



**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ -
IFCE
CAMPUS DE FORTALEZA**

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELECOMUNICAÇÕES

FRANCISCO JORGE COSTA RIBEIRO

**COMPARAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA EM DADOS DE
INTENSIDADE DE SINAIS *WI-FI* PARA LOCALIZAÇÃO DE ATIVOS**

**FORTALEZA
2022**

FRANCISCO JORGE COSTA RIBEIRO

**COMPARAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA EM DADOS DE
INTENSIDADE DE SINAIS *WI-FI* PARA LOCALIZAÇÃO DE ATIVOS**

Dissertação aprovada, apresentada ao curso de Mestrado em Engenharia de Telecomunicações do Instituto Federal de Educação, Ciência e Tecnologia do Ceará – IFCE – Campus Fortaleza, como requisito para obtenção do Título de Mestre.

Orientador: Prof. Dr. Auzuir Ripardo de Alexandria

FORTALEZA

2022

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

- R484c Ribeiro, Francisco Jorge Costa.
 Comparação de Técnicas de Aprendizado de máquina em dados de intensidade de sinais Wi-Fi para
 localização de ativos / Francisco Jorge Costa Ribeiro. - 2022.
 54 f. : il. color.
- Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Engenharia de Telecomunicações,
 Campus Fortaleza, 2022.
 Orientação: Prof. Dr. Auzuir Ripardo de Alexandria.
1. Robótica. 2. Localização de ativos. 3. Wi-Fi/Bluetooth. 4. Aprendizado de máquina. 5. Beacons. I.
 Titulo.

CDD 621.382



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
Av. 13 de maio, 2081 , - Bairro Benfica - CEP 60040-531 - Fortaleza - CE - www.ifce.edu.br

AVALIAÇÃO

Processo: 23256.012382/2022-31

Interessado: Francisco Jorge Costa Ribeiro

FRANCISCO JORGE COSTA RIBEIRO

COMPARAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA EM DADOS DE INTENSIDADE DE SINAIS WIFI PARA LOCALIZAÇÃO DE ATIVOS

Dissertação apresentada ao curso de Pós-Graduação em Engenharia de Telecomunicações do Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE) - *Campus Fortaleza*, como requisito parcial para obtenção do Título de Mestre em Engenharia de Telecomunicações.

Área de concentração: Sistemas de Telecomunicações

Aprovado em: 20 / 10 / 2022

BANCA EXAMINADORA

Prof. Dr. Auzuir Ripardo de Alexandria (orientador)

Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Fortaleza

Prof. Dr. José Wally Mendonça Menezes

Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Fortaleza

Prof. Dr. Michael Santos Duarte

Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Fortaleza



Documento assinado eletronicamente por **Auzair Ripardo de Alexandria, Professor do Ensino Básico, Técnico e Tecnológico**, em 21/10/2022, às 09:48, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



Documento assinado eletronicamente por **Jose Wally Mendonca Menezes, Professor do Ensino Básico, Técnico e Tecnológico**, em 21/10/2022, às 10:44, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



Documento assinado eletronicamente por **Michael Santos Duarte, Professor do Ensino Básico, Técnico e Tecnológico**, em 21/10/2022, às 20:53, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



A autenticidade do documento pode ser conferida no site https://sei.ifce.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 informando o código verificador **4244578** e o código CRC **4D7BA57C**.

Dedico este trabalho primeiramente a minha família, especialmente a minha esposa Keyla Xavier de Matos Ribeiro que foi o principal sustentáculo para que eu conseguisse concluir este trabalho.

AGRADECIMENTOS

A Deus pela força, sabedoria e tenacidade durante o período do curso, e por todas as oportunidades de crescimento pessoal e profissional até aqui.

À família primeiramente pelo apoio em todos os momentos, sendo meu porto seguro em todos os dias que pensei em desistir, por sempre incentivar a continuar em frente em busca de mais esse sonho.

Ao IFCE por abrir as portas, garantindo a um filho de agricultor com uma dona de casa a oportunidade de conquistar o título de Mestre em Engenharia, conquista essa que foi a primeira dentro da família.

Ao Professor Auzuir Ripardo por me acompanhar e orientar este trabalho com dedicação e paciência como quem dedica a um filho, e mais ainda por não me deixar abandonar o curso mesmo em meio a uma pandemia.

“Os valentes lutam, os cautelosos se defendem e os sábios aconselham. De nenhum se perde o talento”

(Sun Tzu)

RESUMO

A partir de uma demanda inicialmente gerada pelos desafios propostos pela robótica móvel originou-se esta pesquisa, localizar ativos móveis em qualquer ambiente sempre é uma tarefa complexa, em ambientes fechados torna-se ainda mais complexa, visto que um dos principais meios utilizados, o GPS, tem sua eficácia e precisão reduzidos. Esta pesquisa objetiva comparar métodos de localização topológica por meio do auxílio de um *framework* de aprendizado de máquina automático, onde foram aplicados dois *datasets* e selecionados 5 modelos com bons parâmetros de eficiência. Aplicando uma abordagem que combina técnicas de aprendizado de máquina aliado a coleta de sinais *Wi-Fi* que podem ser aplicadas tanto em ambientes *indoor* como em ambientes *outdoor*. Resultados obtidos por alguns dos modelos estudados apresentaram índices 100% para Precisão, acurácia, F1 Score para classificação com o *dataset* local e desvio padrão igual a 0, ficando demonstrada a elevada assertividade dos modelos.

Palavras-chave: Robótica. Localização de ativos. *Wi-Fi/Bluetooth*. Aprendizado de máquina. *Beacons*.

ABSTRACT

From a demand initially generated by the challenges posed by mobile robotics this research originated, locating mobile assets in any environment is always a task complex, indoors it becomes even more complex, since one of the main means used, the GPS, has its effectiveness and precision reduced. This research aims to compare topological location methods through the aid of a learning framework automatic machine, where two datasets were applied and 5 models were selected with good efficiency parameters. Applying an approach that combines learning techniques from machine combined with the collection of Wi-Fi signals that can be applied both indoors and outdoors. in outdoor environments. Results obtained by some of the models studied showed 100% indices for Precision, Accuracy, F1 Score for classification with local dataset and deviation default equal to 0, demonstrating the high assertiveness of the models.

Keywords: Robotics. Asset Location. Wi-Fi/Bluetooth. Machine Learning. Beacons.

LISTA DE ILUSTRAÇÕES

Figura 1 – Resumo da classificação de Robôs de Ben-Ari	14
Figura 2 – Exemplo de Classificação com KNN.	17
Figura 3 – Exemplo de Classificação com SVM.	19
Figura 4 – Exemplo de Gráfico ROC com AUC	25
Figura 5 – Comparação Matriz de confusão KNN com dataset público	29
Figura 6 – Comparação Matriz de confusão KNN com dataset local	29
Figura 7 – Comparação Gráficos ROC/AUC para modelo KNN	30
Figura 8 – Comparação Clusters de classificação para modelo KNN	31
Figura 9 – Comparação Matriz de confusão LDA com dataset público	32
Figura 10 – Comparação Matriz de confusão LDA com dataset local	33
Figura 11 – Comparação Gráficos ROC/AUC para modelo LDA	33
Figura 12 – Comparação Clusters de classificação para modelo LDA	34
Figura 13 – Comparação Matriz de confusão ET com dataset público	36
Figura 14 – Comparação Matriz de confusão ET com dataset local	36
Figura 15 – Comparação Gráficos ROC/AUC para modelo ET	37
Figura 16 – Comparação Clusters de classificação para modelo ET	38
Figura 17 – Comparação matriz de confusão LightGBM com dataset público	39
Figura 18 – Comparação matriz de confusão LightGBM com dataset local	40
Figura 19 – Comparação Gráficos ROC/AUC para modelo LightGBM	40
Figura 20 – Comparação clusters de classificação para modelo LightGBM	41
Figura 21 – Comparação Matriz de confusão RF^2 com dataset público	43
Figura 22 – Comparação Matriz de confusão RF^2 com dataset local	44
Figura 23 – Comparação Gráficos ROC/AUC para modelo RF^2	44
Figura 24 – Comparação clusters de classificação para modelo RF^2	45

LISTA DE TABELAS

Tabela 1 – Tecnologias utilizadas para localização <i>indoor</i>	15
Tabela 2 – Resultados com uso de <i>dataset</i> público	27
Tabela 3 – Resultados com uso de <i>dataset</i> local	27
Tabela 4 – Resultados para <i>dataset</i> público com modelo KNN	28
Tabela 5 – Resultados para <i>dataset</i> local com modelo KNN	28
Tabela 6 – Resultados para <i>dataset</i> público com modelo LDA	31
Tabela 7 – Resultados para <i>dataset</i> local com modelo LDA	32
Tabela 8 – Resultados para <i>dataset</i> público com modelo ET	35
Tabela 9 – Resultados para <i>dataset</i> local com modelo ET	35
Tabela 10 – Resultados para <i>dataset</i> público com modelo LightGBM	38
Tabela 11 – Resultados para <i>dataset</i> local com modelo LightGBM	39
Tabela 12 – Resultados para <i>dataset</i> público com modelo RF^2	42
Tabela 13 – Resultados para <i>dataset</i> local com modelo RF^2	42

LISTA DE ABREVIATURAS E SIGLAS

ML	<i>Machine Learninig</i>
MLP	<i>Multi-Layer Perceptron</i>
KNN	<i>K Nearest Neighbor</i>
SVM	<i>Support Vector Machine</i>
S.O.	Sistema Operacional
BLE	<i>Bluetooth Low Energy</i>
SLAM	<i>Simultaneous Localization and Mapping</i>
GPS	<i>Global Position System</i>
FCC	<i>United States Federal Communication Commission</i>
ANATEL	Agência Nacional de Telecomunicações
GSM	<i>Global System Mobile</i>
WSN	<i>Wireless Sensor Networks</i>
LoS	<i>Line-of-Sight</i>
3D	3 Dimensões
GNSS	<i>Global Navigation Satellite System</i>
TP	<i>True Positive</i>
TN	<i>True Negative</i>
FP	<i>False Positive</i>
FN	<i>False Negative</i>
IFCE	Instituto Federal de Ciência, Tecnologia e Educação do Ceará
UECE	Universidade Estadual do Ceará
UFC	Universidade Federal do Ceará
AP	<i>Access Point</i>
SSID	<i>Service Set Identifier</i>

RSSI	<i>Received Signal Strength Indication</i>
RF	Rádiofrequência
ET	<i>Extra Tree Classifier</i>
LDA	<i>Linear Discriminant Analysis</i>
LightGBM	<i>Light Gradient Boost Machine</i>
RF^2	<i>Random Forest</i>
RAM	<i>Random Access Memory</i>
FPGA	<i>Field Programmable Gate Array</i>
MROL	<i>Mobile Robots Odor Localization</i>
ROC	<i>Receiver Operating Characteristic</i>
AUC	<i>Area under the ROC curve</i>

LISTA DE SÍMBOLOS

Σ	Letra grega Sigma
Θ	Letra grega Theta
μ	Letra grega Mi

SUMÁRIO

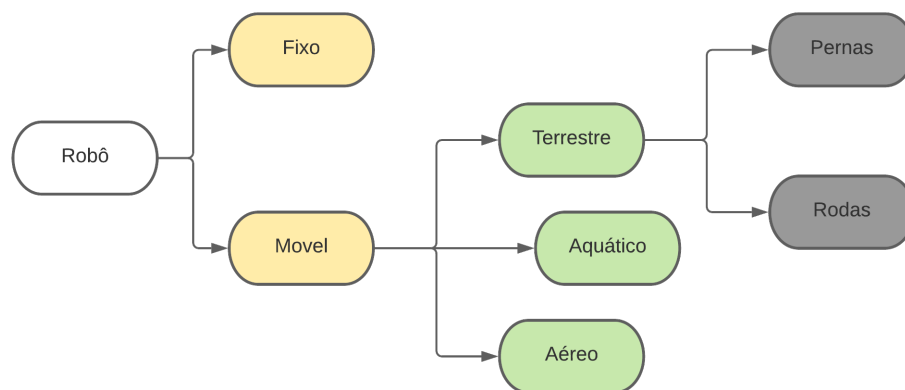
1	INTRODUÇÃO	14
1.1	Objetivos	16
1.1.1	Objetivo geral	16
1.1.2	Objetivos específicos	16
1.2	Organização da Dissertação	16
2	FUNDAMENTAÇÃO TEÓRICA	17
2.1	Aprendizado de máquina	17
2.1.1	KNN - K Nearest Neighbors	17
2.1.2	MLP - Multi Layer Perceptron	18
2.1.3	SVM - Support Vector Machine	18
2.1.4	LDA - <i>Linear Discriminant Analysis</i>	19
2.1.5	Árvores de Decisão	19
2.1.5.1	LightGBM - <i>Light Gradient Boost Machine</i>	20
2.1.5.2	RF^2 - <i>Random Forest</i>	20
2.1.5.3	ET - <i>Extra-Trees</i>	20
2.1.6	AutoML - Auto-machine Learning	20
2.2	Localização	21
3	METODOLOGIA	23
3.1	<i>Datasets</i>	23
3.1.1	Coleta Local	23
3.2	Relatório de Classificação - <i>Classification Report</i>	24
4	RESULTADOS E DISCUSSÃO	27
4.1	<i>K-Nearest Neighbors</i> (KNN)	28
4.2	<i>Linear Discriminant Analysis</i> (LDA)	31
4.3	<i>Extra-Tree</i> (ET)	34
4.4	<i>Light Gradient Boost Machine</i> (LightGBM)	37
4.5	<i>Random Forest</i> (RF^2)	42
4.6	Considerações Finais	45
5	CONCLUSÃO	47
	REFERÊNCIAS	48

1 INTRODUÇÃO

O termo robô foi registrado pela primeira vez na literatura em uma peça de teatro: Robôs Universais de Rossum de Čapek (1921) e traduzido no Brasil em Tchépek (2010) apresentando o conceito de autômatos. Posteriormente, os três princípios ou leis da robótica postuladas por Asimov (1940) e traduzidos para português em Asimov (2015) foram concebidos. Em 1961, George Devol apresentou o primeiro modelo de braço robótico industrial, bem diferente dos robôs propostos por Karel Capek e Isaac Asimov em suas obras literárias.

Segundo Ben-Ari e Mondada (2017, p. 02) apresentam a seguinte definição de robô: *“Uma máquina capaz de executar uma série de ações complexas automaticamente, especialmente programadas por computador”*. A partir dessa definição os autores extraem duas características importantes: Executa ações automaticamente e é programável por computador. Ainda segundo Ben-Ari a classificação de robôs quanto à sua mobilidade podendo serem fixos ou móveis, robôs móveis por sua vez se dividem quanto ao ambiente ao qual foram destinados; classificados desta forma em aquáticos, aéreos ou terrestres; em que os terrestres ainda são subdivididos pelo mecanismo de movimentação que podem ser rodas ou pernas.

Figura 1 – Resumo da classificação de Robôs de Ben-Ari



Fonte: Adaptado de Ben-Ari e Mondada (2017, p. 02).

Na ótica de robôs móveis, pode-se considerar o conceito apresentado na GM Motor Show em 1956, foi apresentado um projeto de carro autônomo. Na proposta inicial, os veículos poderiam “dirigir” de forma autônoma em rodovias, segundo Kim e Choi (2020). Nos dias atuais ocorre o controle de cruzeiro ou piloto automático. Já os veículos completamente autônomos continuam em pesquisas com resultados ainda razoáveis e com grande margem de aprimoramentos. Contudo, ainda assim, esses veículos são considerados robôs móveis.

Existem pesquisas utilizando sistemas *fuzzy* para localização com sistemas *wireless*. Marwan, Mourad e Mousa (2020) apresentam uma revisão do que já existe de pesquisa com este foco. Ainda segundo Marwan, Mourad e Mousa (2020), os sistemas de localização vieram a ganhar seu pico de pesquisas após 1996, devido à obsolescência dos sistemas de posicionamento por satélite com o advento do GPS, bem como a exigência pela Comissão Federal de Comunicação (FCC) Norte Americana, o equivalente a nossa ANATEL, ao obrigar as operadoras de redes

a fornecerem localização dos telefones celulares.

Técnicas de localização usadas a princípio em robôs móveis podem ser aplicadas em diversos setores, como por exemplo Bloch e Pastell (2020) que utilizando *Bluetooth* (BLE) combinando à visão computacional, mapearam a localização de vacas leiteiras em uma fazenda.

Uma considerável parte das pesquisas apresentadas até então nesse estudo é considerado um ambiente aberto (*outdoor*) com poucos obstáculos e uma boa linhas de visão. Para ambientes fechados (*indoor*) Giuliano et al. (2020) trazem uma proposta bem recente para localização por *bluetooth* em ambientes indoor. Nesse artigo é apresentado o *machine learning* (ML) como um fator de aprimoramento de localização bem como de redução dos custos computacionais e energéticos quando se trata do envio dos dados colhidos pelos sensores e do processamento desses dados.

No âmbito da robótica, ao se falar de localização, o mais comum é abordar a estratégia SLAM, não sendo esta a única abordagem e sendo a mesma repleta de outros protocolos e algoritmos, já que SLAM em tradução livre significa: Localização e Mapeamento Simultâneo. A partir dessa abordagem ou premissa, as formas de se atingir o objetivo são as mais variadas possíveis, desde o uso exclusivo de sensores ultrassônicos como Mannay et al. (2020) para localização e mapeamento 3D do ambiente, perpassando pela combinação de GNSS com outras técnicas como apresentadas por Kim e Choi (2020), Yao e Li (2020), Li et al. (2020). Com isso é possível aumentar a precisão da localização ou do mapeamento ao combinar técnicas ou sensores que isolados não apresentariam tais resultados significativos.

Outro método para se obter a posição de um robô em relação a outro é utilizar o *Received Signal Strength Indicator* (RSSI), que em outras palavras é estimar a distância entre dois dispositivos pela intensidade de sinal wireless que eles compartilham, seja ele RSSI de *Wi-Fi*, de *Bluetooth* ou qualquer outra forma de comunicação sem fios que utilize antenas, como apresentam Oliveira et al. (2014) é possível também com esta informação aplicar algoritmos de localização ponderada como apresentam (YAO et al., 2015).

Como apresentam Ashraf, Hur e Park (2018), as tecnologias utilizadas para localização em ambientes *indoor* variam em custos de implementação, em dependência de infraestrutura prévia e também relacionado a precisão da localização obtida, ressaltando que os autores não consideraram custos computacionais, conforme Tabela 1.

Tabela 1 – Tecnologias utilizadas para localização *indoor*

Tecnologia	Infraestrutura	Exatidão	Custo
Infravermelho	Dependente	0,5 a 3 m	Moderado
Sinais de Som	Dependente	Inferior a 1 m	Moderado
Rádiofrequência ID (RFID)	Dependente	1 a 3 m	Moderado
<i>Ultra Wide Band</i> (UWB)	Dependente	Inferior a 1m	Elevado
<i>Pedestrian dead Reckoning</i> (PDR)	Independente	2 a 5m	Isento
WiFi	Dependente	2 a 5 m	Baixo
<i>iBeacon</i>	Dependente	1 a 5 m	Baixo
Visão (Smartphones)	Independente	até 5 m	Isento
Luz Visível	Semi-dependente	até 8 m	Isento
Sistemas Magnéticos	Independente	1 a 10 m	Isento

Fonte: Adaptado de Ashraf, Hur e Park (2018)

Para as tecnologias apresentadas na Tabela 1, uma das formas de utilizar métodos baseados em sinais de rádio frequência (RF) para localização *indoor* é através da potência de sinal recebida que pode ser obtida por (EZPELETA et al., 2015)

$$P = A * dist^{-n}, \quad (1.1)$$

em que P é a potência de sinal recebido, A a potência de sinal emitido, $dist$ a distância entre o emissor e o receptor e por fim n é o coeficiente de atenuação médio.

1.1 Objetivos

1.1.1 Objetivo geral

Identificar modelos de aprendizagem de máquina com melhores parâmetros de eficiência para localização de ativos utilizando RSSI de redes *Wi-fi*

1.1.2 Objetivos específicos

- Identificar modelo com melhores parâmetros de eficiência.
- Comparar rendimento de algoritmos elaborados, usando um *dataset* público de intensidade de sinal *Wi-Fi*.
- Avaliar algoritmos de aprendizado de máquina para localização de robôs utilizando *dataset* local.

1.2 Organização da Dissertação

Capítulo 2: Revisão bibliográfica sobre o tema abordado neste trabalho, desde artigos científicos, patentes e aplicações.

Capítulo 3: Análise básica dos *datasets* e explicação mais detalhada dos algoritmos utilizados.

Capítulo 4: Apresentação e discussão dos resultados obtidos por meio de treinamento dos algoritmos de aprendizado de máquina.

Capítulo 5: Conclusão e considerações finais.

2 FUNDAMENTAÇÃO TEÓRICA

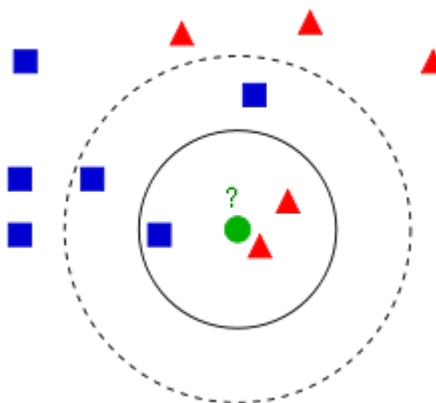
2.1 Aprendizado de máquina

As pesquisas desta área de conhecimento se iniciaram com a publicação de McCulloch e Pitts (1943), apresentando assim um novo modelo neurofisiológico de como funcionam o pensamento e aprendizado e por meio deste artigo formularam uma teoria de aprendizado linear com um limiar predefinido que poderia ou não ser atingido, por meio de uma função do somatório de todas as entradas utilizadas, gerando assim um modelo de classificador binário, posteriormente conhecido como neurônio artificial ou perceptron. Recebendo um aprimoramento da teoria por Rosenblatt (1958), diversos outros modelos com estruturas similares foram desenvolvidos para os mais diferentes problemas de classificação, regressão ou agrupamento, de treinamento supervisionado ou não-supervisionado no que hoje conhece-se por aprendizado de máquina (*machine learning*).

2.1.1 KNN - K Nearest Neighbors

A mais simples em termos matemáticos e custos computacionais das abordagens para classificação foi proposta por Cover e Hart (1967), utilizando-se do cálculo da distância que uma nova amostra apresenta para um grupo de amostras previamente classificadas. Por meio deste resultado, o algoritmo atribui a esta nova amostra uma classe. Tendo este ainda recebido uma nova abordagem proposta por Dudani (1976), considerando também os pesos dessas distâncias para então ser definida a classe à qual a nova amostra irá pertencer, onde esses fatores vão influenciar na determinação da classe para a nova amostra apresentada.

Figura 2 – Exemplo de Classificação com KNN.



Fonte: Imagem pública - Autor desconhecido

O modelo de aprendizagem KNN segue o seguinte procedimento de aprendizagem após a separação do *dataset* em grupo de treino e grupo de teste, ambos com as classes devidamente indicadas em cada amostra de n atributos segundo Cover e Hart (1967):

1. Cria-se uma base de informações com as amostras do grupo de treinamento.
2. Ao apresentar uma nova amostra ao modelo ele calcula a distância (geralmente euclidiana) para cada amostra já conhecida.

3. Calculada a distância lista-se os k vizinhos com maior frequência.
4. A nova amostra é alocada na classe com maior frequência entre os k vizinhos listados no passo anterior.

O valor do parâmetro número de vizinhos k pode ser atribuído arbitrariamente. Embora Lantz (2013) apresente como k ideal que este seja um valor ímpar, isso não é regra, dado que este valor pode ser otimizado por meio de teste automatizado, garantindo assim uma baixa ocorrência de erros às atribuições bem como prevenção do excesso de custo computacional. Observa-se que, conforme se aumenta o número de vizinhos, também se aumenta o tempo de processamento. Para distância euclidiana e duas instâncias u_i e v_i utiliza-se

$$dist(u, v) = \sqrt{\sum_{i=1}^n (u_i - v_i)^2}, \quad (2.1)$$

Pode-se também, para amostras compostas por um vetor de atributos, utilizar o cálculo da distância Manhattan, conforme

$$dist(u, v) = \sum_{i=1}^n |u_i - v_i|, \quad (2.2)$$

em que u e v são vetores n -dimensionais; e o índice i representa a i -ésima amostra nesses vetores, para ambos os cálculos de distancia.

2.1.2 MLP - Multi Layer Perceptron

O estudo que havia iniciado por McCulloch e Pitts e recebeu uma atualização por Rosenblatt, ainda permanecia um classificador binário e linear, décadas depois. Com o aprimoramento do próprio computador esses estudos viram novamente a luz. Werbos (1994) apresentou um dos principais algoritmos utilizados com redes baseadas no perceptron de Rosenblatt (1958), permitindo agora organizá-los em redes com camadas e assim tornando possível a classificação multiclasse.

A regra geral de aprendizagem do *Multi Layer Perceptron* (MLP) é definida por

$$f(u) = \sum_{i=1}^n W_i * X_i - \Theta, \quad (2.3)$$

em que W são pesos e X são atributos, tornando assim u um somatório ponderado dos vetores de atributos dos quais se busca reconhecer um padrão e Θ é o limiar de ativação dessa função, permitindo assim o disparo (nível lógico 1), caso todo o somatório seja maior que o limiar ou não (nível lógico 0) do neurônio, caso o somatório seja igual ou menor que o limiar.

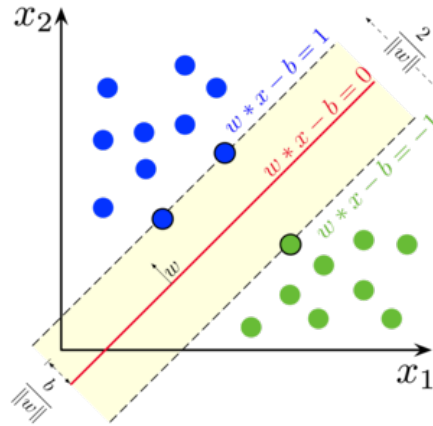
Segundo Werbos (1994), o treinamento de um modelo MLP, na maioria dos casos, utiliza-se do método de retropropagação do erro para correção dos pesos W que é efetuado a cada atributo individualmente na entrada de cada neurônio, e com isso atualiza os pesos de toda a rede com base no erro de saída da mesma.

2.1.3 SVM - Support Vector Machine

Originalmente o *Support Vector Machine* (SVM) foi idealizado por Cortes e Vapnik (1995) para criar um hiperplano separando duas classes distintas, tornando-o essencialmente um

classificador binário. Para superar esta limitação Weston, Watkins et al. (1999) apresentaram uma nova abordagem neste algoritmo que permite classificação multiclasse sem a necessidade de sobreposição de diversas SVMs, como era realizado até então.

Figura 3 – Exemplo de Classificação com SVM.



Fonte: Imagem pública - Autor desconhecido

2.1.4 LDA - Linear Discriminant Analysis

O modelo, assim como o SVM, foi elaborado como um classificador binário e possui sua origem na Função Discriminante de Fischer apresentada em Fisher (1936) onde os termos de ambas as funções possuem intercambialidade.

Este modelo possui extrapolação para classificação multiclasse e entre os diversos usos desse modelo de aprendizagem a localização pode ser vista em Luo et al. (2019), Wang et al. (2021), onde sua regra geral de classificação toma por base a derivada da função discriminante de Fischer, que foi proposta por Rao (1948) e possui a seguinte forma:

$$\sum_b = \frac{1}{C} \sum_{i=1}^C (\mu_i - \mu)(\mu_i - \mu)^T \quad (2.4)$$

A Análise Linear Discriminante (LDA) é mais comumente usada como técnica de redução de dimensionalidade na etapa de pré-processamento para aplicações de classificação de padrões e de aprendizado de máquina. O objetivo é projetar um conjunto de dados em um espaço de menor dimensão com boa separabilidade de classe, a fim de evitar *overfitting* e reduzir os custos computacionais.

2.1.5 Árvores de Decisão

De forma similar à origem biológica das redes neurais artificiais com o perceptron de McCulloch e Pitts (1943), uma classe de algoritmos de decisão foi proposto por, inicialmente, Luckman (1972) como um algoritmo para auxiliar decisões. Este modelo de aprendizagem utiliza uma estrutura de árvore onde há o nó-raiz e local de entrada para o emprego de *dataset* e tem nos nós-folhas, a saída da classificação e nos nós-intermediários ocorrem as subdivisões nos conjuntos de dados para se atingir as classes representadas nos nós-folhas.

Uma aplicação de estrutura similar às árvores de decisão que são as árvores de comportamento (*Behavior Trees* - BT) é na programação de personagens não-jogáveis (NPCs) em jogos digitais disponível nas principais *engines* de desenvolvimentos de jogos *Unreal* com a biblioteca de BT presente de forma nativa e também na *Unity* onde embora não se apresente de forma nativa é possível a instalação da mesma como pacote adicional.

Além dessas e outras aplicação do modelo de árvore de decisão, existem diversos outros modelos derivados e baseados, entre eles podem-se citar os modelos *Light Gradient Boost Machine*, *Extra-trees* e *Random Forest* que no presente trabalho são adotados para classificação supervisionada.

2.1.5.1 *LightGBM - Light Gradient Boost Machine*

Um modelo baseado nos algoritmos de árvore de decisão usado prioritariamente para ordenação e classificação, faz parte do grupo de algoritmos conhecidos como *Gradient Boost Machine*, desenvolvido originalmente pela Microsoft como parte integrante do *framework* (BATCHU; SEETHA, 2022).

O detalhamento da elaboração deste modelo é elaborado em Ke et al. (2017), sendo esta a primeira publicação sobre o modelo. Por outro lado, Kopitar et al. (2020) realizam aplicação deste modelo para detecção de diabetes melitus tipo 2 em estágios iniciais.

A estrutura do algoritmo que é baseada em árvores de decisão foi aprimorada, tornando a mesma mais leve, mais rápida e mais precisa em relação ao algoritmo que tomou por base.

2.1.5.2 *RF² - Random Forest*

Random Forest é um modelo que aumenta a aleatoriedade para o algoritmo de árvores de decisão ao randomizar a seleção do atributo de análise em cada árvore. Foi proposto inicialmente por Ho (1995), e com isso melhorou uma das fragilidades do algoritmo de árvore de decisão que seria a generalização do modelo, visto que no algoritmo base todas as seleções de amostra, de espaço e de *splits* a cada nó são sempre realizados visando a maior ganho de informações para adequação do modelo para o *dataset* apresentado.

2.1.5.3 *ET - Extra-Trees*

Tendo o nome com origem no encurtamento do *Extremely Randomized Trees - Extra Trees* (ET), este algoritmo cria múltiplas árvores de decisão de forma completamente aleatórias em todos os aspectos. A principal diferença em relação ao algoritmo *Random Forest* é justamente a randomização em que o ET se utiliza de mais processos aleatórios que o *RF²*, randomizando até mesmo o *split* de dados aplicado a cada decisão em cada nó.

A utilização deste algoritmo para localização pode ser vista em Khan et al. (2022), onde o autor compara resultados com dados de RF aplicados a 3 modelos que são ET, MPL e *RF²* com objetivo de identificar qual dos modelos propostos traz melhor resultado para localização *indoor*.

2.1.6 *AutoML - Auto-machine Learning*

Com a evolução de algoritmos e *hardware* já é possível também permitir que a máquina escolha o algoritmo mais eficiente permitindo, assim, melhores resultados com menor tempo e usos de recursos computacionais. Estes *frameworks* são conhecidos como AutoML. Dentro dessa categoria podem-se encontrar vários exemplos como *TPOT* Le, Fu e Moore (2020), *MLbox*

Gagana (2021), Auto-sklearn Feurer et al. (2020), Auto-Keras Jin, Song e Hu (2019), Google Cloud AutoML Marcu e Danubianu (2021), e o que é utilizado neste trabalho, o Pycaret (ALI, 2020) provido dentro do serviço em nuvem Google Colab.

Com esses *frameworks*, ocorre situação diferente dos modelos anteriores em que os dados são preparados, normalizados, separados em grupos de treinamento e teste, para então passarem por um treinamento dividido em vários blocos com uma subdivisão no grupo de treinamento com uma parte usada para validação do modelo. Além dessa etapa de treinamento completamente automatizada ainda é possível otimizar os hiper-parâmetros do modelo escolhido, conforme apresentam Nagarajah e Poravi (2019).

2.2 Localização

Os modelos mais antigos para determinação de localização de qualquer plataforma móvel, seja um robô, ou um veículo autônomo é baseado em *encoder* e odometria conforme Chen, Huang e Zhou (2014). Contudo esse método possui uma falha que acumulada ao longo de um trajeto muito longo pode levar a erros consideráveis, como apresentam Tai et al. (2014) causados prioritariamente pelo escorregamento das rodas do robô/veículo e propõem a combinação desses métodos com pontos de referência que poderia ser coletados por sonar e laser como apresentam Dellaert et al. (1999) e câmeras como demonstram Kiriy e Buehler (2002).

Uma abordagem com custo de implementação menor pode ser realizada utilizando *field programmable gate array* (FPGA) como apresentam Amor et al. (2019) com uma técnica que compila informações de localização absoluta e relativa. Nesse artigo, os autores tomam por base como localização absoluta uma região previamente mapeada com 4 marcadores fixos nos limites dessa região, enquanto que o modelo de localização relativa utiliza por base modelo de odometria independente para duas rodas motoras, conforme o modelo de robô autônomo da *K-Team* o modelo *Khepera II*.

A localização pode ser realizada também com dados de RSSI de Bluetooth como abordam Giuliano et al. (2020), Khan e Nguang (2021), Yao et al. (2015), que, embora a aplicação seja de visitantes em museu, a mesma técnica pode ser aplicada para qualquer outra necessidade de localização *indoor*, e para o propósito de localização de robôs é especialmente eficiente quanto combinada com odometria, ou qualquer outro tipo de sensoramento de ambiente. Dependendo da aplicação desta plataforma pode inclusive recorrer ao uso de aprendizado de máquina e redes neurais artificiais.

Em casos mais específicos como é para robôs móveis localizadores de odor (MROL) embora já existam abordagens com diversos tipos de sensores, até mesmo biossensores Shi e Sun (2012), já apresentava uma abordagem implementando Algoritmos Genéticos (GA) tentando resolver o problema de localização do robô e, também, da fonte do odor que originalmente é a missão destas plataformas.

Para localização utilizando apenas o RSSI de redes *WiFi*, Khan e Nguang (2021) discorrem sobre o algoritmo nomeado de *RangeLookup-MDS* que propõe a realizar por meio de 4 etapas a localização de qualquer plataforma, combinando técnicas que dependem da distância e técnicas que não dependem da distância entre o robô móvel e os beacons geradores de sinal, levando também em consideração apenas 1 nível de hierarquia no sinal RSSI para os vizinhos próximos.

Para concluir este capítulo pode-se perceber que várias são as técnicas e algoritmos utilizados para se determinar a localização de robôs moveis e, mesmo estes algoritmos e técnicas não ficam restritos a este tipo de aplicação. Sendo o caminho oposto também possível com as

devidas adequações e ajustes, e para todas elas podemos aplicar várias formas de medição dos parâmetros de rendimento e eficiência.

3 METODOLOGIA

3.1 *Datasets*

Dataset ou conjunto de dados é o conjunto de informações utilizado para treinar modelos matemáticos que neste trabalho são apresentados como modelos de aprendizado de máquina. Especificamente neste trabalho os modelos se utilizam de informações exclusivamente numéricas, sejam elas informações numéricas inteiras ou informações numéricas de ponto flutuante.

São aplicados aos modelos de ML dois *datasets*, sendo um deles é um modelo de disponibilidade pública. Neste primeiro *dataset* são encontradas 2000 amostras de intensidade de sinal *wifi* coletadas simultaneamente de 7 emissores distintos (UCI Machine Learning¹) (ROHRA et al., 2017; BHATT, 2005).

3.1.1 *Coleta Local*

A coleta de dados para criação do *dataset* local foi realizada com a montagem de uma estrutura de sinais *wifi* composta por 5 geradores de sinal, com posição conhecida e 10 de localização desconhecida e 1 sensor, para os elementos conhecidos dessa estrutura foram utilizados 6 placas de desenvolvimento **ESP WROOM 32** sendo 5 delas configuradas no modo AP (ponto de acesso) gerando cada um deles um ponto de acesso de rede sem fio, e a última placa permaneceu conectada ao computador em um *loop* de escaneamento de sinal *wifi* repetido a cada 500 ms.

Entre as respostas de escaneamento que eram obtidas pela saída serial da placa de desenvolvimento e imediatamente impressas no terminal da IDE de desenvolvimento, e posteriormente copiadas para um arquivo de texto que é organizado como o *dataset* local. Ainda sobre a coleta de dados, esta foi realizada em um ambiente *indoor* com distância entre leituras de 1 m cada e com possibilidade de sobreposição de leituras visto que a movimentação de coleta não foi mapeada previamente.

Cada zona de coleta sendo identificada por meio de um rótulo no *dataset*, também é possível para algoritmos de aprendizagem de máquina identificar áreas diferentes e associá-las em seus devidos grupos. Esta separação deve ser feita com pontos de coleta o mais próximo possível para que o tempo de coleta permita uma navegação e localização guiada pela intensidade dos sinais *wifi*.

O recurso utilizado para a elaboração e treinamento dos algoritmos, gravação das placas de desenvolvimento, coleta dos dados de intensidade RSSI foi um *notebook* com as especificações de *hardware* e *software* abaixo:

- Processador Ryzen 5 3500U;
- Memória 12 GB DDR4 2400 MHz (10 GB utilizável);
- HDD com 1 TB;
- Gráficos AMD Radeon RX Vega 8;
- S.O. Windows 10 home Edition.

¹ <https://archive.ics.uci.edu/ml/datasets/Wireless+Indoor+Localization>

A saída serial apresenta seu resultado em forma de texto com marcadores de início e término de cada ciclo de escaneamento, dados dos sinais identificados sejam eles de interesse para a presente pesquisa ou não, todos esses sinais são inicialmente identificados com o SSID da rede e o RSSI dessa mesma rede na seguinte forma: "SSID (RSSI)", com o RSSI expresso por meio de um número que pode ser entre 0 e -100 dB. Entretanto, na prática, esse número nunca é superior a -30 dB e inferior a -98 dB.

Para fins de comunicação por sinal sem fio, o ideal é que o RSSI permaneça entre -30 dB e -79 dB, visto que valores de ganho superiores a -30 db são praticamente inalcançáveis com antenas comuns de Rede sem fio, e ganhos inferiores a -79 db tendem a interromper a comunicação tornando inviável qualquer utilização prática.

Complementando este *hardware* para execução dos modelos acima citados localmente, foi também utilizado o serviço Jupyter em nuvem provido pelo *Google Computer Engine* que é o *Google Colab*, em que, com apenas alguns comandos, é possível executar nesse ambiente de desenvolvimento *python* o *framework* de *auto-machine learning* *pycaret*, permitindo assim que o próprio algoritmo e *framework* indiquem qual o algoritmo de aprendizado de máquina mais eficiente para o seu modelo de dados.

Para efeitos de comparação, ambos os conjuntos de dados foram aplicados como entrada no *framework* de *auto-machine learning*, e os resultados de ambos os modelos serão tratados na próxima seção, na apresentação dos dados foram selecionados 3 modelos com melhores índices de classificação para cada *dataset* e realizada uma complementação com os modelos que não foram vistos nos resultados de ambos os *datasets*, alcançando assim um total de 5 modelos com resultados a serem apresentados e tratados.

3.2 Relatório de Classificação - *Classification Report*

Precisão é a capacidade do classificador em não atribuir um rótulo a uma instância positiva que na verdade é negativo, para cada classe é definido um percentual de **verdadeiros positivos - TP** por

$$Preciso = \frac{TP}{TP + FP}, \quad (3.1)$$

em que TP corresponde a *True Positives*, FP a *False Positives* e Precisão é a acurácia de predições positivas.

Por outro lado, *Recall* é a capacidade do classificador de encontrar todas as instâncias positivas. Para cada classe é definido percentualmente por

$$Recall = \frac{TP}{TP + FN}, \quad (3.2)$$

em que FN corresponde a *False Negatives* e Recall à quantidade de Positivos corretamente identificados.

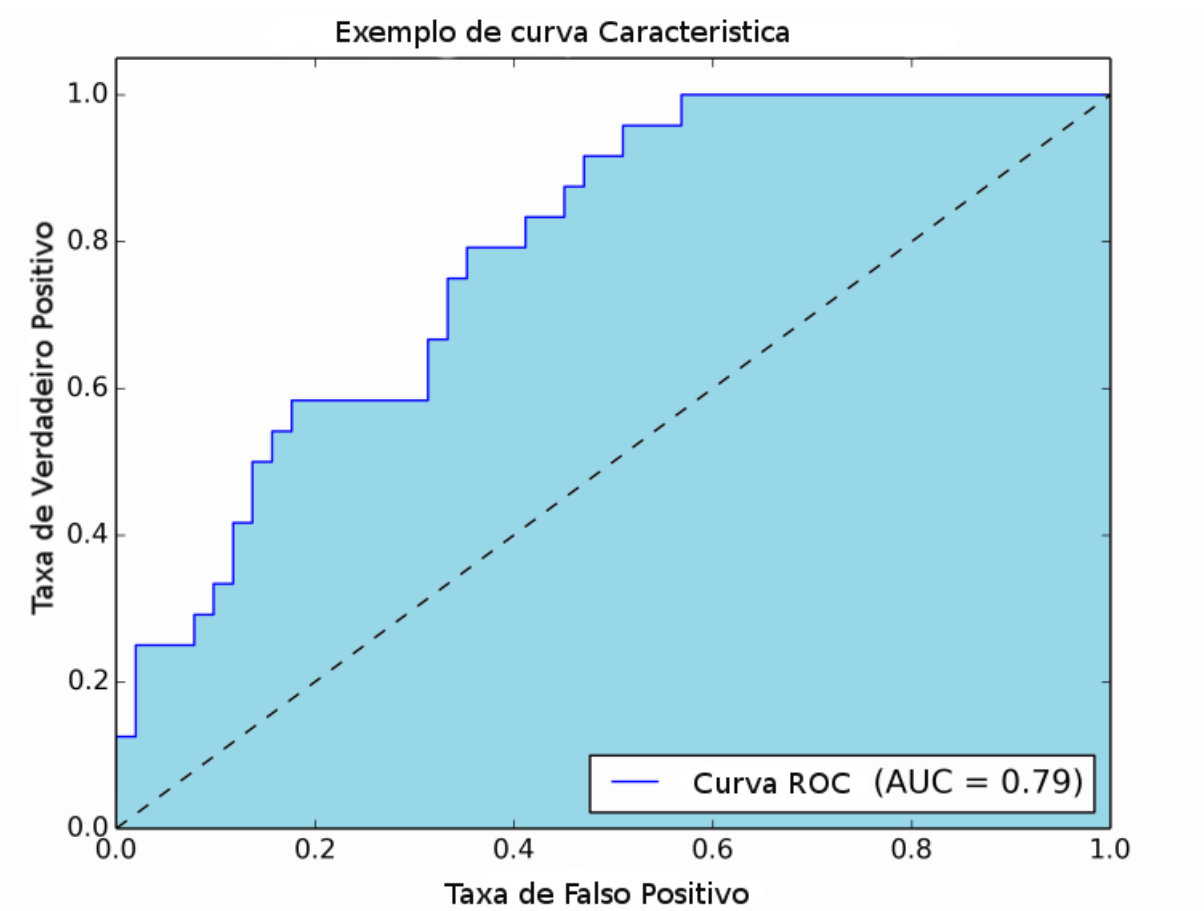
Abordando o conceito de *F1 score*, este busca responder a questão: “Qual o percentual de predições positivas que estão corretas?”.

O *F1 Score* é a média harmônica ponderada da precisão e do *recall* onde a melhor pontuação é 1 e a pior é 0. Geralmente, o F1 é menor que a precisão e o *recall* na mesma execução. Como regra geral, a média ponderada de F1 deve ser usada para comparar os modelos do classificador, não a precisão global. F1 é definido por

$$F1 = 2 * \frac{(Recall * Precision)}{Recall + Precision}. \quad (3.3)$$

Devemos considerar também dois parâmetros que se baseiam também nos critérios de VP, FP, VN e FN, que são a curva *Receiver Operating Characteristic* (ROC), que é representada de forma gráfica traçando uma curva *TP Rate* vs *FP rate* permitindo a fácil visualização da eficiência e da qualidade de classificação para o modelo ao qual essa curva foi traçada. A área abaixo dessa curva, *Area under the ROC Curve* (AUC), é a derivada da função que resulta no gráfico ROC, um valor adimensional que varia entre 0 e 1 resume em apenas um valor a informação contida no gráfico ROC.

Figura 4 – Exemplo de Gráfico ROC com AUC



Fonte: Adaptada de tópico sobre ROC e AUC no fórum *Stackexchange*.

Outra métrica relevante na análise de rendimento e eficiência de um algoritmo é o tempo de execução que no caso de algoritmos de aprendizagem de máquina o tempo de execução para associação dos padrões apresentados ao grupo que faz parte aquele padrão, aquela amostra, isto tomando por base modelos de aprendizagem supervisionada. A tendência é que quanto menor o tempo, melhor é o algoritmo, pois mais rápido executa a mesma tarefa quando comparado com outros algoritmos realizando a mesma tarefa ou utilizando o mesmo *dataset*.

Estas métricas apresentadas acima não são as únicas formas de se medir rendimento e eficiência em algoritmos de aprendizagem de máquina. Entretanto, para os modelos utilizados

na presente pesquisa, são os mais aplicados visto que são de fácil extração e entendimento dos dados.

4 RESULTADOS E DISCUSSÃO

Os resultados apresentados neste capítulo são obtidos com a aplicação do *framework pycaret*, e os dois *datasets* previamente apresentados neste trabalho, com todos os valores de tabelas e gráficos extraídos diretamente do *framework*. Os resultados serão apresentados e discutidos abordando cada modelo de aprendizagem com valores individuais de classificação e gráficos complementares dos resultados.

Com a utilização do *pycaret* e a aplicação dos dois *datasets* nesse *framework* de auto ML, o esperado seria pelo menos entre os algoritmos mais precisos se repetirem em ambos os *datasets* visto que possuem estrutura geral altamente similar. Entretanto o número de amostras e o número atributos é que difere.

As Tabelas 2 e 3 apresentam resultados de precisão, acurácia e tempo de execução do *pycaret* no treinamento do modelo obtido durante a execução da comparação de modelos realizada pelo próprio *framework* de auto ML.

Tabela 2 – Resultados com uso de *dataset* público

Modelo	Precisão	AUC	Acurácia	F1 Score	Tempo
KNN	98,42%	0,9975	98,36%	98,36%	0,123 s
ET	98,41%	0,9996	98,36%	98,36%	0,468 s
LightGBM	98,36%	0,9994	98,28%	98,28%	0,219 s
RF^2	98,20%	0,9993	98,14%	98,14%	0,524 s
LDA	97,10%	0,9981	96,78%	96,81%	0,019 s

Fonte: Autor (2022).

Tabela 3 – Resultados com uso de *dataset* local

Modelo	Precisão	AUC	Acurácia	F1 Score	Tempo
LightGBM	100%	1,0000	100%	100%	0,036 s
ET	100%	1,0000	100%	100%	0,426 s
RF^2	100%	1,0000	100%	100%	0,463 s
KNN	88,02%	0,9583	86,73%	86,10%	0,118 s
LDA	87,55%	0,9697	85,13%	84,77%	0,015 s

Fonte: Autor (2022).

Com base nos resultados preliminares das Tabelas 2 e 3, podem-se aprofundar os resultados para cada modelo em ambos os *datasets*, utilizando como método de análise os resultados de cada modelo após a criação do mesmo pelo *framework PyCaret*. Isso ocorre após a utilização da função (*tune_model*), que otimiza o modelo ao qual ela é vinculada, presente no *framework*, comparando assim os resultados por meio dos parâmetros previamente apresentados e incluindo a essa comparação a Matriz de Confusão para cada modelo e também uma clusterização da saída desses modelos antes e após a otimização.

Os resultados apresentados a seguir foram extraídos dos modelos executados por meio do *framework pycaret* que em suas configurações padrões aplicando-se a técnica de validação

KFold Validation, em que o K tem valor 10 e aliado a esse método de validação executa cada modelo 10 vezes, resultando em um total de 100 iterações para se obter os valores de rendimento de cada iteração.

4.1 *K-Nearest Neighbors* (KNN)

Detalhando-se melhor os resultados de cada modelo, obtém-se a Tabela 4 com alguns dos parâmetros de eficiência que foram apresentados na Seção 3.2, Relatório de Classificação do Capítulo 3, para os *datasets* público e local, nesta ordem.

Tabela 4 – Resultados para *dataset* público com modelo KNN

		Precisão	AUC	Acurácia	F1 Score
KNN padrão	Média	98,42%	0,9975	98,36%	98,36%
	Desvio	0,0124	0,0024	0,0128	0,0128
KNN Otimizada	Média	98,34%	0,9871	98,28%	98,28%
	Desvio	0,0122	0,0030	0,0125	0,0126

Fonte: Autor (2022).

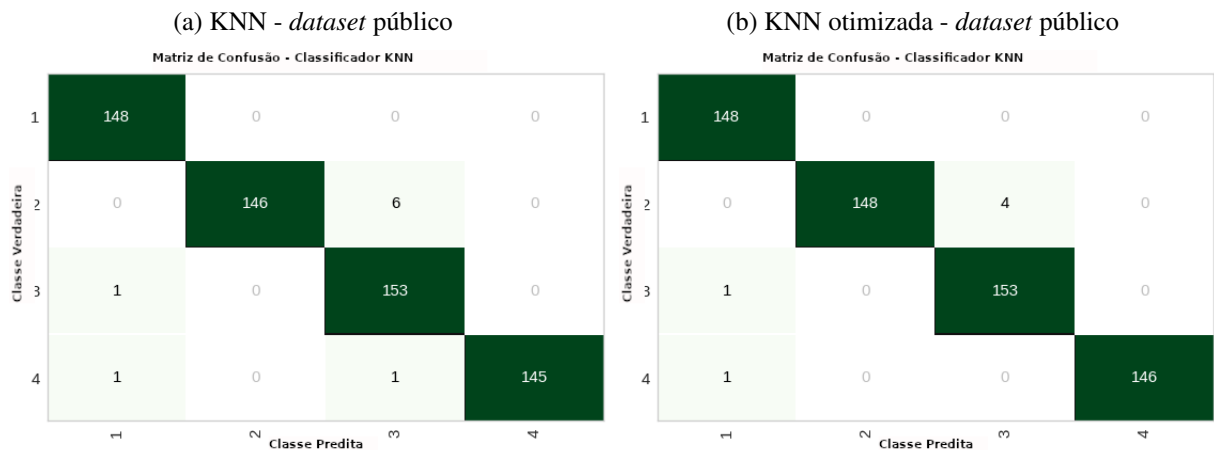
Tabela 5 – Resultados para *dataset* local com modelo KNN

		Precisão	AUC	Acurácia	F1 Score
KNN padrão	Média	88,02%	0,9583	86,73%	86,10%
	Desvio	0,0941	0,0408	0,0976	0,1026
KNN Otimizada	Média	92,76%	0,9874	91,35%	91,05%
	Desvio	0,0729	0,0210	0,0839	0,0877

Fonte: Autor (2022).

Diferente do que se observa na Tabela 5, é perceptível que a otimização gera um resultado mais adequado e isso se torna mais observável quando são analisados todos os parâmetros e percebe-se que o menor incremento dos parâmetros com métrica percentual foi no *F1 Score* com um incremento de 4,95 pontos percentuais acompanhada de redução também do desvio padrão. Essas reduções proporcionam, que na prática, que o modelo acerte mais as predições e diminua também a variação de resultados em torno dos resultados corretos.

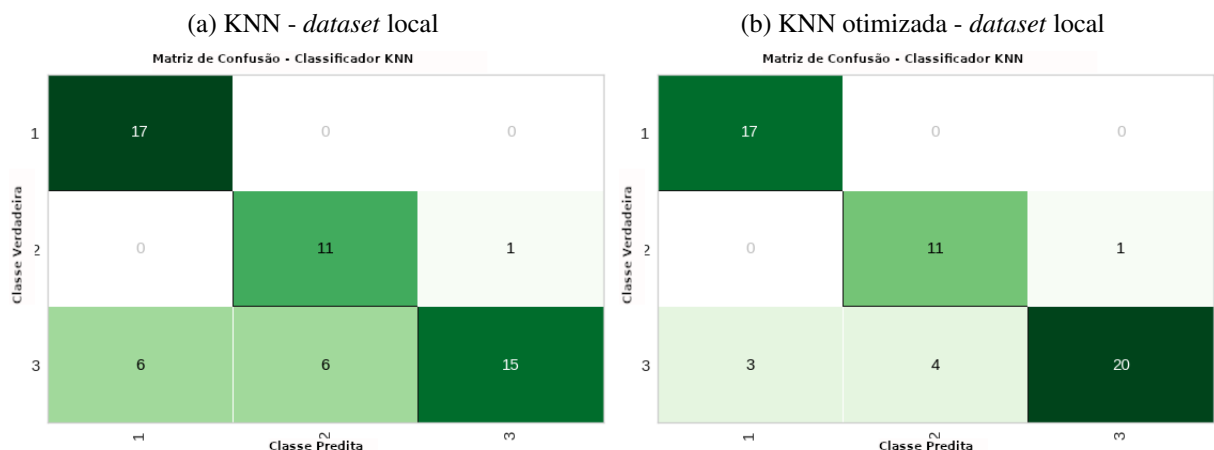
Figura 5 – Comparação Matriz de confusão KNN com dataset público



Fonte: Autor (2022).

Dada uma iteração de qualquer modelo de aprendizagem supervisionada o modelo registra como foi classificada cada amostra. Portanto, é possível fazer um apanhado de como as amostras foram classificadas dentro daquela iteração. Essa estrutura é conhecida como matriz de confusão. Deve-se ressaltar que a matriz de confusão por si não representa um bom critério de avaliação de rendimento por se tratar de um resultado de uma iteração isolada. A Figura 5 apresenta justamente uma comparação da última iteração do modelo KNN com sua versão otimizada, neste caso para o *dataset* público.

Figura 6 – Comparação Matriz de confusão KNN com dataset local

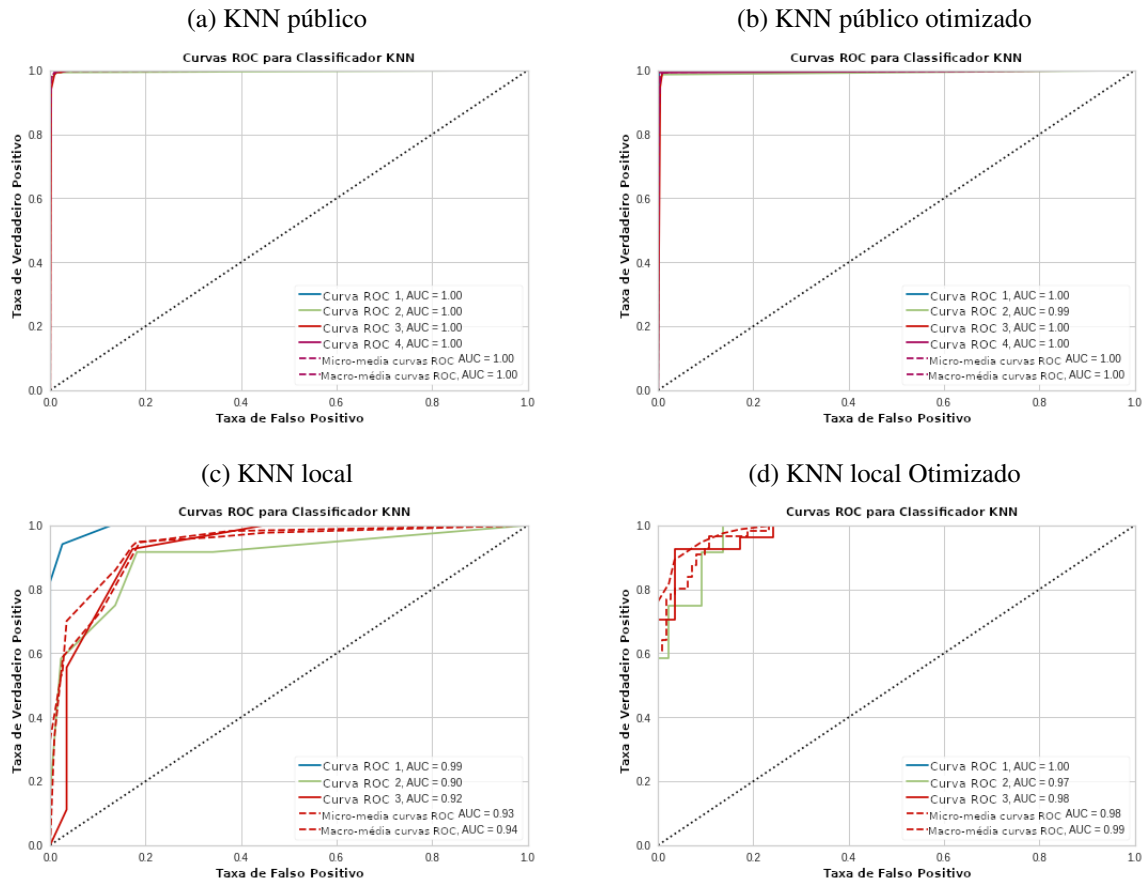


Fonte: Autor (2022).

Nota-se na Figura 5 que mudança de apenas 3 amostras foram classificadas de forma diferente num espaço amostral de 600 amostras de teste. Observa-se que essa variação de classificação encontra-se dentro do aceitável até mesmo se fossem diferentes iterações do algoritmo sem otimização. Contudo na Figura 6 nota-se uma melhoria até mesmo nesse parâmetro tão intrínseco a cada iteração, pois o espaço amostral é significativamente menor que o apresentado na Figura 5 e mesmo assim a quantidade de amostras que estava classificada de forma errada foi corrigida com a maior otimização.

Na primeira linha da Figura 7 podem-se observar as curvas ROC e valores AUC do modelo de aprendizado de máquina KNN por meio dos Gráficos 7.(a) e 7.(b) com o detalhamento

Figura 7 – Comparação Gráficos ROC/AUC para modelo KNN



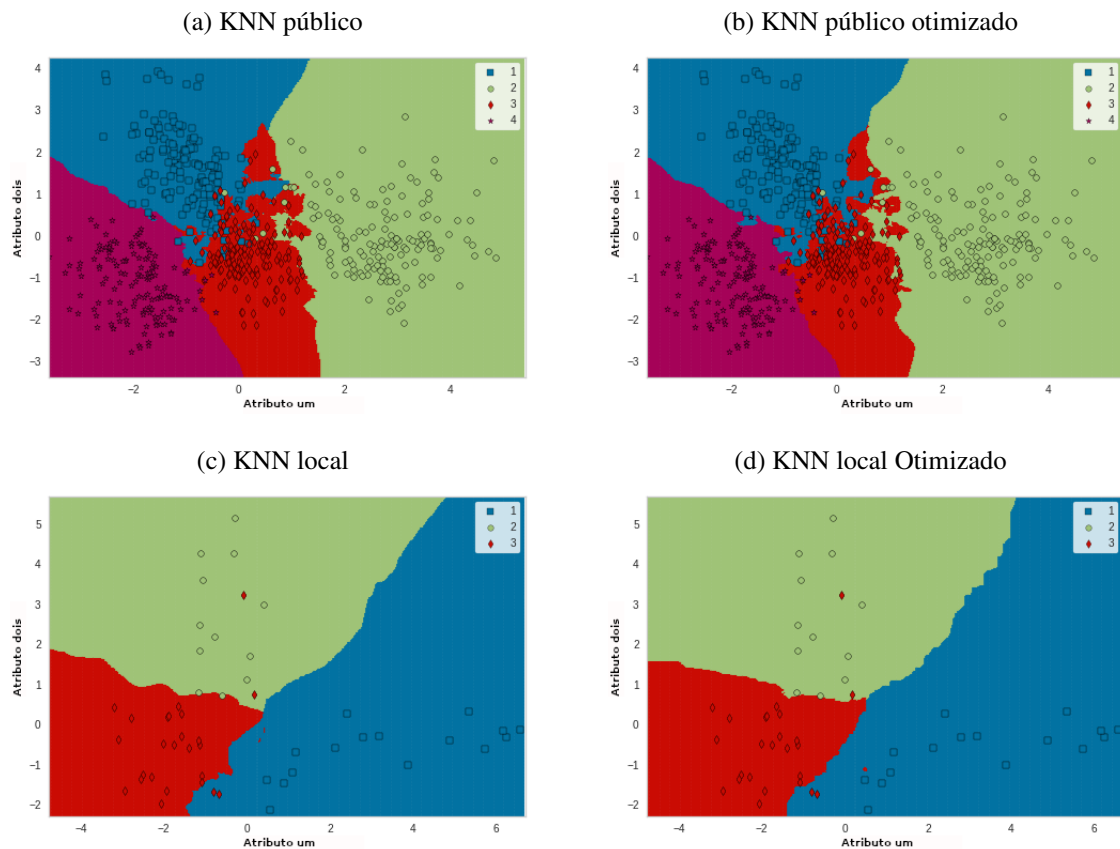
Fonte: Autor (2022).

de curvas ROC e valores AUC individuais para cada classe em cada modelo, uma curva de micro-média e uma curva de macro-média, que estão mais visíveis nos Gráficos 7.(c) e 7.(d) os quais apresentam informações do modelo KNN, mas quando aplicado ao *dataset* construído localmente.

Ao observar a clusterização presente na Figura 8 que denota o resultado de saída dos modelos pode-se ver que, há sutis alterações na classificação com a otimização de qualquer modelo. Percebe-se, na região onde há encontro das classes 2, 3 e 4 pelas Figuras 8.(a) e 8.(b) que há uma clara troca de regiões onde possíveis amostras localizadas nessas regiões mudam de classe entre o modelo normal e o modelo otimizado. Comportamento similar ocorre com a curva de separação entre as classes 1 e 2 para a utilização do *dataset* local visível nas Figuras 8.(c) e 8.(d).

Com base nas informações apresentadas sobre o modelo KNN, pode-se considerar, primeiramente, que a otimização de modelos de aprendizagem de máquina pode até não alterar significativamente os índices e parâmetros de rendimento e eficiência do modelo, mas ainda assim, conforme pode-se ver na Figura 8, muda sim o mapeamento de classificação daquele modelo, que para o caso de um espaço amostral maior pode resultar e mudança nos parâmetros de rendimento e eficiência.

Figura 8 – Comparação Clusters de classificação para modelo KNN



Fonte: Autor (2022).

4.2 Linear Discriminant Analysis (LDA)

Da mesma forma que no modelo KNN, da subseção anterior onde a otimização apresenta melhoria dos parâmetros de rendimento adotados para utilização do *dataset* local, aqui tem-se resultados em que a criação do modelo já apresenta o melhor rendimento em ambos os *datasets*. Vale ressaltar que justamente neste modelo, embora tenha ocorrido melhorias em ambos os *datasets*, esta melhoria só é realmente significativa para o *dataset* local.

Tabela 6 – Resultados para *dataset* público com modelo LDA

		Precisão	AUC	Acurácia	F1 Score
LDA padrão	Média	97,10%	0,9981	96,78%	96,81%
	Desvio	0,0112	0,0017	0,0129	0,0128
LDA Otimizada	Média	97,34%	0,9886	97,07%	97,08%
	Desvio	0,0104	0,0014	0,0125	0,0125

Fonte: Autor (2022).

Considerando as Tabelas 4 e 6 observa-se que ambos os casos apresentam variações dos parâmetros de rendimento uma com redução e a outra com crescimento, respectivamente. Contudo, em ambos os caso, as variações são desprezíveis quando comparam-se o modelo recém criado e o modelo otimizado. Observa-se, também, que em ambos os casos a variação ocorre dentro da faixa prevista no desvio padrão.

O aprimoramento do modelo, apresentado na Tabela 7, é ainda mais significativo que o aprimoramento visto na Tabela 5. O que torna mais notável é justamente a variação percebida na AUC, embora não seja o único parâmetro aprimorado.

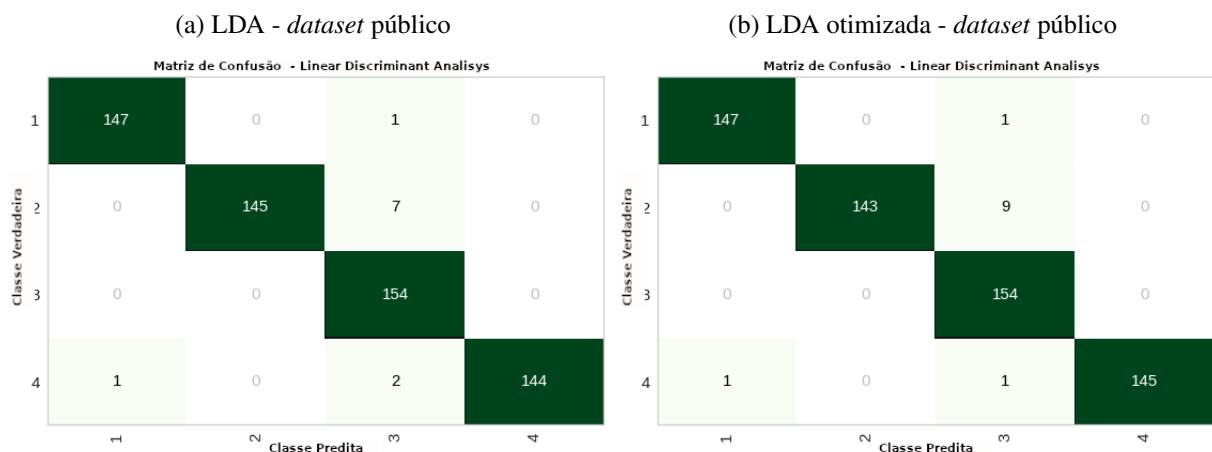
Tabela 7 – Resultados para *dataset* local com modelo LDA

		Precisão	AUC	Acurácia	F1 Score
LDA padrão	Média	87,55%	0,9697	85,13%	84,77%
	Desvio	0,1024	0,0344	0,1151	0,1191
LDA Otimizada	Média	98,09%	1,0000	97,63%	97,58%
	Desvio	0,0293	0	0,0363	0,0371

Fonte: Autor (2022).

Os outros parâmetros alcançaram, também, um aprimoramento, assim como aconteceu no AUC. Para estes outros parâmetros, o aprimoramento foi superior a 10 pontos percentuais e quando comparam-se a variação presente na Tabela 7 com a variação vista no modelo KNN presente na Tabela 5, que ficava em torno de 5 pontos percentuais, estas variações são percebidas no sentido que a otimização ocorre no mínimo com o dobro de ganho na otimização realizada.

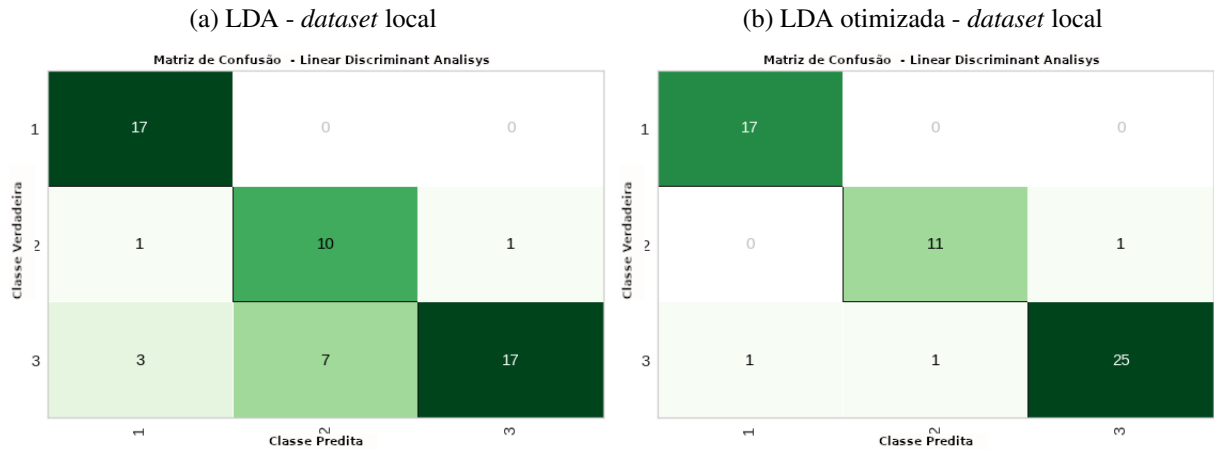
Figura 9 – Comparação Matriz de confusão LDA com *dataset* público



Fonte: Autor (2022).

Ao considerar a classificação apresentada na Figura 9 percebem-se que, se comparada com a Figura 5, a otimização no LDA resulta em mais erros que na versão convencional, o que não ocorreu no KNN, embora nos parâmetros de rendimento o comportamento desses dois modelos tenham apresentados resultados invertidos como observa-se nas Tabelas 4 e 6. Contudo mesmo essas variações de resultados podem ser tratadas como desprezíveis. Esse cenário não se repete ao se utilizar o *dataset* local como mostrado na Figura 10.

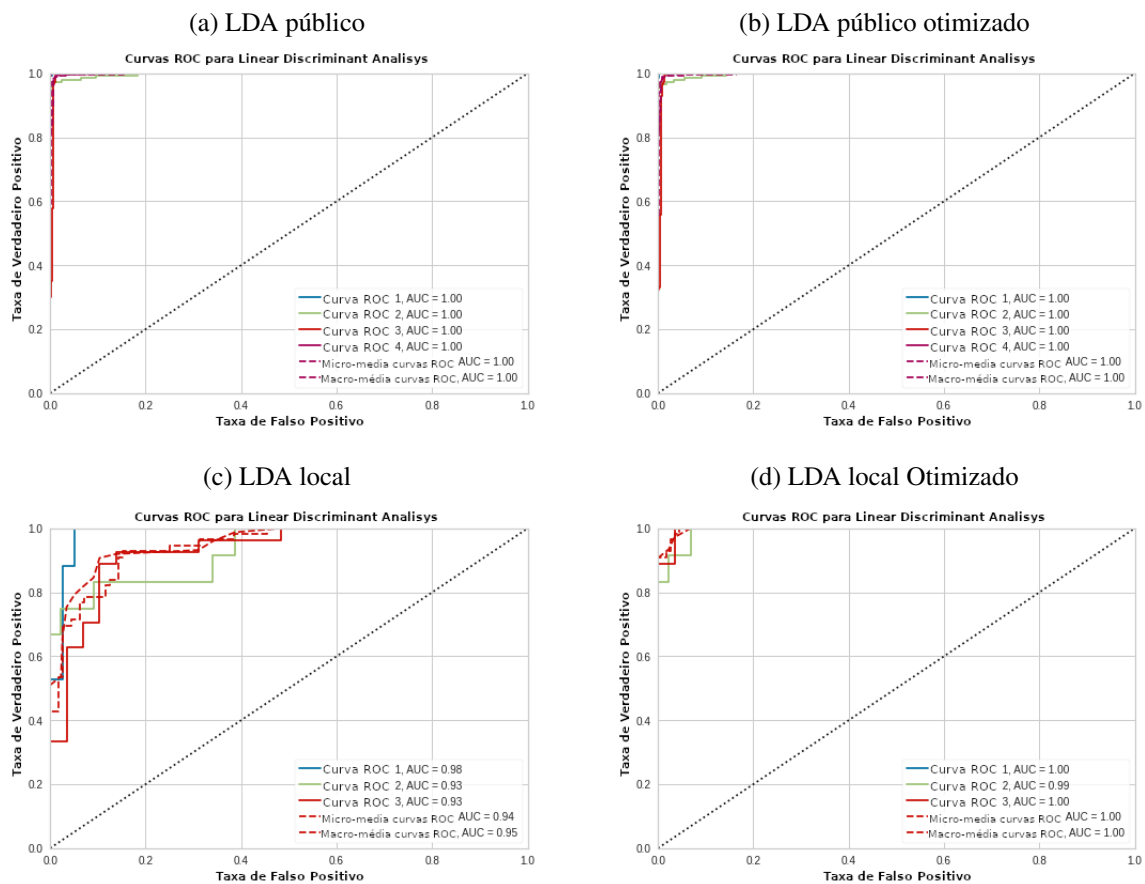
Figura 10 – Comparação Matriz de confusão LDA com dataset local



Fonte: Autor (2022).

Assim como acontece no modelo KNN, onde a mudança de *dataset* torna significativa a otimização pelo visto na Figura 6, aqui, na Figura 10, nota-se o mesmo comportamento de redução dos erros.

Figura 11 – Comparação Gráficos ROC/AUC para modelo LDA

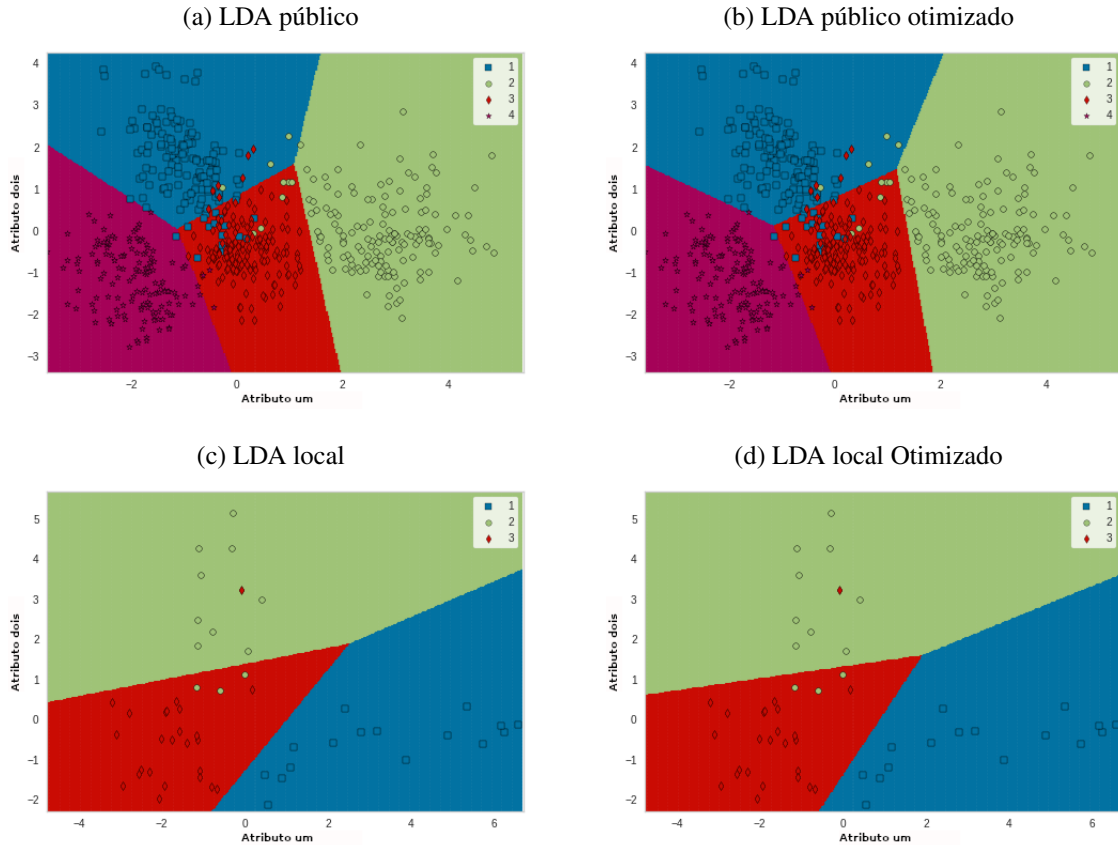


Fonte: Autor (2022).

Observando as as curvas ROC na Figura 11 nota-se que praticamente não há alterações para o modelo utilizando o *dataset* local. Contudo ao utilizar o *dataset* público, há sim um

ganho bastante significativo na AUC que indica um aumento na taxa de acertos do modelo e combinando essa informação com a Tabela 7. Assim, percebe-se que a estabilidade do modelo é ainda maior pois não ocorrem variações na AUC para este modelo.

Figura 12 – Comparação Clusters de classificação para modelo LDA



Fonte: Autor (2022).

Diferente do que foi possível observar na Figura 8, onde em ambos os *datasets* os limites de cada classe foram alterados, aqui na Figura 12 observa-se o exato oposto os limites de cada classe permaneceram visualmente inalterados. Contudo, a classificação, acontece com ambos os *datasets*, mesmo com a presença de *outliers*, realizando a correta identificação mesmo que o ponto referente a amostra apareça em uma área do *cluster* no qual só deveriam aparecer amostras de outra classe.

Observando os resultados apresentados nesta subseção e na anterior, leva-se a crer que em ambos os modelos, mesmo apresentando melhorias nos resultados depois da otimização, ainda deixam uma margem de erro e possui um tempo de execução que é elevado para algumas aplicações, embora ambos os algoritmos vistos até o momento se mantenham com tempo de execução abaixo de meio segundo o que os torna algoritmos bem rápidos, desde que a aplicação final não exija tempos tão baixos assim.

4.3 Extra-Tree(ET)

Para os resultados deste modelo, tem-se a Tabela 8 com alguns dos parâmetros de eficiência que são apresentados na Seção 3.2 - Relatório de Classificação do Capítulo 3, para os *datasets* público e local nessa ordem. Diferente do modelo KNN e do LDA onde a otimização apresentou

resultados consideráveis apenas para o *dataset* local conforme vemos nas Tabelas 5 e 7, contudo mesmo esses resultados que apresentam isoladamente cada um uma taxa relativamente boa de acertos ainda não é suficiente para torná-los os melhores modelos aplicáveis, sem a devida comparação.

Entretanto, a partir deste modelo, observam-se resultados bastante diferentes principalmente para modelos utilizando o *dataset* local com resultados que até mesmo dispensam a otimização, conforme se observa em todos os resultados a partir deste modelo.

Tabela 8 – Resultados para *dataset* público com modelo ET

		Precisão	AUC	Acurácia	F1 Score
ET padrão	Média	0,9996	99,96%	98,36%	98,36%
	Desvio	0,0077	0,0004	0,0079	0,0079
ET Otimizada	Média	0,9813	98,92%	98,00%	98,00%
	Desvio	0,0101	0,0008	0,0110	0,0109

Fonte: Autor (2022).

Os resultados até aqui não foram diferentes em relação aos modelos KNN e LDA. Os resultados presentes na Tabela 8 possuem o mesmo comportamento do modelo KNN, visto na Tabela 4, com uma leve redução nos índices e, em alguns casos, aumento do desvio padrão. Entretanto tais variações são tão pequenas diante dos resultados e do próprio espaço amostral que, assim como aconteceu nos modelos anteriores utilizando o *dataset* público, são variações desprezíveis, pois as mesmas acontecem até mesmo entre iterações do modelo com o mesmo *dataset*.

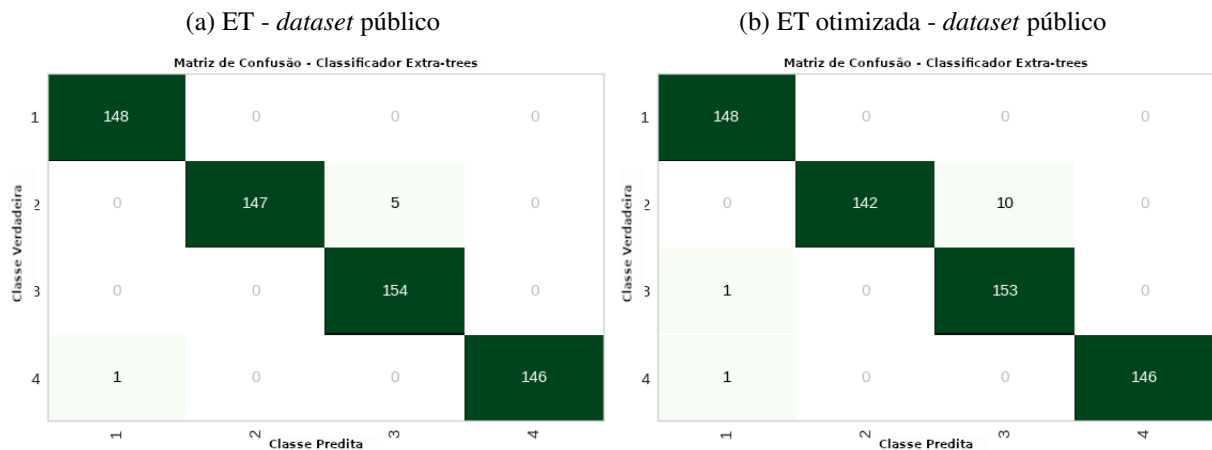
Tabela 9 – Resultados para *dataset* local com modelo ET

		Precisão	AUC	Acurácia	F1 Score
ET padrão	Média	100%	1,0000	100%	100%
	Desvio	0	0	0	0
ET Otimizada	Média	100%	1,0000	100%	100%
	Desvio	0	0	0	0

Fonte: Autor (2022).

Em contraste com os modelos anteriores, especialmente utilizando o *dataset* local, observa-se que sequer é necessária qualquer otimização do modelo, pois o mesmo já alocou corretamente as amostras corretamente em todas as iterações. Isso pode ser observado na Tabela 9 em que os valores de todos os índices aparecem com seus máximos e o desvio padrão deles é 0.

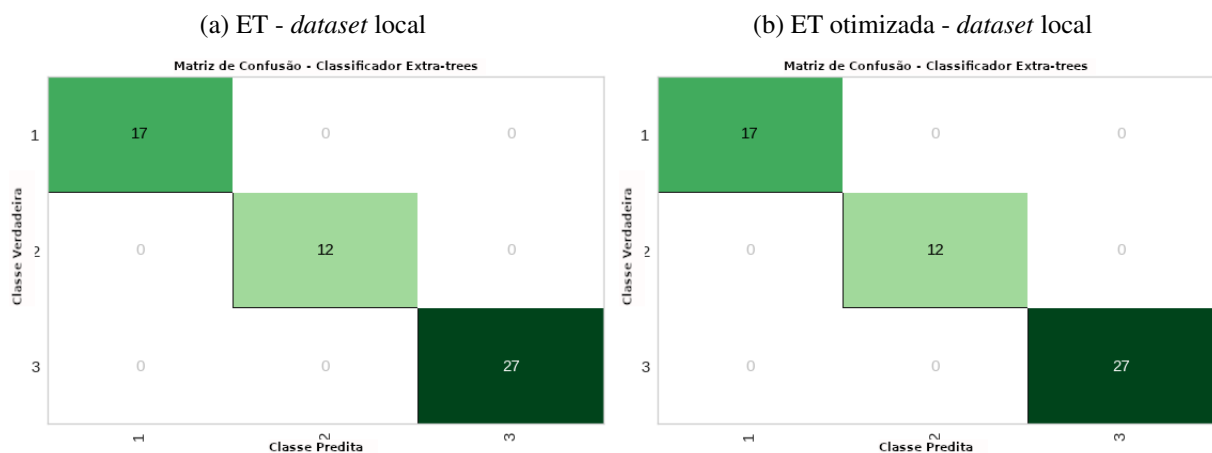
Figura 13 – Comparação Matriz de confusão ET com dataset público



Fonte: Autor (2022).

Na comparação visível na Figura 13 pode-se notar alguns erros, como já apresentado pelos dados constantes na Tabela 8 e assim como nas Figuras 5 e 9, não há alterações significativas nas classificações quando se utiliza o *dataset* público. Contudo, na Figura 14, pode-se verificar que esse cenário de erros não se repete quando é alterado o *dataset* para o local.

Figura 14 – Comparação Matriz de confusão ET com dataset local

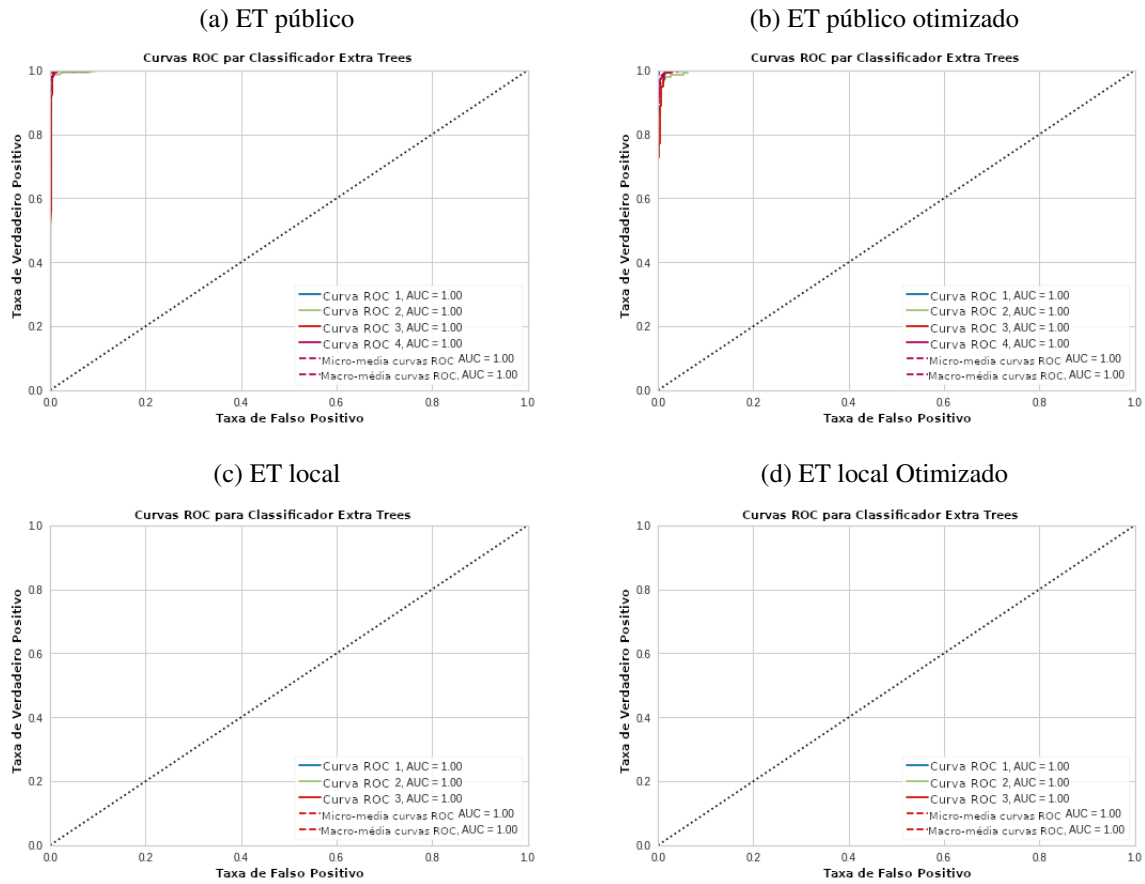


Fonte: Autor (2022).

Pelos resultados apresentados na Figura 14, percebe-se que o modelo para esta situação acertou em 100% dos casos. A princípio pode parecer um *overfitting*, comportamento de modelos de aprendizagem que se ajustam de forma excessiva aos dados de treinamento não acertando corretamente novas amostras apresentadas ao modelo. Essa característica de *overfitting* ficaria visível ao se realizar um processo de validação cruzada, como ocorre no treinamento por meio do *framework*.

Complementando, tem-se que, diferentemente dos modelos KNN e LDA, com as Figuras 6 e 10, para o modelo ET não há a necessidade de aprimoramento em nenhum dos *datasets*, visto que o treinamento sem otimização já apresenta o melhor resultado possível do modelo para os *datasets* utilizados, inclusive com rendimento máximo utilizando o *dataset* local.

Figura 15 – Comparação Gráficos ROC/AUC para modelo ET



Fonte: Autor (2022).

Conforme pode-se observar, as curvas ROC/AUC mostram um comportamento diferente em relação aos modelos anteriores, KNN e LDA, que em ambos os *datasets* observa-se curvas específicas para cada classe e as médias, especialmente nas curvas com o *dataset* local, como observa-se nas Figuras 15.(c) e 15.(d). Diferentemente dos modelos KNN e LDA, tem-se nas Figuras 7 e 11, respectivamente, a existência das curvas, enquanto na Figura 15 não aparecem estas curvas quando se observam os gráficos referentes ao *dataset* local.

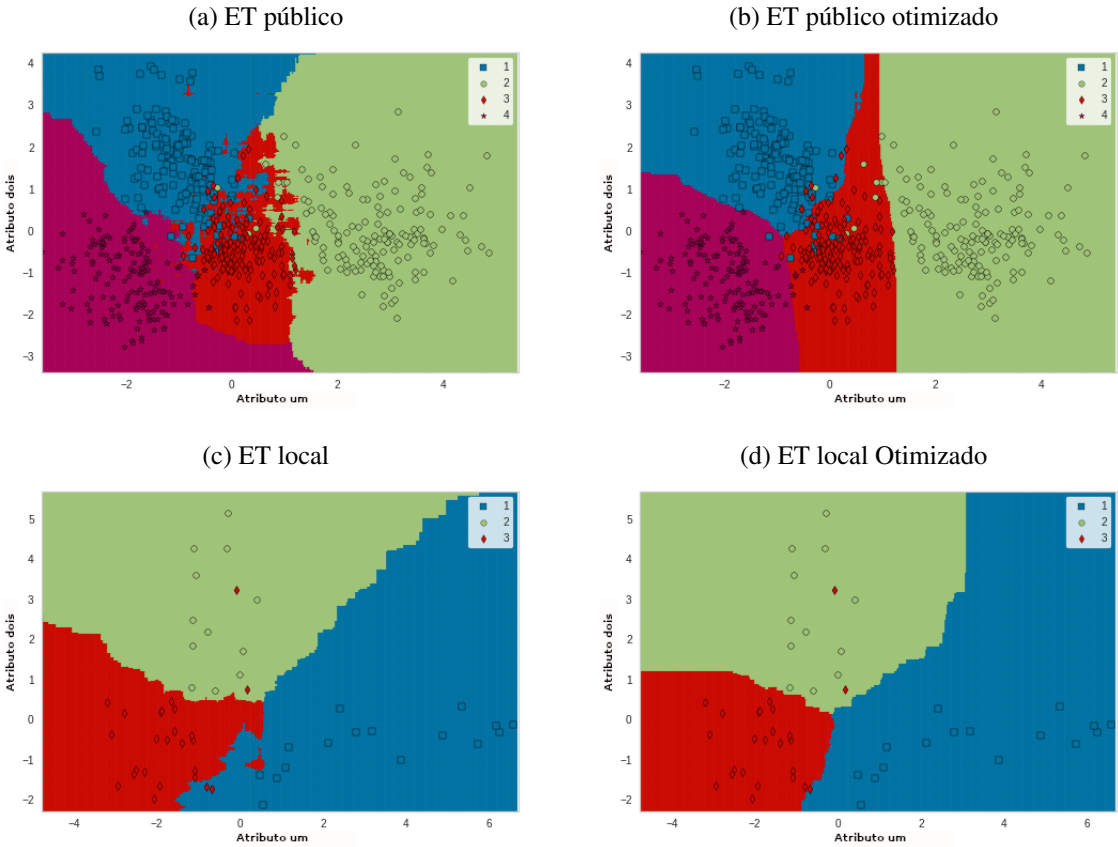
E seguindo na linha de alterações que podem ser consideradas desprezíveis do ponto de vista dos *datasets*, identifica-se na figura 16 que há algumas alterações em ambas as clusterizações mesmo com alteração de áreas mais visíveis em cada *dataset*, se comparado ao KNN pela Figura 8.

4.4 Light Gradient Boost Machine (LightGBM)

Seguindo com a análise dos resultados agora com o Modelo *Light Gradient Boost Machine - LightGBM*, observa-se que, assim como todos os outros modelos, este apresenta um rendimento uniforme independente de se realizar ou não a otimização do modelo, assim como aconteceu com todos os outros modelos previamente analisados.

De forma idêntica ao que aconteceu com o modelo visto na subseção anterior, o modelo *Extra-tree Classifier - ET*, observa-se, conforme tabela de resultados (Tabela 11) que independente do *dataset* utilizado, implicando em um nível de precisão idêntico ao do classificador

Figura 16 – Comparação Clusters de classificação para modelo ET



Fonte: Autor (2022).

Tabela 10 – Resultados para *dataset* público com modelo LightGBM

		Precisão	AUC	Acurácia	F1 Score
LightGBM padrão	Média	98,36%	99,94%	98,28%	98,28%
	Desvio	0,0098	0,0006	0,0102	0,0103
LightGBM Otimizada	Média	98,29%	99,94%	98,21%	98,22%
	Desvio	0,0089	0,0006	0,0092	0,0092

Fonte: Autor (2022).

ET.

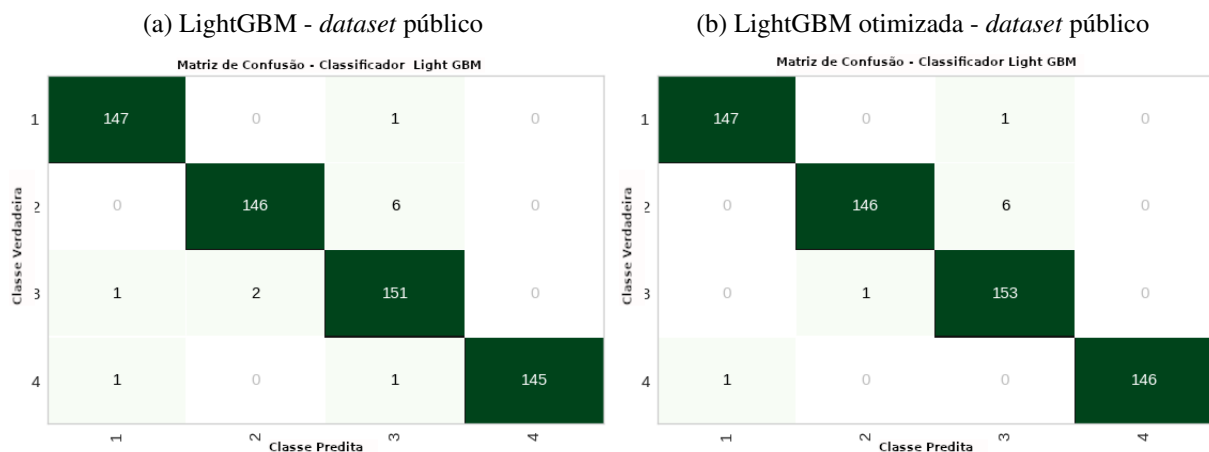
Tabela 11 – Resultados para *dataset* local com modelo LightGBM

		Precisão	AUC	Acurácia	F1 Score
LightGBM padrão	Média	100%	100%	100%	100%
	Desvio	0	0	0	0
LightGBM Otimizada	Média	100%	100%	100%	100%
	Desvio	0	0	0	0

Fonte: Autor (2022).

Da mesma forma que ocorreu com o ET na Tabela 9, especialmente utilizando o *dataset* local, observa-se que sequer é necessária qualquer otimização do modelo, pois o mesmo já alocou corretamente todas as amostras corretamente em todas as iterações. Isso pode ser observado pela Tabela 11, em que os valores de todos os índices aparecem com seu máximos e o desvio padrão deles é 0 e se identifica que ambas as tabelas apresentam exatamente o mesmo resultado.

Figura 17 – Comparação matriz de confusão LightGBM com *dataset* público

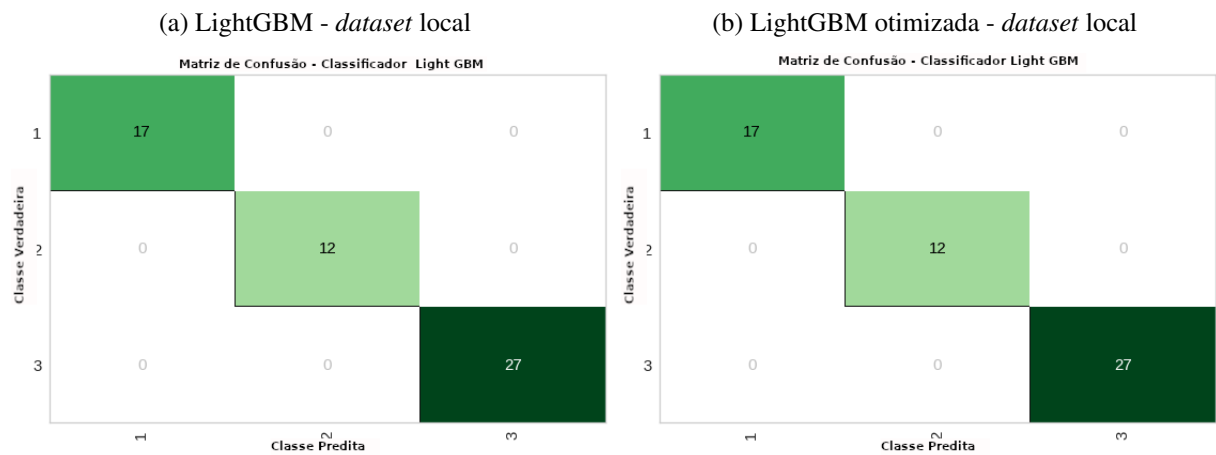


Fonte: Autor (2022).

A Figura 17 apresenta o resultado similar a todos os modelos previamente analisados, em que a variação de erros de classificação entre o modelo normal na Figura 17.(a) e o modelo otimizado na Figura 17.(b) é completamente desprezível.

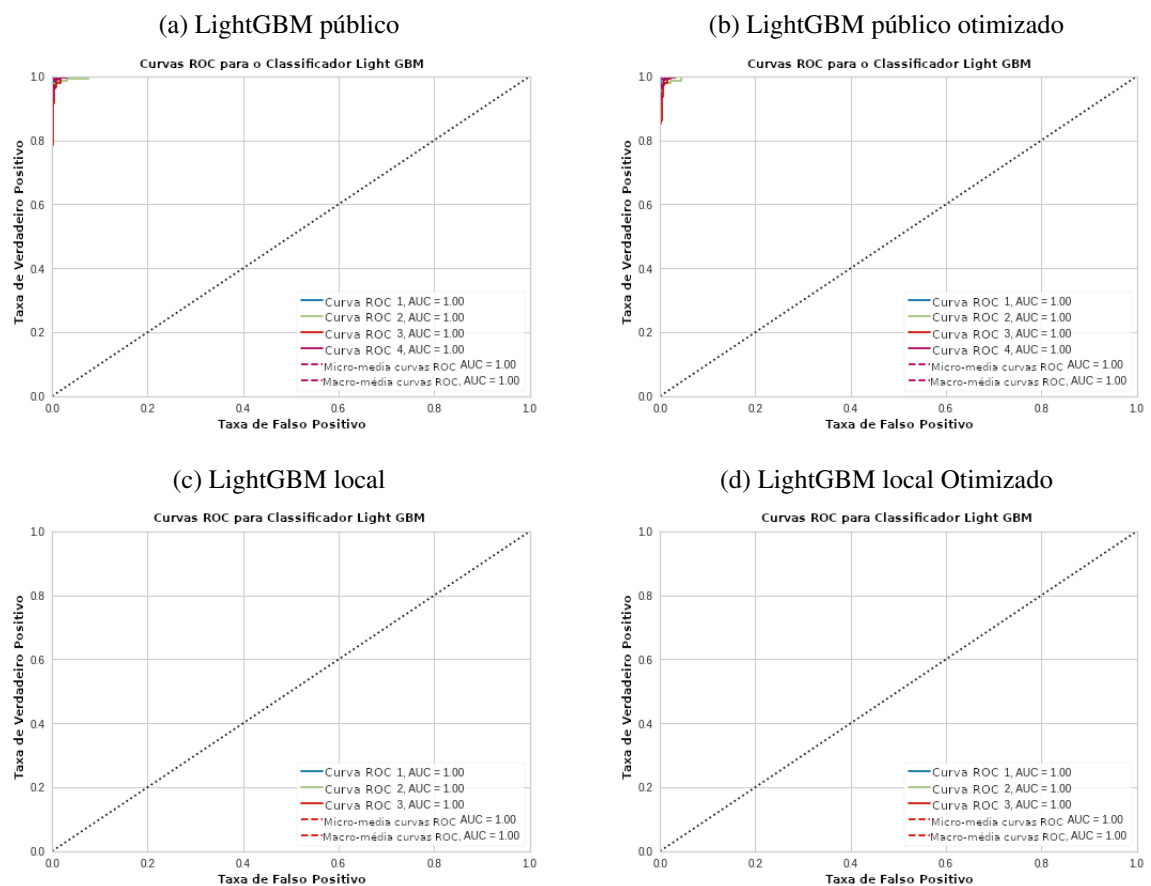
Complementando, tem-se que diferentemente dos modelos KNN e LDA, com as Figuras 6 e 10, da mesma maneira que o modelo ET na Figura 14 não há a necessidade de aprimoramento em nenhum dos *datasets*, visto que o treinamento sem otimização já apresenta o melhor resultado possível do modelo para os *datasets* utilizados, inclusive com rendimento máximo utilizando o *dataset* local.

Figura 18 – Comparação matriz de confusão LightGBM com dataset local



Fonte: Autor (2022).

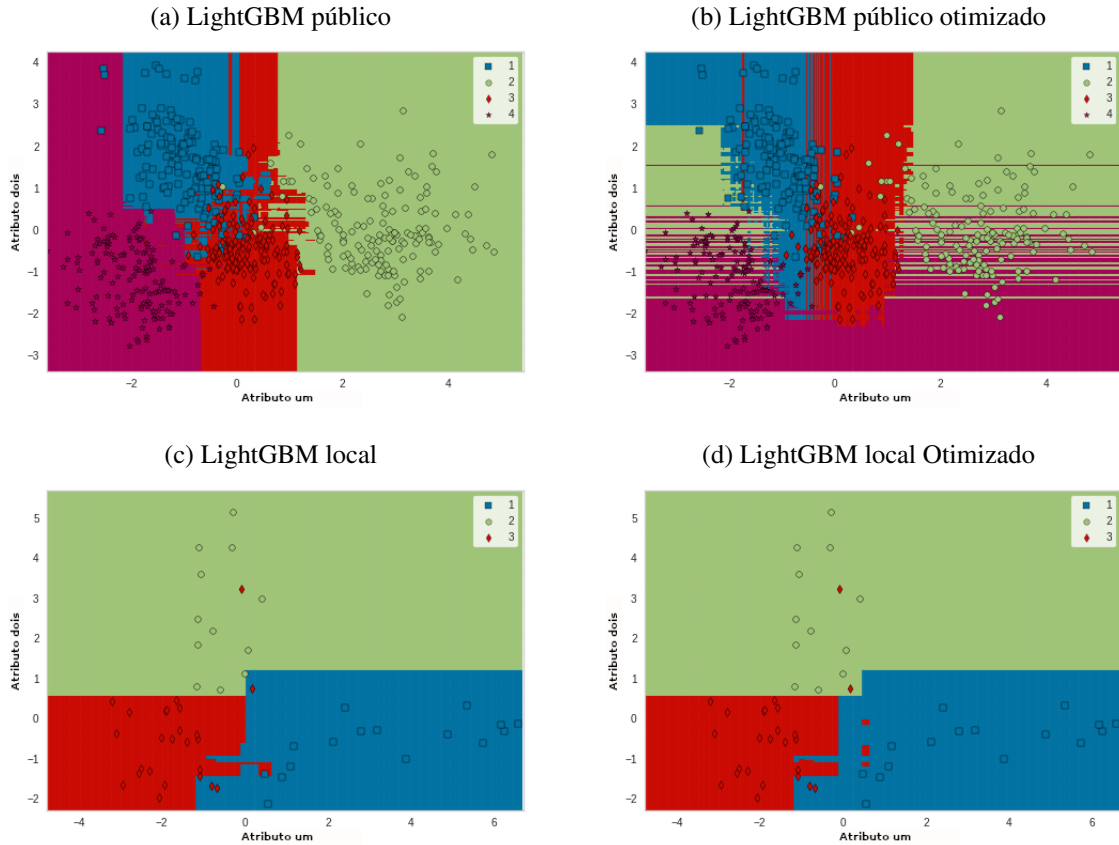
Figura 19 – Comparação Gráficos ROC/AUC para modelo LightGBM



Fonte: Autor (2022).

A taxa de acerto de 100% tornou inviável a visualização das curvas ROC nos gráficos ROC/AUC para o *dataset* local assim como o modelo ET.

Figura 20 – Comparação clusters de classificação para modelo LightGBM



Fonte: Autor (2022).

Conforme pode-se observar as curvas ROC/AUC mostram um comportamento diferente em relação aos modelos anteriores, KNN e LDA, que em ambos os *datasets* observam-se curvas específicas para cada classe e as médias, especialmente nas curvas com o *dataset* local, como observado nas Figuras 19.(c) e 19.(d). Diferentemente dos modelos KNN e LDA, tem-se nas Figuras 7 e 11, respectivamente, a existência das curvas, enquanto na Figura 19 não aparecem estas curvas quando se observam os gráficos referentes ao *dataset* local.

Já com relação ao cluster de classificação deste modelo como apresentado na Figura 20 percebe-se que não há alteração das áreas de cada classe com o *dataset* local mesmo com a otimização, o que não se repete com o *dataset* público em que se pode ver uma mudança muito significativa nas áreas de cada classe bem como no padrão de cobertura aplicado pelo modelo.

4.5 Random Forest (RF^2)

Detalhando melhor os resultados de cada modelo, tem-se os resultados apresentados na Tabela 12, com alguns dos parâmetros de eficiência que foram apresentados na Seção 3.2 - Relatório de Classificação do Capítulo 3, para os *datasets* público e local, nessa ordem. Diferente do modelo KNN, em que a otimização só funciona para utilização do *dataset* local, aqui tem-se resultados onde a criação do modelo já apresenta o melhor rendimento em ambos os *datasets*.

Tabela 12 – Resultados para *dataset* público com modelo RF^2

		Precisão	AUC	Acurácia	F1 Score
RF^2 padrão	Média	98,20%	99,93%	98,14%	98,14%
	Desvio	0,0109	0,0007	0,0112	0,0112
RF^2 Otimizada	Média	97,66%	99,86%	97,57%	97,57%
	Desvio	0,0086	0,0011	0,0091	0,0091

Fonte: Autor (2022).

Invariavelmente assim como todos os modelos apresentados neste trabalho, com o RF^2 não foi diferente o comportamento de resultados utilizando o *dataset* público. Vale ressaltar que este modelo apresenta os tempos médios de execução mais elevados, quando da comparação inicial realizada pelo *framework* com **0,542 s** para o *dataset* público e **0,463 s** para o *dataset* local, que como dado isolado não significa de maneira alguma ser um tempo de execução elevado, mas se a aplicação exige uma frequência de leituras e execução igual ou superior a duas vezes por segundo esse já pode significar um fator excludente para este modelo, independente da maior taxa de acerto e da maior acurácia.

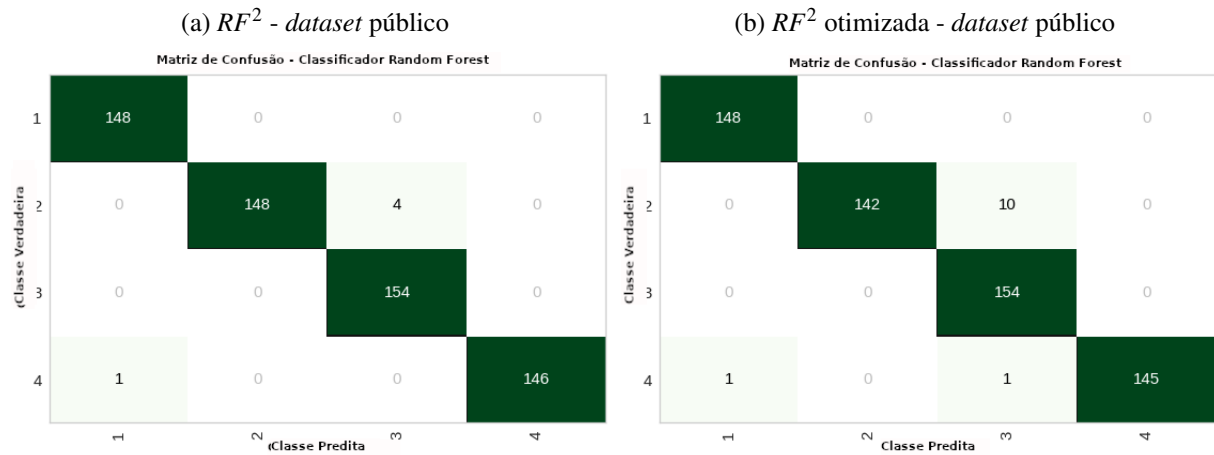
Tabela 13 – Resultados para *dataset* local com modelo RF^2

		Precisão	AUC	Acurácia	F1 Score
RF^2 padrão	Média	100%	100%	100%	100%
	Desvio	0	0	0	0
RF^2 Otimizada	Média	100%	100%	100%	100%
	Desvio	0	0	0	0

Fonte: Autor (2022).

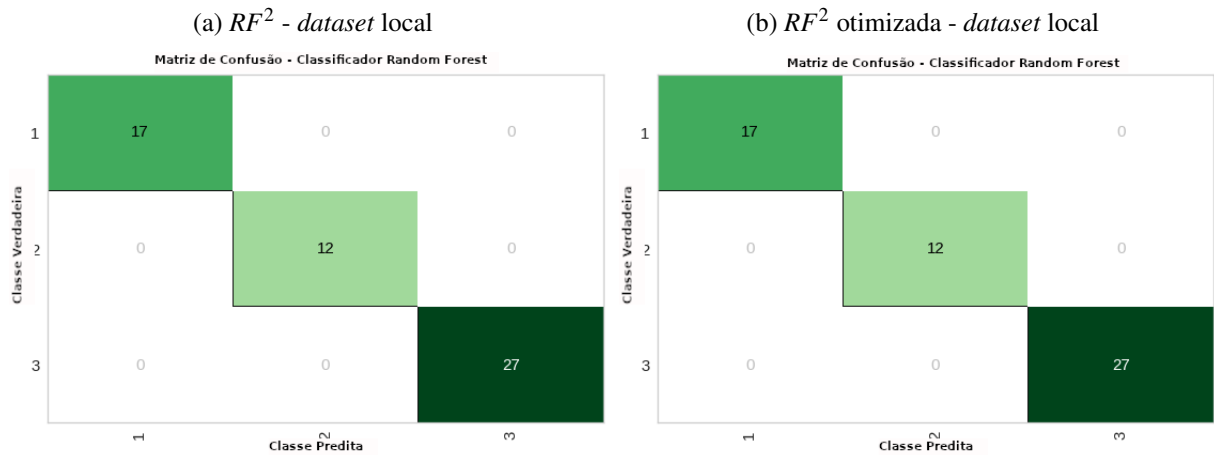
Em contraste com os modelos anteriores, especialmente utilizando o *dataset* local, tem-se que sequer é necessária qualquer otimização do modelo, pois o mesmo já alocou corretamente todas as amostras corretamente em todas as iterações. e isso pode ser observado por meio da Tabela 9, onde os valores de todos os índices aparecem com seu máximos e o desvio padrão deles é 0.

Na comparação visível na Figura 21, pode-se notar que a quantidade de erros numericamente mais que dobra entre o modelo básico e o otimizado. Contudo, conforme já foi mencionado antes sobre todos os outros modelos, esse comportamento é aceitável dado o tamanho do espaço amostral, bem como por ocorrer na matriz de confusão um resultado pontual da última execução em cada caso, que pode não ter apresentado os melhores parâmetros analisados quando se aplica a validação cruzada.

Figura 21 – Comparação Matriz de confusão RF^2 com dataset público

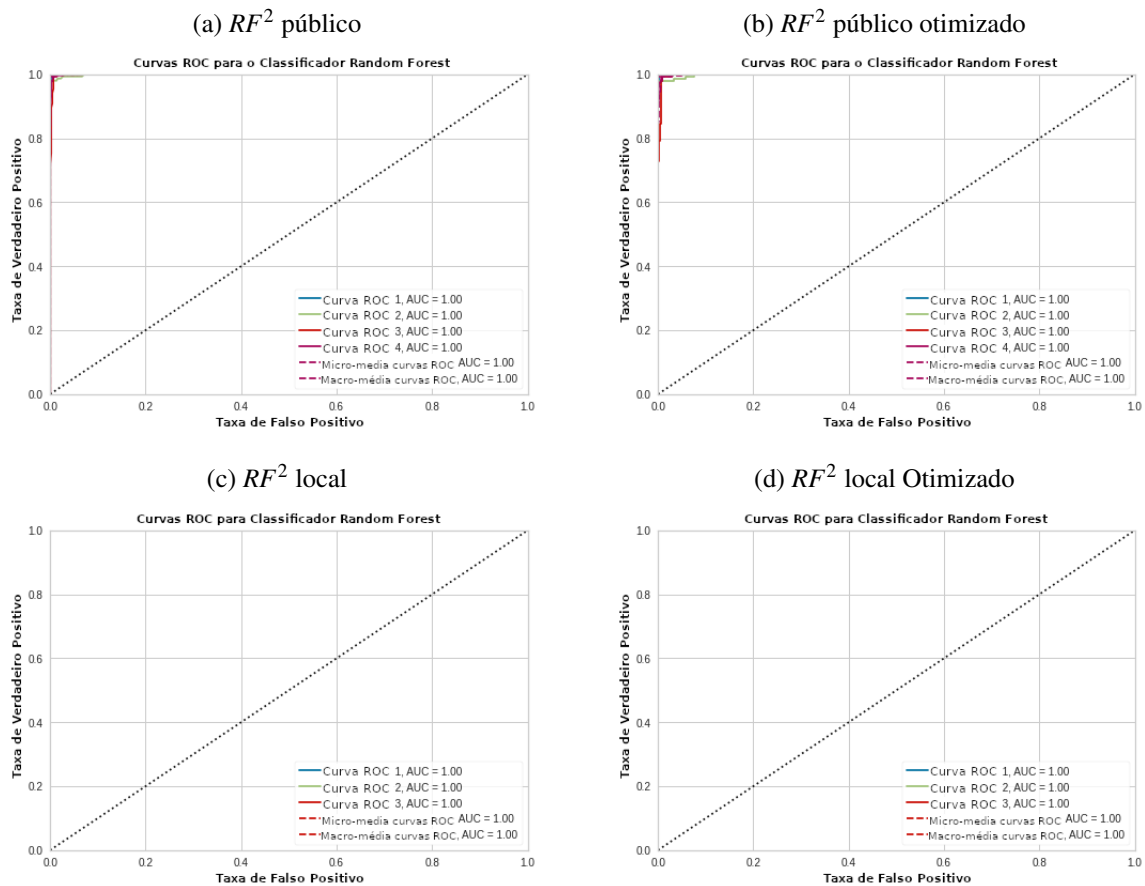
Fonte: Autor (2022).

Esse comportamento não se mostrou incomum pois apenas os modelos LightGBM e KNN utilizando o *dataset* público tiveram realmente uma redução de erros na comparação via matriz de confusão. Um detalhe que chama a atenção é que os modelos ET e RF^2 apresentam esse cenário de dobrar os erros, contudo essa informação por apenas não traz indícios de perda de precisão do modelo pois todos os demais parâmetros demonstram a estabilidade do modelo com o *dataset* público.

Figura 22 – Comparação Matriz de confusão RF^2 com dataset local

Fonte: Autor (2022).

Pelos resultados apresentados na Figura 22 percebe-se que o modelo para esta situação acerta em 100% dos casos, assim como acontece com os modelos LightGBM e ET, mais uma vez tornando completamente desnecessária a otimização do modelo. Contudo para fins de coleta de dados assim como dos modelos anteriores foi realizada a otimização deste modelo também.

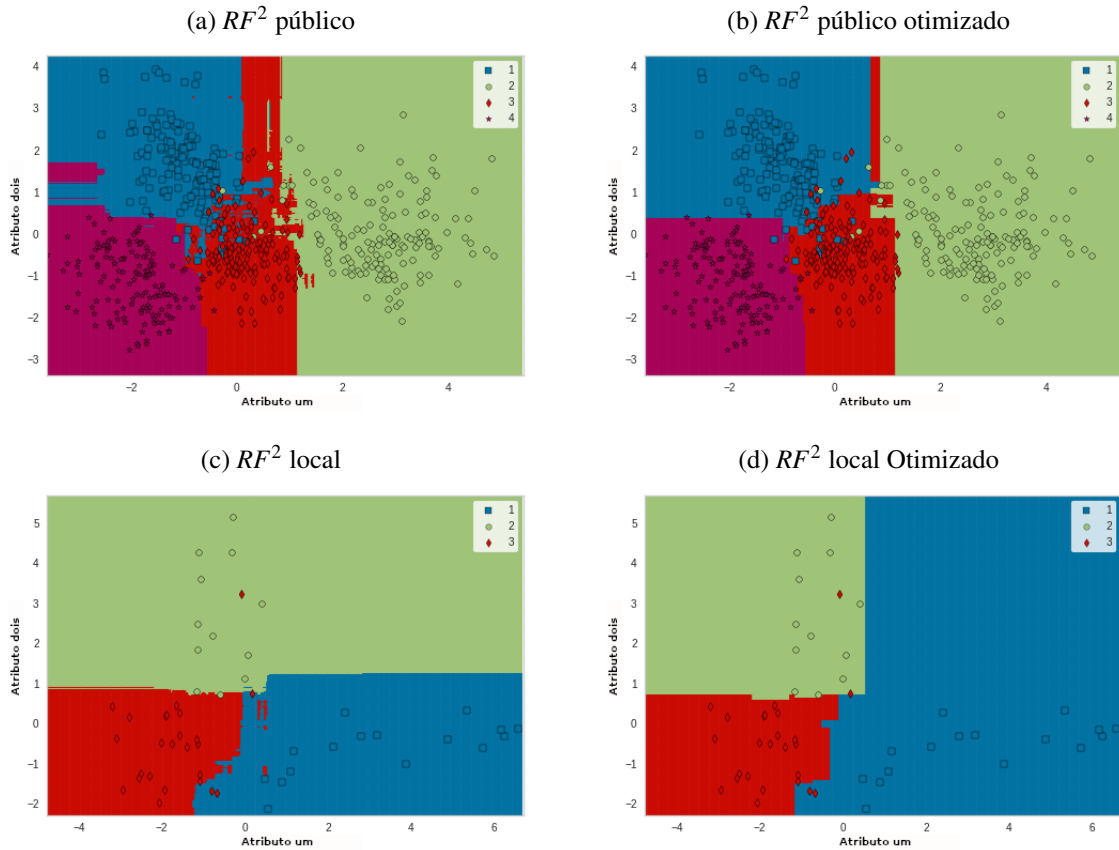
Figura 23 – Comparação Gráficos ROC/AUC para modelo RF^2 

Fonte: Autor (2022).

Da mesma forma que se vislumbram na Figura 15 e na figura 19, Curvas ROC/AUC dos

modelos ET e LightGBM, respectivamente, é notório que para o modelo RF^2 , quando se utiliza o *dataset* local, o comportamento das curvas ROC/AUC é o mesmo visto nas figuras citadas acima: gráfico mostrando acerto total para os modelos ET, LightGBM e RF com AUC 1 e apenas o tempo é diferencial.

Figura 24 – Comparação clusters de classificação para modelo RF^2



Fonte: Autor (2022).

No gráfico presente na Figura 24, observa-se que, da mesma forma que todos os demais modelos analisados, ocorrem alterações para o cluster de cada classe, tanto para *dataset* público como para o local. Contudo a mudança de cluster mais expressiva nesse caso, acontece justamente com o *dataset* local com uma mudança de classe que abrange cerca de 25% da área do gráfico, conforme pode-se identificar comparando-se os Gráficos 24.(c) e 24.(d), em que a classe 1 cede área para a classe 2, após a otimização do modelo.

4.6 Considerações Finais

Com base em todos os dados e resultados apresentados, pode-se inferir que todos os modelos apresentam resultados adequados, considerando o *dataset* público e o principal fator diferencial deles é o tempo de execução do algoritmo, conforme apresentado na Tabela 2. Entretanto, ao substituir o *dataset* pelo contruído localmente ocorrem algumas mudanças no resultado, ressaltando as capacidades de classificação de 3 modelos que atingiram a precisão e acurácia máximos.

Os modelos que apresentaram melhores resultados passam, portanto, a serem os classificadores: *Extra-tree* (ET), *Light Gradient Boost Machine* (LightGBM) e *Random Florest* (RF^2),

e novamente o tempo de execução é um fator diferencial entre os modelos, conforme observado na Tabela 3, e, portanto, deve considerado conforme a necessidade da aplicação.

Para fins de exemplo, pode-se analisar a importância sobre a consideração do tempo de execução na hora de escolha do modelo, tomando por base apenas os dados do *dataset* público. Com a necessidade de localização constante de um ativo móvel, em que a leitura do RSSI é realizada 2 vezes a cada segundo e após o ajuste do vetor de dados enviado ao modelo para cálculo da localização topográfica, diante desse cenário torna-se inviável a utilização dos modelos ET e RF^2 , pois o primeiro tem um tempo de execução muito próximo a 0,5 s e o segundo já ultrapassa esse tempo, e o modelo mais interessante para esta aplicação passa a ser LDA, pois atende os requisitos de tempo sem grandes perdas de precisão e acurácia.

Em um segundo exemplo: pode-se tomar por base agora os resultados obtidos com o *dataset* local. A necessidade de localização nos exige acurácia e precisão acima de 95%, bem como permite que o cálculo de localização seja realizado apenas 1 vez por segundo. Diante desse novo cenário, os modelos que tornam-se inviáveis são KNN e LDA, pois embora possuam bons tempos de execução suas precisão e acurácia estão abaixo do exigido. Os demais modelos para este cenário são aplicáveis, ficando ao interesse do desenvolvedor escolher entre eles, embora a sugestão mais interessante seja o modelo *LightGBM* por apresentar o menor tempo de execução.

Seguindo a linha dos exemplos apresentados acima, aplicando estes modelos à forma como foi construído o *dataset* local, com os resultados apresentados por ele, os modelos ET e RF^2 , que são desenvolvidos com base no algoritmo de aprendizagem *Decision Tree*, em que ambos possuem intrinsecamente em seus algoritmos centenas de vezes o modelo de inspiração, justificando assim a velocidade de execução menor em relação aos outros modelos.

5 CONCLUSÃO

Dados os resultados apresentados e discutidos no capítulo anterior pode-se, com base também nos pre-requisitos exigidos em cada aplicação, definir dentre os modelos pesquisados o mais eficiente, rápido e adequado a demanda, não restrito apenas aos modelos pesquisados, mas todos os que são fornecidos pelo *framework* seguindo o mesmo procedimento adotado na presente pesquisa. Assim, torna-se a utilização do *framework PyCaret* um ambiente de simulação muito interessante para testes de modelos de aprendizagem de máquina.

Apresentados os resultados de cada modelo, é possível realizar comparação entre estes, no mesmo modelo com e sem otimização fornecida pelo *framework*, e para o mesmo modelo com alterações de *dataset*, embora as Tabelas 2 e 3 apresentem uma prévia dos modelos sem otimização e em ordem decrescente dos parâmetros de eficiência, percebem-se que esses valores, após a otimização em alguns casos, sofrem alterações para melhor ou para pior, mesmo que na maioria destes essas variações sejam desprezíveis para fins de aplicação prática.

No Capítulo 4 é realizada a apresentação de resultados para os modelos *K-Nearest Neighbors* (KNN), *Linear Discriminant Analysis* (LDA), *Extra-Tree Classifier* (ET), *Light Gradient Boost Machine* (*LightGBM*) e *Random Florest* (RF^2). Esta comparação é realizada com intuito de gerar informações para uma escolha mais assertiva do modelo de acordo com a demanda, e mesmo que o presente trabalho tenha apresentado resultados com apenas estes 5 modelos, o *framework* utilizado possui mais de 20 modelos de classificadores.

Ainda sobre a comparação realizada no capítulo anterior, em que foram apresentados 2 *datasets* a cada modelo, um deles está disponível de forma pública na *web*, o outro é construído localmente utilizando uma infraestrutura híbrida de *beacons ESP* e *Access Points* residenciais, por meio de escaneamentos de intensidades RSSI, a cada meio segundo, com intuito de apresentar também uma aplicação prática experimental. O mais interessante de todos os resultados é que a ordem de eficiência dos modelos foi alterada quando pareadas as tabelas de cada *dataset*, indicando assim que o *dataset* público serviu mais como elemento de calibração dos modelos do que mesmo base de ordenação de eficiência. Além disso, o tempo de execução mostra-se como um dos fatores de grande importância.

Considerando trabalhos futuros, podem-se propor as seguintes atividades:

- Localização em tempo real de ativos;
- Localização de ativos em tempo real com RSSI de *Bluetooth*;
- Localização com sinais de telefonia 5G;
- Localização em protocolo *Wi-Fi* 6;
- Localização combinada em RSSI de *Wi-Fi* e *Bluetooth*;
- Uso de RSSI de *Wi-Fi* ou *bluetooth* para validação de rotas;
- Programação de rotas em veículos autônomos com RSSI;
- Detecção de presença por RSSI de ativos.

REFERÊNCIAS

- ALI, M. Pycaret: An open source, low-code machine learning library in python. **PyCaret version**, v. 2, 2020.
- AMOR, N. B. et al. Efficient embedded software implementation of a low cost robot localization system. In: IEEE. **2019 IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS)**. [S.l.], 2019. p. 2–6.
- ASHRAF, I.; HUR, S.; PARK, Y. mpilot-magnetic field strength based pedestrian indoor localization. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 18, n. 7, p. 2283, 2018.
- ASIMOV, I. **I. Robot**. [S.l.]: Narkaling Productions., 1940.
- ASIMOV, I. **Eu, robô**. [S.l.]: Aleph, 2015.
- BATCHU, R. K.; SEETHA, H. A hybrid detection system for ddos attacks based on deep sparse autoencoder and light gradient boost machine. **Journal of Information & Knowledge Management**, World Scientific, p. 2250071, 2022.
- BEN-ARI, M.; MONDADA, F. **Elements of robotics**. [S.l.]: Springer Nature, 2017.
- BHATT, R. Fuzzy-rough approaches for pattern classification: Hybrid measures, mathematical analysis, feature selection algorithms, decision tree algorithms, neural learning, and applications. **Decision tree algorithms, neural learning, and applications**. Amazon books, 2005.
- BLOCH, V.; PASTELL, M. Monitoring of cow location in a barn by an open-source, low-cost, low-energy bluetooth tag system. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 20, n. 14, p. 3841, 2020.
- ČAPEK, K. **Rossumovi univerzální roboti**. [S.l.]: Prag, Čekya, 1921.
- CHEN, C. L.; HUANG, S. H.; ZHOU, J. H. Mobile robot localization by tracking built-in encoders. In: IEEE. **2014 International Symposium on Computer, Consumer and Control**. [S.l.], 2014. p. 840–843.
- CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995.
- COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE transactions on information theory**, IEEE, v. 13, n. 1, p. 21–27, 1967.
- DELLAERT, F. et al. Monte carlo localization for mobile robots. In: IEEE. **Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C)**. [S.l.], 1999. v. 2, p. 1322–1328.
- DUDANI, S. A. The distance-weighted k-nearest-neighbor rule. **IEEE Transactions on Systems, Man, and Cybernetics**, IEEE, n. 4, p. 325–327, 1976.
- EZPELETA, S. et al. Rf-based location using interpolation functions to reduce fingerprint mapping. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 15, n. 10, p. 27322–27340, 2015.

- FEURER, M. et al. Auto-sklearn 2.0: Hands-free automl via meta-learning. **arXiv:2007.04074 [cs.LG]**, 2020.
- FISHER, R. A. The use of multiple measurements in taxonomic problems. **Annals of eugenics**, Wiley Online Library, v. 7, n. 2, p. 179–188, 1936.
- GAGANA, B. New insights from old data: Multimodal classification of schizophrenia using automated deep learning configurations. **bioRxiv**, Cold Spring Harbor Laboratory, p. 2020–11, 2021.
- GIULIANO, R. et al. Indoor localization system based on bluetooth low energy for museum applications. **Electronics**, Multidisciplinary Digital Publishing Institute, v. 9, n. 6, p. 1055, 2020.
- HO, T. K. Random decision forests. In: IEEE. **Proceedings of 3rd international conference on document analysis and recognition**. [S.l.], 1995. v. 1, p. 278–282.
- JIN, H.; SONG, Q.; HU, X. Auto-keras: An efficient neural architecture search system. In: ACM. **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. [S.l.], 2019. p. 1946–1956.
- KE, G. et al. Lightgbm: A highly efficient gradient boosting decision tree. In: GUYON, I. et al. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2017. v. 30. Disponível em: <<https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>>.
- KHAN, F.; NGUANG, S. K. Distributed localization algorithm for wireless sensor networks using range lookup and subregion stitching. **IET Wireless Sensor Systems**, Wiley Online Library, v. 11, n. 5, p. 179–205, 2021.
- KHAN, M. Z. et al. Comparative analysis of artificial intelligence on contactless human activity localization. In: IEEE. **2022 International Telecommunications Conference (ITC-Egypt)**. [S.l.], 2022. p. 1–3.
- KIM, H.; CHOI, Y. Comparison of three location estimation methods of an autonomous driving robot for underground mines. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 10, n. 14, p. 4831, 2020.
- KIRIY, E.; BUEHLER, M. Three-state extended kalman filter for mobile robot localization. **McGill University, Montreal, Canada, Tech. Rep. TR-CIM**, Citeseer, v. 5, p. 23, 2002.
- KOPITAR, L. et al. Early detection of type 2 diabetes mellitus using machine learning-based prediction models. **Scientific reports**, Nature Publishing Group, v. 10, n. 1, p. 1–12, 2020.
- LANTZ, B. **Machine learning with R**. [S.l.]: Packt publishing ltd, 2013.
- LE, T. T.; FU, W.; MOORE, J. H. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. **Bioinformatics**, Oxford University Press, v. 36, n. 1, p. 250–256, 2020.
- LI, Q. et al. Localization in unstructured environments: Towards autonomous robots in forests with delaunay triangulation. **Remote Sensing**, Multidisciplinary Digital Publishing Institute, v. 12, n. 11, p. 1870, 2020.
- LUCKMAN, J. **Principles of Operations Research, with Applications to Managerial Decisions**. [S.l.]: Wiley Online Library, 1972.

LUO, J. et al. Indoor multifloor localization method based on wifi fingerprints and lda. **IEEE Transactions on Industrial Informatics**, IEEE, v. 15, n. 9, p. 5225–5234, 2019.

MANNAY, K. et al. Characterization of an ultrasonic local positioning system for 3d measurements. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 20, n. 10, p. 2794, 2020.

MARCU, D.; DANUBIANU, M. Sentiment analysis from images-comparative study of sai-g and sai-c models' performances using automl vision service from google cloud and clarifai platform. **International Journal of Computer Science & Network Security**, International Journal of Computer Science & Network Security, v. 21, n. 9, p. 179–184, 2021.

MARWAN, A.; MOURAD, O.; MOUSA, H. A survey of fuzzy logic in wireless localization. **EURASIP Journal on Wireless Communications and Networking**, Springer Nature BV, v. 2020, n. 1, 2020.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943.

NAGARAJAH, T.; PORAVI, G. A review on automated machine learning (automl) systems. In: **2019 IEEE 5th International Conference for Convergence in Technology (I2CT)**. [S.l.: s.n.], 2019. p. 1–6.

OLIVEIRA, L. et al. Rssi-based relative localisation for mobile robots. **Ad Hoc Networks**, Elsevier, v. 13, p. 321–335, 2014.

RAO, C. R. The utilization of multiple measurements in problems of biological classification. **Journal of the Royal Statistical Society. Series B (Methodological)**, JSTOR, v. 10, n. 2, p. 159–203, 1948.

ROHRA, J. G. et al. User localization in an indoor environment using fuzzy hybrid of particle swarm optimization & gravitational search algorithm with neural networks. In: SPRINGER. **Proceedings of Sixth International Conference on Soft Computing for Problem Solving**. [S.l.], 2017. p. 286–295.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

SHI, Z.-B.; SUN, J.-b. Mobile robots odor localization with a genetic algorithm. In: IEEE. **2012 Fourth International Conference on Computational Intelligence and Communication Networks**. [S.l.], 2012. p. 875–878.

TAI, F.-C. et al. Decentralized eif-based global localization using dead-reckoning, kinect and laser scanning for autonomous omnidirectional mobile robot. In: IEEE. **2014 International Conference on Advanced Robotics and Intelligent Systems (ARIS)**. [S.l.], 2014. p. 85–90.

TCHÁPEK, K. A fábrica de robôs. **Traduzido por Vera Machac. São Paulo: Hedra**, p. 37, 2010.

WANG, C. et al. Secure and reliable indoor localization based on multi-task collaborative learning for large-scale buildings. **IEEE Internet of Things Journal**, IEEE, 2021.

WERBOS, P. J. **The roots of backpropagation: from ordered derivatives to neural networks and political forecasting**. [S.l.]: John Wiley & Sons, 1994. v. 1.

WESTON, J.; WATKINS, C. et al. Support vector machines for multi-class pattern recognition. In: **Esann**. [S.l.: s.n.], 1999. v. 99, p. 219–224.

YAO, L.; LI, F. Mobile robot localization based on vision and multisensor. **Journal of Robotics**, Hindawi, v. 2020, 2020.

YAO, Y. et al. A rssi-based distributed weighted search localization algorithm for wsns. **International Journal of Distributed Sensor Networks**, SAGE Publications Sage UK: London, England, v. 11, n. 4, p. 293403, 2015.