier Using Combination of Disjoint Training Subsets. In: SANSONE, C.; KITTLER, J.; ROLI, F. (Ed.). *Multiple Classifier Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. v. 6713, p. 237–248. ISBN 978-3-642-21556-8 978-3-642-21557-5. Series Title: Lecture Notes in Computer Science. Disponível em: <[http://link.springer.com/10.1007/978-3-642-21557-5\\_26](http://link.springer.com/10.1007/978-3-642-21557-5_26)>. Acesso em: 02 jan. 2023.

REMESEIRO, B.; BOLON-CANEDO, V. A review of feature selection methods in medical applications. *Computers in Biology and Medicine*, v. 112, p. 103375, 2019. ISSN 0010-4825. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0010482519302525>>. Acesso em: 02 jan. 2023.

ROSA, G. H. d.; PAPA, J. P. OPFython: A Python implementation for Optimum-Path Forest. *Software Impacts*, v. 9, ago. 2021. ISSN 2665-9638. Publisher: Elsevier. Disponível em: <[https://www.softwareimpacts.com/article/S2665-9638\(21\)00044-0/fulltext](https://www.softwareimpacts.com/article/S2665-9638(21)00044-0/fulltext)>. Acesso em: 02 jan. 2023.

ROSA, G. H. de; RODRIGUES, D.; PAPA, J. P. Opytimizer: A Nature-Inspired Python Optimizer. arXiv, 2019. Disponível em: <<https://arxiv.org/abs/1912.13002v2>>. Acesso em: 02 jan. 2023.

SAMMUT, C.; WEBB, G. I. (Ed.). *Encyclopedia of Machine Learning and Data Mining*. Boston, MA: Springer US, 2017. ISBN 978-1-4899-7685-7 978-1-4899-7687-1. Disponível em: <<http://link.springer.com/10.1007/978-1-4899-7687-1>>. Acesso em: 20 dez. 2022.

SERPICO, S. B.; BRUZZONE, L. A new search algorithm for feature selection in hyperspectral remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, v. 39, n. 7, p. 1360–1367, jul 2001. ISSN 01962892.

TOO, J.; ABDULLAH, A. R. Binary atom search optimisation approaches for feature selection. *Connection Science*, Taylor & Francis, v. 32, n. 4, p. 406–430, 2020. Disponível em: <https://doi.org/10.1080/09540091.2020.1741515>. Acesso em: 02 jan. 2023.

WANG, S.; TANG, J.; LIU, H. Feature selection. In: *Encyclopedia of Machine Learning and Data Mining*. [s.n.], 2016. p. 1–9. Disponível em: [https://www.researchgate.net/publication/308879182\\_Feature\\_Selection](https://www.researchgate.net/publication/308879182_Feature_Selection). Acesso em: 03 jan. 2023.

YANG, X.-S. Firefly Algorithms for Multimodal Optimization. In: WATANABE, O.; ZEUGMANN, T. (Ed.). *Stochastic Algorithms: Foundations and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. v. 5792, p. 169–178. ISBN 978-3-642-04943-9 978-3-642-04944-6. Series Title: Lecture Notes in Computer Science. Disponível em: [http://link.springer.com/10.1007/978-3-642-04944-6\\_14](http://link.springer.com/10.1007/978-3-642-04944-6_14). Acesso em: 02 jan. 2023.

YANG, X.-S. *Nature-inspired optimization algorithms*. First edition. Amsterdam ; Boston: Elsevier, 2014. OCLC: ocn866615538. ISBN 978-0-12-416743-8.

YANG, X.-S.; Suash Deb. Cuckoo Search via Lévy flights. In: *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*. Coimbatore, India: IEEE, 2009. p. 210–214. ISBN 978-1-4244-5053-4. Disponível em: <http://ieeexplore.ieee.org/document/5393690/>. Acesso em: 02 jan. 2023.