UNIVERSIDADE FEDERAL DE MINAS GERAIS - UFMG
Programa de Pós-Graduação em Engenharia Elétrica - PPGEE
Machine Intelligence and Data Science Laboratory - MINDS

Samara Silva Santos

Indução de Árvores de Decisão Oblíquas como Explicadores de Predições por Modelos de Aprendizado de Máquina

Belo Horizonte - Minas Gerais

Julho, 2022

Samara Silva Santos

Indução de Árvores de Decisão Oblíquas como Explicadores de Predições por Modelos de Aprendizado de Máquina

Dissertação de mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do Título de Mestre em Engenharia Elétrica.

Orientador: Frederico Gadelha Guimarães

Belo Horizonte - Minas Gerais
Julho, 2022

Santos, Samara Silva.
S237i Indução de árvores de decisão oblíquas como explicadores de predições por modelos de aprendizado de máquina [recurso eletrônico] / Samara Silva Santos. - 2022.
1 recurso online (147 f. : il., color.) : pdf.

Orientador: Frederico Gadelha Guimarães.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Inclui bibliografia.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Aprendizado do computador - Teses. 3. Árvores de decisão - Teses. 4. Inteligência artificial - Teses. I. Guimarães, Frederico Gadelha. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pelo bibliotecário Reginaldo César Vital dos Santos CRB/6 2165
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

FOLHA DE APROVAÇÃO

"INDUÇÃO DE ÁRVORES DE DECISÃO OBLÍQUAS COMO EXPLICADORES DE PREDIÇÕES POR MODELOS DE APRENDIZADO DE MÁQUINA"

SAMARA SILVA SANTOS

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

Aprovada em 14 de julho de 2022. Por:

Prof. Dr. Frederico Gadelha Guimarães
DEE (UFMG) - Orientador

Profa. Dra. Tatiane Nogueira Rios
DCC (UFBA)

Profa. Dra. Sandra Eliza Fontes de Avila
IC (UNICAMP)

Prof. Dr. Jaime Arturo Ramírez
DEE (UFMG)

Documento assinado eletronicamente por **Frederico Gadelha Guimaraes**, **Professor do Magistério Superior**, em 14/07/2022, às 15:45, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1580164** e o código CRC **F802ED76**.

Referência: Processo nº 23072.239399/2022-92 SEI nº 1580164

À Maria de Fátima Araújo Silva Santos

Agradecimentos

Com imensa alegria de poder contribuir de alguma forma para a ciência em uma área que eu tanto amo, posso dizer que "Até aqui o Senhor nos ajudou" (1 Samuel 7:12). Diante disso, agradeço e dedico ao meu Senhor e Salvador, Jesus Cristo, este trabalho. Ao meu querido esposo, Luamós Úride, pelo amor e constante apoio durante todo este trabalho, minha eterna gratidão. Sou extremamente grata e feliz por passar essa fase com você ao meu lado. Agradeço também aos meus pais, senhor Sebastião Domingos dos Santos e dona Maria de Fátima A. S. Santos, pelo constante amor e pelas incansáveis orações para que tudo ocorresse bem em todos os momentos, e por todo o esforço ao longo da vida para que meus sonhos e projetos se realizassem. Este título também é de vocês.

Ao meu querido orientador, Frederico Gadelha Guimarães, que não é somente um mentor e instrutor, mas uma pessoa inspiradora. De fato, fogem-me as palavras para descrevê-lo. Mas gostaria que soubesse o quanto sou grata por todo apoio, compreensão e instrução, pois sem eles, com certeza eu não chegaria até aqui. Agradeço a todos os professores que estiveram envolvidos nessa formação. Fico muito feliz ao perceber o quanto que aprendi nestes últimos dois anos, e vocês foram essenciais para isso. Agradeço aos colegas do MINDS, o laboratório de pesquisa incrível, onde os professores incentivam continuamente a cooperação entre os estudantes. Agradeço em especial à Tamires Rezende e ao Marcos Antônio Alves.

Agradeço às minhas amigas que tanto oraram, aconselharam e me ajudaram nos momentos de dificuldades, em especial a tia Vivi e Maria Cristina. Ao meu amigo, Lucas Thione, que esteve comigo desde o inicio deste processo, como colega de disciplina e amigo para a vida.

Agradeço ao CEFET-MG pela licença para capacitação concedida, a todos que estiveram envolvidos direta e indiretamente para que isso fosse possível, em especial, aos servidores Claiston Cosme Damião e Marcos Fernando. Por último, mas não menos importante, agradeço às agências de fomento CAPES, CNPq e FAPEMIG pelo constante apoio ao PPGEE-UFMG.

"Quand on se fait entendre, on parle toujours bien". Molière

Resumo

Os métodos de Aprendizado de Máquina (*Machine Learning* ou ML) têm sido largamente utilizados em diversas aplicações, devido ao alto poder de generalização e pela capacidade de obter relações complexas entre dados. Embora os sistemas consigam este feito, normalmente não existe uma relação clara do porque determinada decisão foi tomada, bem como no impacto da mudança dos atributos nas saídas geradas. A necessidade de compreender esses métodos torna-se ainda mais presente diante de leis que garantam o "direito à explicação", como previsto no artigo 20 da Lei Geral de Proteção de Dados (LGPD), e em outras regulamentações nesse sentido em todo o mundo. Em virtude disso, neste trabalho foi proposta a investigação quanto a indução de Árvores de Decisões Oblíquas - também conhecidas como Perceptron Decision Tree ou PDT - como método de interpretabilidade local para modelos de ML complexos. Uma vez que a PDT é transparente, pode-se utilizá-la para simular localmente o comportamento de modelos mais complexos e assim extrair informações sobre eles por meio dela. Tendo isso em vista, foi proposta a aproximação local das predições do método complexo a ser explicado, através da indução de PDTs, as quais têm os pesos evoluídos por meio de uma técnica heurística de otimização, baseada em computação evolucionária. Com a árvore evoluída, gera-se explicações sobre as decisões locais de modelos opacos, por meio do fornecimento das regras seguidas para obtenção das saídas, exposição da hierarquia de importância local dos atributos e limites de decisões associados a cada um deles. Foi apresentado também um novo modelo de PDT para problemas de regressão, o qual é utilizado para gerar explicações locais para este tipo de problema. A aplicação final gerada foi nomeada como *Perceptron Decision Tree Explainer* (ou *PDTX*), que em suma, é um método de interpretabilidade local agnóstico em relação ao modelo, que trabalha com dados tabulares estruturados, e que consegue fazer uma aproximação melhor do que alguns métodos clássicos da literatura, mantendo além da estabilidade das explicações geradas, a simplicidade delas. Adicionalmente, foi feito o estudo do efeito da aplicação de três técnicas de amostragem local em conjunto com o PDTX, concernente à estabilidade das explicações geradas, e da redução de dimensionalidade por cinco métodos de redução de atributos presentes na literatura, no impacto da qualidade da aproximação local. Os resultados obtidos são promissores: em comparação com o LIME (*Local Interpretable Model-Agnostic Explanations*) e Árvores de Decisões (DT), o PDTX obteve desempenho significativamente melhor para as métricas conhecidas como fidelidade e estabilidade, tanto no contexto de classificação, como no de regressão, e é comparável ao LIME em termos de simplicidade.

Palavras-chave: Inteligência Artificial Explicável, Interpretabilidade em IA, Inteligência Artificial, Aprendizado de Máquina, Árvores de Decisões Oblíquas.

Abstract

Machine Learning methods (*Machine Learning* or ML) have been widely used in several applications, due to the high power of generalization and the ability to obtain complex relationships between data. Although systems achieve this feat, there is usually no clear relationship as to why a particular decision was made, as well as the impact of changing attributes on the generated outputs. The need to understand these methods becomes even more present in the face of laws that guarantee the "right to explanation", as provided for in article 20 of the General Data Protection Law (LGPD), and in other regulations in this sense throughout the world. As a result, this work proposes to investigate the induction of Oblique Decision Trees - also known as Perceptron Decision Tree or PDT - as a method of local interpretability for complex ML models. Since the PDT is transparent, it can be used to locally simulate the behavior of more complex models and thus extract information about them through it. With this in mind, a local approximation of the predictions of the complex method to be explained was proposed, through the induction of PDTs, whose weights evolved through a heuristic optimization technique, based on evolutionary computation. With the grown tree, explanations about the local decisions of opaque models are generated, by providing the rules followed to obtain the outputs, exposing the hierarchy of local importance of the attributes and decision limits associated with each one of them. A new PDT model for regression problems was also presented, which is used to generate local explanations for this type of problem. The final application generated was named *Perceptron Decision Tree Explainer* (or *PDTX*), which, in short, is a model-agnostic local interpretability method, which works with structured tabular data, and which can make a better approximation than some classical methods in the literature, maintaining, in addition to the stability of the generated explanations, their simplicity. Additionally, a study was made on the effect of applying three local sampling techniques together with PDTX, concerning the stability of the generated explanations, and the reduction of dimensionality by five methods of reduction of attributes present in the literature, on the impact of the quality of the local approach. The results obtained are promising: compared to LIME (*Local Interpretable Model-Agnostic Explanations*) and Decision Trees (DT), PDTX performed significantly better for known metrics such as fidelity and stability, both in the context of classification, as in regression, and is comparable to LIME in terms of simplicity. *Keywords: Explainable AI, AI Interpretable, Artificial Intelligence, Machine Learning, Oblique Decision Trees.*

Lista de Ilustrações

Lista de Tabelas

Lista de Algoritmos

Sumário

Capítulo 1

Introdução

O início do século XXI foi marcado pela ascensão da capacidade de processamento computacional e pelo vultoso volume de dados gerados através do uso intensivo de ferramentas digitais. Esse potencial ocasionou a Quarta Revolução Industrial, também chamada de Indústria 4.0, que introduziu conceitos como *Big Data*, Internet das Coisas, além de larga utilização de soluções de Inteligência Artificial (IA) na pluralidade de áreas do conhecimento. Modelos baseados em *Machine Learning* (ML, Aprendizado de Máquina) e *Deep Learning* (DL, Aprendizagem Profunda) têm alcançado resultados promissores, muitos dos quais inimagináveis em décadas passadas (Došilović et al., 2018).

Embora seja uma abordagem relativamente nova, com principais contribuições a partir de 2012, a partir do trabalho de Krizhevsky et al. (2012), a DL tem apresentado bons retornos em diversos campos, como reconhecimento de fala e de imagens, tradução de linguagem, dentre outras aplicações de IA. Todavia, assim como em alguns modelos de ML tradicionais, a falta de transparência ainda é o principal problema atrelado ao uso dessa metodologia, que carece de ser resolvido, conforme apontado por Marcus (2018).

De acordo com Gunning (2017), ao contrário das soluções transparentes, onde se é possível obter uma clara relação entre entrada e saída, os modelos de ML não são intuitivos, e por isso são considerados complexos. Por causa disso, a busca por um equilíbrio entre a performance e a explicabilidade se faz necessária, visto que possibilita ao usuário fazer deliberações baseadas nas explicações fornecidas pelo sistema.

Segundo o autor, em sistemas caixa-preta (*black-box models*) não existe uma clareza do porquê de uma decisão e não outra, ou em que situações há sucesso e quando há falhas, ou quando é possível confiar no sistema e o que deve ser feito para corrigir um erro. As expectativas são que, futuramente, com maior transparência, haja o entendimento para essas questões e uma compreensão mais completa do sistema, que traga mais confiabilidade. Nesse contexto, a Inteligência Artificial Explicável (XAI, *Explainable Artificial Intelligence*) objetiva a criação de técnicas que provejam explicabilidade aos modelos de ML e DL, capacitando os seres humanos a gerenciá-los e a agirem com maior corretude.

No que diz respeito à utilização de modelos opacos, corre-se o risco de que as soluções desrespeitem legislações, entrem em conflito com valores éticos, ou que produzam efeitos que sejam prejudiciais, dentro do contexto de utilização. Com isso, existe uma crescente demanda por modelos explicáveis, para que as decisões produzidas pelo sistema sejam corretamente avaliadas por profissionais que consigam dimensionar o impacto da escolha dessas soluções.

Além disso, modelos compreensíveis ajudam as empresas a criarem produtos seguros e mais confiáveis e a entender a responsabilidade decorrente do emprego dessas tecnologias. Eles propiciam às partes envolvidas compreenderem essas explicações, entenderem como as decisões são tomadas pelo sistema automatizado, seus pontos fortes e fracos, e como isso pode ajudar em decisões melhores no futuro. Nesse sentido, a imposição de que tanto a IA como a ML sejam compreensíveis relaciona-se com a validação de qualidade e corretude no que tange às respostas desses sistemas, fator que é especialmente importante para que tais saídas estejam alinhadas com valores humanos. Ademais, em áreas como medicina, biologia e ciências econômicas, essa importância é requisitada não somente para aceitação desses resultados, como também para progresso científico dessas áreas (Pedreschi et al., 2019).

À vista disso, essa temática tem uma relevância fundamental dentro da IA, pois espera-se que desenvolvedores e usuários consigam compreender as justificativas das ações ou resultados dessas metodologias, bem como que tais explicações exprimam a inteligenciabilidade do processo. Para tal, carecem do uso de representações comunicáveis por meio de recursos matemáticos, lógicos, linguísticos ou visuais (Preece, 2018).

A literatura apresenta uma grande discussão nesse sentido, sendo que em alguns trabalhos são explanadas as principais diretrizes que conduzem as discussões a respeito, como citado na sequência. Arrieta et al. (2020) introduz o conceito de Inteligência Artificial Responsável (*Responsible Artificial Inteligence*), enumera e diferencia conceitos relacionados aos métodos de explicabilidade de IA (XAI) existentes na literatura, como explicabilidade, interpretabilidade, inteligibilidade, compreensibilidade e transparência. Também são sintetizadas as perguntas que direcionam as discussões em relação a este tema (o quê, por quê, para quê, e como), além de examinarem diferentes níveis de transparência tanto dos modelos de IA, quanto de abordagens de explicabilidade. Os autores apresentam modelos que são intuitivamente transparentes, como: Regressão linear, árvores de decisão, K-ésimo Vizinho mais Próximo, aprendizado baseado em regras, dentre outros, e técnicas *post-hoc* para explicabilidade dos modelos não transparentes. Ao final, o trabalho traz uma taxonomia, cujo intuito é servir como material de referência àqueles que objetivam adotar a XAI, além de estimular avanços nesse campo.

Já no trabalho de Adadi e Berrada (2018) é fornecida uma visão geral dos princípios, razões e implicações em se obter elucidação a partir dos sistemas inteligentes.

São ressaltadas as necessidades da XAI no sentido de justificar os resultados, permitir o controle, prevenir erros e permitir a agilidade nas ações de correção deles, quando ocorrerem, bem como promover constantes melhorias do sistema.

Em Miller (2019) é estabelecida a conexão entre as discussões presentes nas ciências sociais e a explicabilidade na IA, onde três conclusões principais foram destacadas pelos autores: as explicações são contrafactuais, pois os humanos tendem a entender porque um determinado evento aconteceu ao invés de alguns outros eventos. Além disso, as explicações são seletivas e se concentram em uma ou duas causas possíveis – ao invés de todas as causas possíveis – para uma decisão ou recomendação; e por fim, as explicações são uma conversa e interação social com o propósito de transferir conhecimento, implicando que o explicador deve ser capaz de alavancar o modelo mental do usuário enquanto se envolve no processo de explicação (Confalonieri et al., 2021).

Embora muitas técnicas de explicações tenham sido propostas, conforme apresentado, uma das soluções mais citadas na atualidade para este propósito é o LIME, que é um explicador local, que pode ser utilizado para qualquer método ML caixa-preta, e que gera explicações com base em um modelo globalmente linear. Este explicador tem como proposta simular o comportamento local dos métodos opacos utilizando pra isso um modelo mais simples – linear. Todavia, essa abordagem possui algumas desvantagens comumente citadas na literatura, como baixa capacidade de reprodução local (fidelidade) (Messalas et al., 2019; Tian e Liu, 2020; Jiang, 2022), além da produção de resultados instáveis (Zafar e Khan, 2019; Visani et al., 2020a; Zhou et al., 2021). Alguns trabalhos evidenciam que há um tipo especial de árvore de decisão, com partições oblíquas, que é transparente, e normalmente produz bons resultados quando aplicada à solução de problemas de ML de classificação. Dessa forma, entende-se que quando utilizada para reprodução local do comportamento de um método mais complexo, pode gerar resultados mais promissores que métodos globalmente lineares ou árvores de decisões comuns. Sabe-se que o desempenho deste método depende da técnica de indução utilizada, e que muitas alternativas são apresentadas na literatura neste propósito (Bot e Langdon, 2000; Vukobratović e Struharik, 2015; Lopes et al., 2012). Uma possibilidade, é com a utilização o algoritmo evolucionário de Evolução Diferencial (Storn e Price, 1997), que é uma técnica de otimização baseada em população, que desde que foi proposto apresentou resultados interessantes para a solução de problemas contínuos, e posteriormente para problemas combinatórios e com múltiplos objetivos (Prado et al., 2010; Liang et al., 2019). Assim, a seleção de um método de indução que possua características que justifiquem sua aplicação nessa tarefa, também constitui um dos objetivos deste trabalho. Uma vez que a Evolução Diferencial mostrou-se interessante em (Lopes et al., 2012), delimita-se o escopo a partir de opções que evoluem este algoritmo de otimização. Dessa forma, partindo-se do princípio que a árvore utilizada consiga reproduzir com boa fidelidade o método caixa-preta em estudo, pode-se utilizá-la para gerar explicações por meio dela.

1.1 Motivação

Assim como existe um padrão social que considera que determinadas ações são corretas ou incorretas, de acordo com um conjunto de princípios sociais e éticos explícitos ou implícitos, também há uma preocupação com que agentes autônomos comportem-se de maneira alinhada a esses valores. À vista disso, para que as escolhas desses sistemas sejam aceitas, é fundamental que as decisões tomadas por eles sejam adequadas com preceitos e normas humanas, para isso é preciso investigar como restringir as ações em um sistema de IA dentro dos limites que estes softwares devem operar (Noothigattu et al., 2018).

A partir do momento em que as soluções são elucidadas, torna-se possível mensurar os impactos positivos e negativos dentro do campo de aplicabilidade, bem como atribuir a devida responsabilização pelo seu uso e decorrências. A utilização de ML sem a devida compreensão pode levar a injustiças, ao desrespeito à privacidade, à manipulação de opiniões, além de poder causar grandes perdas, insegurança e desrespeito ao usuário. A explicabilidade ajuda os especialistas no controle sobre como e quando aplicar tais soluções, e suportam os governos na definição de legislações que envolvam a regulamentação quanto ao uso, manipulação de dados e aplicabilidades de soluções baseadas em IA.

1.1.1 Regulamentação da IA no mundo

Outro fator que reforça a importância da XAI é que a partir da segunda década do século XXI cresceu a preocupação governamental no que tange à proteção de dados e quanto ao uso de softwares de IA, que atuam direta ou indiretamente na tomada ou na influência de decisões. Com isso, diversos países lançaram regulamentações que limitam ou condicionam a utilização dessas tecnologias.

Em 2015 a Organização das Nações Unidas (ONU) criou, por meio do UNICRI (*United Nations Interregional Crime and Justice Research Institute*, ou Instituto Inter-regional de Pesquisas das Nações Unidas para o crime e a Justiça – em tradução livre), um centro de IA e Robótica no intuito de compreender tantos os benefícios quanto os riscos decorrentes da IA e da Robótica, e por extensão, contribuir para empreender políticas para prevenção de crimes. No mesmo ano ocorreu o evento: *"Rising to the Challenges of International Security and the Emergence of Artificial Intelligence"* (Ahmad e Gesley, 2019).

No ano seguinte, a União Europeia concebeu o Regulamento Geral de Proteção de Dados que, além de tratar tópicos que dizem respeito à coleta e armazenamento de dados pessoais, traz também no artigo 22 proibições a respeito do uso de softwares de tomadas de decisões, os quais relacionam-se propriamente com sistemas que utilizam recursos de aprendizagem de máquina (Goodman e Flaxman, 2017).

Já em 2018, o Brasil promulgou a Lei de Proteção de Dados Pessoais (Lei

13.709/2018) que trata da proteção e privacidade de dados pessoais dos cidadãos brasileiros, que entrou em vigor em 14 de agosto de 2020. E em 2020 foi criado o Projeto de Lei (PL) 21/2020, o qual correlaciona-se a outros três PL (5051/2019, 5691/2019 e 872/2021), todos de autoria do Senado brasileiro, e que buscam regulamentar especificamente o uso da IA no Brasil. Segundo o autor deste primeiro, o projeto objetiva estimular o desenvolvimento da IA, e proteger os cidadãos do mau uso dela (Júnior, 2020). Dentre os fundamentos de disciplinar o uso da IA no Brasil, é mencionado no art. 2º do PL 5051 parágrafos IV e V "a transparência, a confiabilidade e a possibilidade de auditoria dos sistemas e a supervisão humana". Esse PL determina ainda que "a responsabilidade civil por danos decorrentes da utilização desse tipo de sistema será de seu supervisor". Além disso, o PL 872 art. 4º inciso VI determina que as soluções baseadas em IA deverão "prover decisões rastreáveis e sem viés discriminatório ou preconceituoso", o que reforça a necessidade do entendimento das decisões internas para garantir que o funcionamento encontra-se não só de acordo com o esperado, mas sem a reprodução de viéses e preconceitos sociais muitas vezes existentes nos dados utilizados para modelagem da IA. De forma semelhante, no PL 5691, no art. 4º inciso IV também menciona que tais soluções devem "ser inteligíveis, justificáveis e acessíveis".

Devido ao interesse coletivo em saber como são beneficiados (ou prejudicados) pela IA e de como os dados estão sendo usados, essa pauta tem sido cada vez mais demandada. Apesar disso, o controle pode ser diferente de um país para outro e isso interfere diretamente nas empresas que fazem uso desse ramo. Isto posto, o desenvolvimento de ferramentas que provêm maior transparência e inteligibilidade em sistemas autônomos são fundamentais para continuidade da evolução dessa área do conhecimento e para proteção dos usuários.

1.2 Objetivos gerais

Tendo em vista o apresentado, o objetivo geral deste trabalho é o desenvolvimento de um explicador local, que seja baseado em árvores de decisão com hiperplanos lineares (chamadas de *Perceptron Decision Tree* ou PDT) para geração de interpretabilidade de modelos de ML de classificação e regressão caixa-preta, que utilizam dados estruturados. A aplicação resultante deve gerar explicações locais com maior completude que modelos populares da literatura, por meio das regras de decisão, da apresentação da hierarquia de importância dos preditores e dos limites de decisão de cada característica.

1.2.1 Objetivos específicos

1. Fazer a revisão bibliográfica de abordagens para interpretabilidade local de métodos de ML. Compreender as características gerais dos principais explicadores, as vantagens

e limitações de cada um deles. A partir disso, fazer o levantamento do estado da arte e das demandas atuais da XAI;

2. Propor a evolução dos pesos de uma PDT com o uso de técnica de otimização que tenha histórico de desempenho satisfatório aplicados a problemas contínuos multidimencionais;

3. Propor um novo método de ML baseado em PDT para problemas de regressão, com evolução paramétrica semelhante ao que ocorre nas PDTs para problemas de classificação;

4. Propor um novo explicador local para ML, com explicações obtidas a partir das PDTs para classificação e regressão aqui apresentadas;

5. Detalhar o processo de extração de explicações a partir de uma PDT;

6. Avaliar a estabilidade local de três tipos de amostragem local: Aleatória a partir de uma distribuição gaussiana, amostragem por Hipercubo Latino e Amostragem a partir da sequência Hammersley;

7. Avaliar estabilidade local na presença de aplicação de métodos de seleção de características;

8. Comparar a fidelidade local, estabilidade e comprimento de regras quantitativamente com algoritmos da literatura;

9. Comparar qualitativamente o PDTX com *baselines* da literatura.

1.3 Contribuições

A partir deste trabalho foi publicado e apresentado no XV Congresso Brasileiro de Inteligência Computacional o artigo "*PDTX: A novel local explainer based on the Perceptron Decision Tree*" (Santos et al., 2021). Este trabalho aborda como o PDTX extrai explicações locais para métodos ML para problemas de classificação que utilizam dados tabulares. Uma vez que este método possui uma capacidade de aproximação local muito melhor do que a de alguns métodos conhecidos (LIME e DT), as explicações fornecidas por ele possuem maior confiabilidade que outros estudados. Além disso, o método proposto combina as vantagens dos dois métodos comparados, pois além de fornecer a hierarquia de importância entre os atributos e os limites de decisão locais, semelhante ao LIME, provê também as regras de decisões utilizadas para obtenção das classificações, como ocorre com a DT, e uma visualização geral de como ocorre a classificação em regiões próximas.

Este trabalho foi expandido para problemas de regressão em um artigo que encontra-se em produção. Um novo método de ML interpretável baseado em PDT foi

proposto para este tipo de problema, com estrutura interna semelhante à árvore de classificação, mas que aplica regressões lineares múltiplas nos nós folhas. As aproximações locais realizadas por este método foram comparadas com aquelas realizadas por outros explicadores em termo de fidelidade local e também referente à estabilidade. Compara-se ainda em termos qualitativos as explicações fornecidas. Os resultados mostraram superioridade em termos de fidelidade e estabilidade quando comparados com o LIME, e superioridade em termo de fidelidade quando comparado com a DT.

Considerando-se que uma das motivações que justificam a XAI é a necessidade de confiabilidade, rastreabilidade e dentre outros requisitos em áreas críticas, dentre elas a área da saúde, também encontra-se em produção um artigo contemplando o "Estudo da interpretabilidade de métodos de aprendizado de máquina no diagnóstico de doenças: estudo de caso sobre câncer de mama, doenças cardiovasculares e diabetes". Nesse trabalho são feitas avaliações das explicações fornecidas pelo PDTX, LIME e DT no contexto da classificação e regressão para problemas comuns na área da saúde.

Além de dados tabulares, um dos problemas comuns a ser estudado a aplicação de métodos de interpretabilidade são redes profundas que utilizam reconhecimento de padrão para imagens. Nesse contexto, foi proposta a comparação do LIME, na versão padrão, com outra versão onde utiliza-se uma DT para substituir o método linear interno a este método, e tal versão foi aplicada para gerar explicações para o diagnóstico de câncer de pele. Este trabalho foi enviado ao Congresso Brasileiro de Automática (CBA) com o nome *"Interpretable Diagnosis of Skin Cancer Using Deep Learning"*, e mostra que a estabilidade do processo de geração de explicação local é melhorada quando a DT é utilizada para mensurar as regiões das imagens mais importantes para a predição.

Por fim, este mestrado gerou também a publicação de um capítulo de livro chamado "Redes Neurais Artificiais: Uma visão histórica", publicado pela editora Poisson, no livro "Engenharia, Gestão e Inovação - Volume 1" (Freitas et al., 2022).

1.4 Organização dos capítulos

No Capítulo 2, discorre-se sobre o conceito de Explicabilidade em Inteligência Artificial, onde é apresentado o chamado "problema da caixa-preta", bem como os principais conceitos e terminologias associados a esta área. Denota-se os modelos que são naturalmente transparentes, apresenta-se algumas técnicas para geração de interpretabilidade para modelos caixa-preta, além do levantamento do estado da arte do tema e da exposição do problema de pesquisa tratado neste trabalho. Na sequência, no Capítulo 3 é tratado em detalhes o processo de otimização realizado pelo método utilizado neste trabalho para indução das PDTs. Nesse contexto, foi feito o histórico memorial de versões intermediárias que conecta o algoritmo básico de Evolução Diferencial, a versão conhecida como jSO,

utilizada neste trabalho, ressaltando cada melhoria que foi feita por metodologia; e assim, justifica-se a motivação de adotação da heurística em questão. Já no Capítulo 4 a metodologia proposta é retratada, e o processo de evolução e extração de explicações por meio de uma PDT é descrito. Na sequência, os Resultados e Discussões são exibidos no Capítulo 5. Finalmente, as Considerações Finais e Conclusões são apresentadas no Capítulo 6 .

Capítulo 2

Explicabilidade em Inteligência Artificial

Embora o termo XAI seja relativamente novo, introduzido em Van Lent et al. (2004), a demanda por explicabilidade é antiga, datada na década de 80 por Moore e Swartout (1988) e tem obtido cada vez mais relevância à medida que a alta performance obtida por muitos métodos de aprendizado recentes contrasta com a falta de transparência destes modelos. Com isso, a XAI lida com a implementação de transparência e rastreabilidade de métodos estatísticos de aprendizado de máquina de caixa preta, especialmente os de DL (Holzinger et al., 2019), visa tornar os resultados dos sistemas de IA mais compreensíveis para os humanos, e objetiva capacitar usuários *experts* ou não a compreenderem profundamente e com nível adequado de confiança os métodos de aprendizado automático e permitir a adoção dessas tecnologias mais amplamente (Clinciu e Hastie, 2019).

Os modelos de ML classificados como caixa-branca são inerentemente interpretáveis, por exemplo as árvores de decisões ou modelos de regressão (Doran et al., 2017). Nesses casos, o entendimento prévio sobre o funcionamento destas técnicas possibilitam a compreensão de cada um dos resultados alcançados por elas, pois são transparentes. Da mesma forma, caso haja mudanças na entrada, é possível rastrear o impacto gerado na saída. Todavia, os algoritmos de ML caixa-preta costumam obter desempenho de aprendizado melhores do que os interpretáveis, todavia sem fornecerem explicações em relação a como as saídas foram obtidas. Em virtude disso, são necessárias estratégias para elucidar as decisões tomadas por eles.

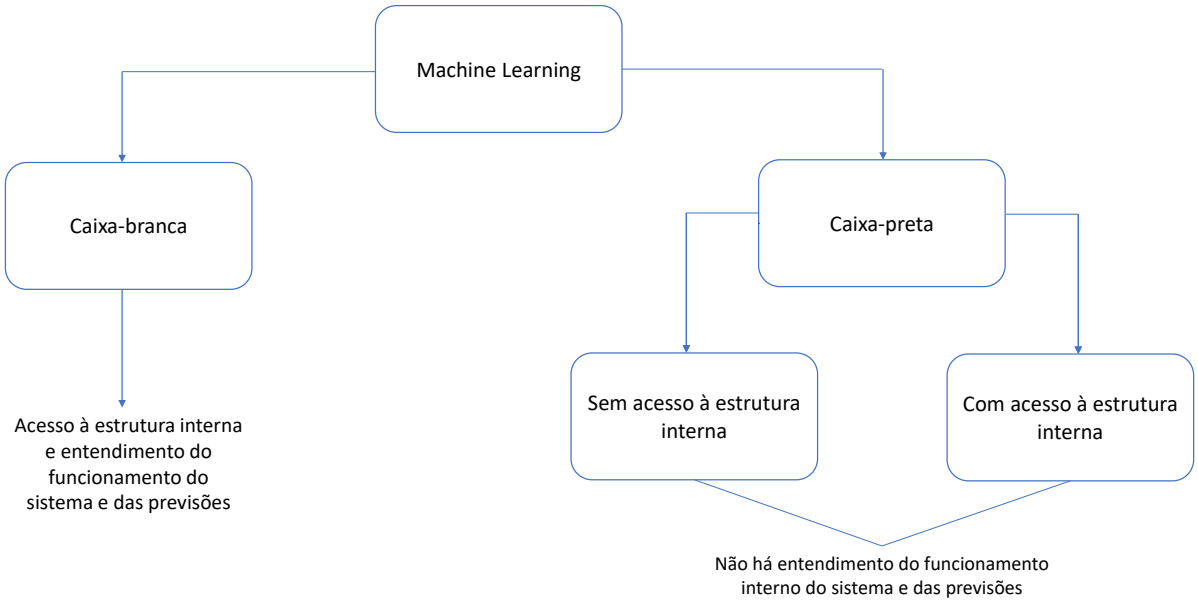
Levando isso em conta, a XAI visa a geração de modelos explicáveis que mantenham esse alto desempenho de aprendizado, mas que ao mesmo tempo permita que usuários entendam, confiem e possam gerenciar essas aplicações de IA (Gunning, 2017).

2.1 O problema da caixa-preta

Um sistema pode ser considerado opaco quando não existe um mapeamento claro entre as entradas e saídas, ou explicações que elucidem como e porque as predições foram feitas. Esse tipo de sistema engloba tanto as IA proprietárias de código fonte fechado, como também outras soluções classificadas como "caixa-preta". Por outro lado, nos sistemas interpretáveis, as soluções podem ser vistas, estudadas e compreendidas, pois existe uma correlação matemática entre entrada e saída, que fundamenta a transparência desses modelos.

Há métodos que também podem ser considerados caixa-preta mesmo quando as informações internas estão disponíveis. Nesse caso, eles são considerados opacos no sentido de que é difícil saber exatamente como estão programados ou como as decisões são tomadas. Tendo em vista que os desenvolvedores exercem uma influência relativamente limitada na configuração dos valores dos parâmetros, pode não haver uma clareza de como esses parâmetros contribuem para o comportamento do sistema. Além disso, mesmo quando os valores de parâmetros individuais são conhecidos, o fato de que eles podem interagir de forma não linear, bem como de forma recorrente, significa que é quase impossível entender, prever ou intervir sistematicamente na maneira como uma entrada específica é transformada de modo a gerar uma saída particular (Zednik, 2021). A Figura 1 apresenta um esquema simplificado do problema mencionado.

Figura 1 – O problema da caixa-preta



Fonte: Os autores

Um exemplo disso ocorre com a AlexNet (Krizhevsky et al., 2012), vencedora da

competição "Desafio de Reconhecimento Visual em Grande Escala ImageNet" em 2012, que possui 650 mil neurônios e 60 milhões parâmetros. Embora os pesos dos neurônios estejam acessíveis, a rede é altamente complexa, e portanto, não é possível inferir qualquer comportamento olhando unicamente para essas informações.

Para Pedreschi et al. (2019) o problema da caixa-preta pode ser tratado de duas maneiras. A primeira seria utilizando Explicação como Padrão (*eXplanation by Design*), que como o nome sugere, trata-se de construir soluções que já produzem explicações nas tomadas de decisões; enquanto a segunda, chamada de Explicação Caixa Preta (*Black Box eXplanation*), tem a ver com produzir a explicação a partir do sistema caixa-preta já pronto. Considerando essa e outras referências, este trabalho traz neste capítulo o estado da arte do assunto, abordando especialmente os principais conceitos e soluções já propostas que contribuíram com o tema. Sendo assim, a seção 2.2 aborda os conceitos e terminologias da XAI, na sequência, em 2.3 é tratado a importância da temática, com base nos principais artigos de *survey* da área. Em 2.4 é exposto a taxonomia da área e ao final, em 2.5 encontra-se um breve resumo com os principais exemplos de métodos de interpretabilidade/explicabilidade da atualidade.

2.2 Conceitos e Terminologias

Apesar de alguns dos dicionários de língua inglesa não possuírem o conceito dos termos equivalentes a explicabilidade ou explicável (Rosenfeld e Richardson, 2019), (Doran et al., 2017), existe a definição do termo *explanation* (explicação), que segundo o Oxford English Dictionary significa, em tradução livre,

> *Uma declaração ou conta que deixa algo claro; uma razão ou justificativa dada para uma ação ou crença.*

A língua portuguesa contém a definição para o vocábulo explicabilidade:

> *Característica do que é explicável; qualidade daquilo que se consegue explicar* (RIBEIRO e NEVES, 2022).

Outros termos comumente trabalhados na literatura, neste âmbito, são o de interpretabilidade e transparência. Enquanto alguns autores os consideram sinônimos (Rosenfeld e Richardson, 2019), outros os diferenciam em seus trabalhos. Para Arrieta et al. (2020), por exemplo, explicabilidade, em sentido técnico, pode ser vista como uma característica ativa de um modelo, que tem a capacidade de prover entendimento ou esclarecer as ações tomadas por ele, ou de detalhar o seu funcionamento interno. Já a interpretabilidade refere-se a capacidade de prover entendimento a um observador humano, podendo ser tida como um sinônimo de transparência, segundo os autores.

No trabalho de Roscher et al. (2020) onde aborda-se a explicabilidade de métodos de ML no contexto das ciências naturais, o fluxograma de processos baseado em ML vai desde a entrada de dados até a saída de informação científica, e neste intervalo relaciona-se com conceitos como transparência, interpretabilidade e explicabilidade, os quais também se associam com o conhecimento do domínio de aplicação. Neste contexto, a transparência tem a ver com a abordagem de ML, enquanto que a interpretabilidade utiliza-se do modelo de ML junto aos dados, e a explicabilidade considera o modelo, os dados e o envolvimento humano. Em termos de explicabilidade, os autores ressaltam que o conceito vai além do de interpretabilidade, pois são necessárias informações adicionais contextuais, a partir do domínio do conhecimento, para que se consiga explicar as decisões. Assim, para obter um resultado científico, o cientista usa os dados, a transparência do método e sua interpretação para explicar os resultados fazendo uso do conhecimento de domínio.

Já para Rosenfeld e Richardson (2019), a explicabilidade pode ser tida como um resultado da interpretabilidade em conjunto com seis métodos: Transparência, análise de características, análise de protótipo, ferramenta de modelo, ferramentas de saída e ferramentas de visualização, que os autores representaram a inter-relação entre eles por meio de um diagrama de Venn. Assim, na visão deles, interpretabilidade pode ser entendida como um meio de prover explicabilidade.

Enquanto para Holzinger et al. (2019), a explicabilidade, no contexto da IA, considera as partes ativas em determinados modelos algorítmicos, que contribuem para certas decisões, ou para a precisão do modelo, em cima dos dados utilizados, sem incluir de forma explícita um modelo humano. Além disso, os autores apresentam também o conceito de causalidade, que eles definem como uma extensão da explicabilidade já definida, mas que considere esse aspecto humano, trazendo uma compreensão causal das decisões, respondendo não somente como, mas também o porquê delas.

No que diz respeito aos sistemas compreensíveis, são aqueles que emitem símbolos junto às saídas, que permitem ao usuário elaborar, por conta própria, uma explicação, baseado na experiência pessoal (Doran et al., 2017).

Além dessas, outras nomenclaturas e definições são trabalhadas na bibliografia, como Compreensibilidade (Weitz et al., 2019), (Gleicher, 2016), (Craven, 1996), Transparência (Chiticariu et al., 2015), (Wood, 2018), Equidade (Amann et al., 2020), dentre outras.

Considerando as diferentes definições presentes na literatura, neste trabalho usa-se aquelas compiladas na Tabela 1, consoante os trabalhos de Arrieta et al. (2020) e Adadi e Berrada (2018). Nesse trabalho, os termos "explicabilidade" e "interpretabilidade" foram tratados como sinônimos.

Tabela 1 – Terminologias XAI

Nomenclatura	**Definição**
Explicabilidade	está relacionada com a noção de explicação como uma interface entre os humanos e um tomador de decisão. É tanto um proxy preciso do tomador de decisão quanto compreensível para os humanos.
Interpretabilidade	é ser capaz de explicar ou fornecer significado de uma forma compreensível para um ser humano. Ou ainda, o grau em que uma pessoa pode entender o motivo de um resultado.
Entendibilidade	refere-se ao conhecimento humano das funções do sistema sem explicações sobre suas funcionalidades internas.
Compreensibilidade	é a habilidade de um sistema de aprendizagem de mostrar seu conhecimento aprendido de uma forma inteligível para o ser humano.
Transparência	um modelo é transparente se for auto-compreensível. Diz respeito à necessidade de descrever, inspecionar e reproduzir os mecanismos pelos quais os sistemas de IA tomam decisões e aprendem a se ajustar ao seu ambiente e à governança dos dados utilizados criados.
Complexidade	é o nível de esforço exigido por um usuário para entender uma explicação levando em consideração o treinamento do usuário ou quaisquer restrições de tempo necessárias para a compreensão.
IA Responsável	é a IA que considera os valores sociais e as preocupações morais e éticas.

2.3 Razões para Explicabilidade

Existem muitas razões que justificam essa demanda por softwares explicáveis. Algumas são apresentadas a seguir:

2.3.1 Impacto da falha

A importância de se compreender as decisões de algoritmos de ML está, em certo nível, relacionada com o impacto que as falhas desses podem causar. Sejam as seguintes situações:

1. Um sistema de suporte diagnóstico de doenças, baseado em IA, indica a utilização de uma determinada medicação para um paciente, com base no conjunto de sintomas que ele apresenta;

2. Um software que faz recomendação de livros para os usuários, e para isso utiliza aprendizagem de máquina, tendo como base o histórico de leitura desses;

3. Um algoritmo de previsão de valorização/desvalorização de investimentos, que sugere compras ou vendas de títulos.

Na primeira situação, o erro do sistema pode levar ao tratamento incorreto para determinada doença, gerar piora do quadro médico, além de possibilitar a ocorrência de efeitos adversos, pelo uso de medicação imprópria. Sendo assim, o erro desse sistema pode causar consequências significativas.

Já na segunda, a consequência de uma recomendação incorreta não é proporcional à anterior, pois a falha não oferece malefício significante ao usuário. E finalmente, na terceira, uma indicação incorreta de venda e/ou compra de títulos pode levar a altos prejuízos, sendo significante para quem faz uso desse tipo de ferramenta. Dito isso, em situações em que o impacto é elevado, é importante que o sistema seja confiável àqueles que o utilizam. Se o sistema fornece sinais que permitem a um especialista avaliar os motivos pelos quais o levou à determinada previsão, ele é capaz de estudá-la em conjunto com esses dados e aceitar ou rejeitar a decisão entendendo como ela foi tomada.

Por causa do impacto negativo que a adoção de métodos de IA em áreas críticas, várias aplicações de XAI foram propostas nos últimos anos. Pode-se citar o diagnóstico automatizado de falhas em manutenção de aviação (Zeldam, 2018), diagnóstico de doenças (Kavya et al., 2021; Deperlioglu et al., 2022; Biswas et al., 2021), na indústria (Kotriwala et al., 2021), em sistemas militares (Svenmarck et al., 2018), etc.

2.3.2 Aceitação social

A importância de explicações se relaciona também com a aceitação social da utilização desse tipo de ferramenta. É comum existir resistência na adoção de tecnologias quando não há disponível o entendimento completo de como elas funcionam, e como impactam os usuários que as utilizam.

No geral, os sistemas são considerados interpretáveis quando são entendíveis por humanos e é possível interpretar o processo realizado para a tomada de decisão (Joshi et al., 2021). Ou seja, usando as explicações, é possível compreender como o sistema chegou a uma decisão específica. A interpretabilidade ajuda a garantir a imparcialidade no processo de tomada de decisão, a robustez do modelo, e também esclarece as variáveis relevantes que afetaram os resultados (Arrieta et al., 2020).

2.3.3 Detecção de vieses

Em geral, os modelos de ML aprendem vieses presentes nos dados de treinamento, o que pode fazê-los reproduzir preconceitos sociais existentes. À vista disso, a interpretabilidade ajuda na depuração do funcionamento desses sistemas, na identificação de

tendências, e a reformular tais sistemas para que trabalhem de maneira ética, compatível com preceitos sociais.

Um dos problemas exemplificado por Molnar (2019) é o viés na aprovação ou rejeição automática de solicitações de crédito, que pode discriminar uma minoria que tenha sido historicamente privada de seus direitos. Para o autor, ainda que o objetivo principal do sistema seja conceder empréstimos apenas a pessoas que irão eventualmente reembolsá-los, a incompletude da formulação deste reside no fato de que deve-se não apenas minimizar a inadimplência de empréstimos, mas também não discriminar pessoas com base em certos dados demográficos. Esta é uma restrição adicional que, conforme argumentado, deve fazer parte da formulação do problema (concessão de empréstimos de forma compatível e de baixo risco) que, naturalmente, não é coberta pela função de perda para a qual o modelo de aprendizado de máquina foi otimizado.

2.3.4 Medidas de segurança

Existem modelos ML que realizam tarefas do mundo real que exigem medidas de segurança e testes, pois o funcionamento incorreto pode levar a acidentes ou incidentes. Um exemplo disso são os carros autônomos, onde ferramentas de AI, especialmente SML (*Statistical Machine Learning*), é empregada em funções críticas de segurança. Todavia, muitos casos que levam a acidentes cruciais devido a um controle não intencional causado por erro de julgamento do SML foram relatados (Morikawa e Matsubara, 2020). As explicações do funcionamento interno desses sistemas, podem ajudar a prevenir e corrigir falhas detectadas, por permitir entender o que as geraram.

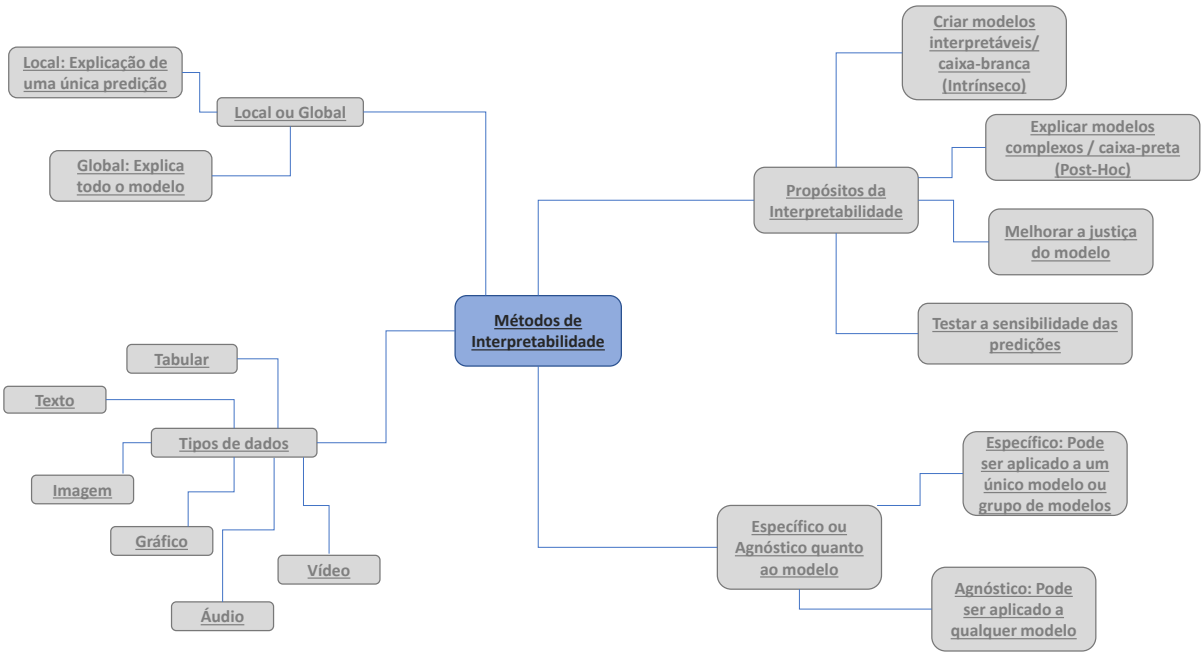
2.3.5 Sistema depurável e auditável

Um sistema só pode ser depurado quando pode ser entendido. A explicabilidade não é importante apenas para justificar decisões, mas para ajudar a evitar que fatos negativos ocorram, pois entender mais sobre o comportamento do sistema fornece maior visibilidade sobre vulnerabilidades e falhas desconhecidas e ajuda a identificar e corrigir erros rapidamente em situações de baixa criticidade (depuração) (Adadi e Berrada, 2018).

2.4 Taxonomia dos métodos de interpretabilidade

Os métodos de interpretabilidade podem ser classificados em diferentes categorias, a saber: Em relação ao tipo de técnica para prover interpretabilidade, podendo ser Intrínseco ou *post-hoc*. Específico ou agnóstico em relação ao modelo a ser explicado, e quanto ao escopo, pode ser local e/ou global. Nas próximas subseções será tratado cada uma dessas classificações. A Figura 2 sumariza essas classificações.

Figura 2 – Métodos de Interpretabilidade



Fonte: Linardatos et al. (2020)- Adaptado

2.4.1 Intrínseco ou post-hoc

Existem duas classes principais de técnicas que fornecem interpretabilidade no aprendizado de máquina, a saber interpretabilidade intrínseca e interpretabilidade *post-hoc* (Pintelas et al., 2020; Molnar, 2019). A primeira refere-se a modelos que são naturalmente interpretáveis (caixa-branca), enquanto as segundas visam explicar e interpretar as previsões de cada modelo, sem acessar a estrutura interna dele, como os pesos. Para essa tarefa, essas técnicas frequentemente desenvolvem um segundo modelo que tem o papel de explicador do primeiro. Normalmente, o sistema a ser interpretado é caixa-preta, enquanto a técnica usada para explicação é interpretável. São exemplo de abordagens *post-hoc*: O LIME (*Local Interpretable Model-agnostic Explanations*) (Ribeiro et al., 2016), explicações contrafactuais (Wachter et al., 2017) e explicações aditivas de Shapley (SHAP) (Lundberg e Lee, 2017), dentre outros.

2.4.2 Específico ou agnóstico em relação ao modelo

Outra possível subdivisão refere-se ao tipo de modelo em que é possível aplicar técnica de explicação. Ferramentas de interpretação que podem ser aplicadas à classes de modelos específicas são classificadas como tal. Ou seja, uma ferramenta que for aplicável somente à Redes Neurais Artificiais, por exemplo, são específicas para este modelo. Por outro lado, aquelas que podem ser utilizadas para qualquer tipo de modelo ML, são

consideradas agnósticas quanto ao modelo. Por padrão, nesse ultimo não é acessado os componentes internos do modelo caixa-preta a ser explicado.

Dentre as vantagens em se utilizar as ferramentas de interpretação agnósticas, quando se comparado a métodos de interpretação específico quanto ao modelo, pode-se citar: flexibilidade em relação ao modelo, pois tais métodos podem ser aplicados para qualquer algoritmo de ML, flexibilidade em termos de explicação, pois não há limitação do tipo de explicação ofertada e flexibilidade quanto à representação, diferentes recursos de acordo com o tipo de modelo que está sendo explicado (Molnar, 2019).

São exemplos de modelos específicos para Redes Neurais Artificiais e Convolucionais os propostos em Ying et al. (2019); Wang et al. (2020), e de métodos agnósticos em relação ao modelo o LIME (Ribeiro et al., 2016), o SHAP (Lundberg e Lee, 2017), o GPX (Ferreira et al., 2020), o DALEX (Biecek, 2018), dentre outros.

2.4.3 Escopo das Explicações: local ou global

Finalmente, a última classificação aqui abordada refere-se ao escopo em que as explicações são feitas. Embora um modelo caixa-preta possa ser muito complexo para ser completamente entendido, existe a possibilidade de que uma determinada instância possa ser explicada utilizando a vizinhança em torno dela (Ribeiro et al., 2016). A explicação local consiste em recuperar as regras que descrevam os motivos pelos quais uma determinada decisão foi tomada por um modelo caixa preta b para uma única instância x, enquanto a explicação global consiste em encontrar as razões para a classificação para qualquer instância em X (Guidotti et al., 2018).

Em outras palavras, os modelos locais interpretáveis fazem uma aproximação local de como a caixa-preta funciona, de forma que as explicações são construídas para as decisões tomadas por um modelo de caixa preta sobre uma instância ou um conjunto delas (Confalonieri et al., 2021), e fornecem o entendimento de como a previsão do modelo muda, se determinada entrada for ligeiramente alterada. São exemplos de métodos que fornecem explicações locais: LIME (Ribeiro et al., 2016) e suas extensões (Botari et al., 2020; Visani et al., 2020a; Sokol et al., 2019; Zhao et al., 2021), SHAP (Lundberg e Lee, 2017), GPX (Ferreira et al., 2020), MAPLE (Plumb et al., 2018), CASTLE (La Gatta et al., 2021), dentre outros.

E as explicações globais são capazes de descrever a lógica geral do método avaliado. Contudo, tais abordagens têm requisitos mais rígidos, já que os usuários podem exigir acesso tanto ao modelo quanto aos dados usados para treinar e validar o modelo (Setzu et al., 2021). São exemplos de métodos que fazem esse tipo de explicação o SHAP (Lundberg e Lee, 2017), TCAV (Kim et al., 2018), GAM (Ibrahim et al., 2019), (Schrouff et al., 2021; Wu et al., 2020), dentre outros.

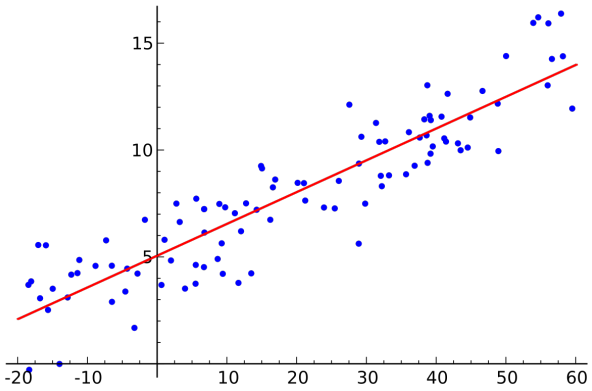
2.5 Métodos de ML interpretáveis

Uma maneria simples de alcançar interpretabilidade é utilizar algoritmos que são interpretáveis, como regressão linear, regressão logística ou árvores de decisões.

2.5.1 Regressão Linear e logística

A regressão linear é um modelo matemático que estima o valor de uma variável dependente y em função de um vetor de variáveis x, sendo que a relação entre x e y é linear. A Figura 3 apresenta um exemplo de regressão linear de uma variável independente. Este modelo é considerável interpretável visto que os coeficientes descrevem a relação matemática entre cada variável independente e a variável dependente.

Figura 3 – Regressão Linear



Fonte: Alves (2019)

A diferença principal entre a Regressão Linear e a logística, é que o resultado dessa segunda é uma variável categórica (ou discreta). Além disso, as duas também se diferenciam em termos da interpretação dos coeficientes, já que esta interpretação depende da família (binomial, Poisson, etc.) e do link (log, logit, log inverso etc.) utilizado, no caso da Regressão logística.

2.5.2 Árvore de Decisão

As árvores de decisões (DT) são métodos estatísticos muito utilizados para tarefas de classificação e regressão. Na implementação padrão, o espaço de recursos (ou dados) é particionado recursivamente em sub-regiões, a partir de uma função de impureza (geralmente o cálculo da medida de entropia de cada atributo). Com base neste valor, ocorre a partição em sub-regiões disjuntas, até que cada uma delas seja majoritariamente homogênea em relação a uma classe particular. Neste método supervisionado, que utiliza a estratégia dividir para conquistar, é possível extrair regras do tipo "se-então" para inferência da saída resultante.

Em uma DT, cada um dos nós internos, inclusive o nó raiz, representa um hiperplano que é paralelo em relação ao eixo do atributo utilizado para esse particionamento. Já os nós folhas representam as classes do problema, de forma que o caminho entre o nó raiz e o nó folha utilizado para fazer uma predição é facilmente identificado. Em virtude disso, esse método é considerável interpretável. A Figura 4 traz um exemplo didático de uma DT.

Figura 4 – Árvore de Decisão

Fonte: Tech (2019)

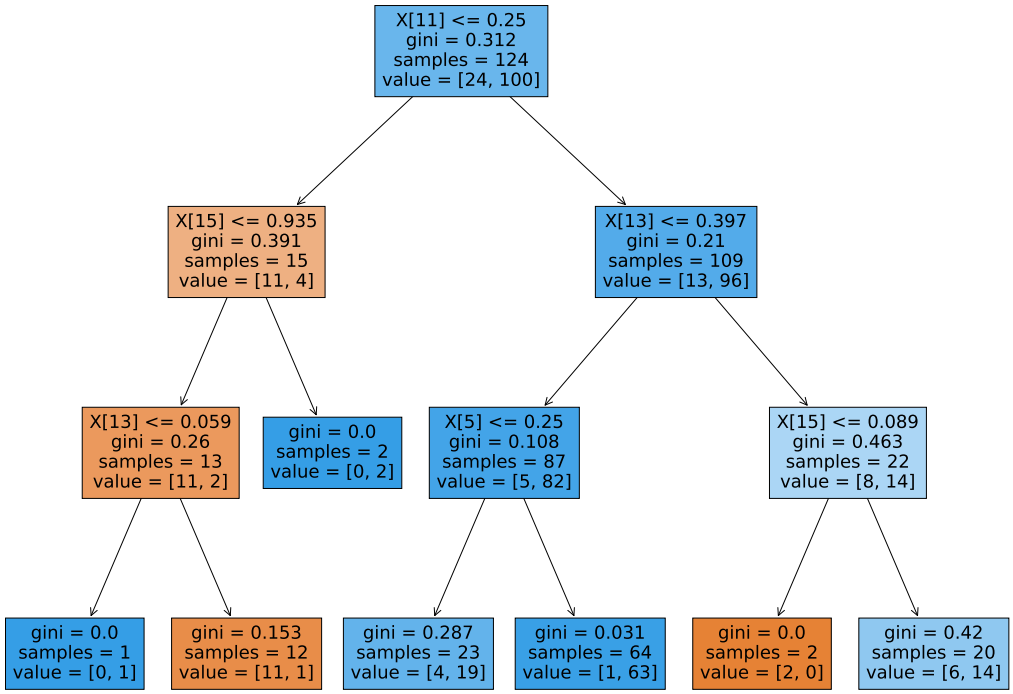
Se utilizada como classificador ou regressor, esse modelo possui interpretabilidade direta, ou seja, é possível entender o caminho gerador da predição via inspeção visual. Esse modelo também pode ser utilizado para fornecer explicações para instâncias específicas, semelhante ao que ocorre com algumas técnicas de interpretabilidade, como o LIME e a GPX. A Figura 5 traz um exemplo de uma DT, ajustada para a predição de hepatite, a partir da base de dados "hepatitis" disponível no UCI (Asuncion e Newman, 2007).

2.6 Técnicas *post-hoc* de interpretabilidade

2.6.1 LIME

O LIME (Local Interpretable Model-Agnostic Explanations) (Ribeiro et al., 2016) é um algoritmo de interpretabilidade local agnóstico em relação ao modelo, que explica as predições de qualquer classificador, aproximando-o localmente de um modelo interpretável. O funcionamento deste explicador consiste em modificar somente um dado da amostra e verificar o impacto dessa mudança na variável de saída, resultando em uma lista de explicações que relaciona o impacto de cada um dos atributos nas predições. Ou seja, são feitas pertubações da amostra de interesse, e o efeito dessas perturbações sobre a saída é mensurado. A saída deste interpretador consiste em um conjunto de explicações que apresenta a contribuição de cada recurso para a previsão realizada. Com isso, a técnica consegue fornecer interpretabilidade local das previsões individuais.

Figura 5 – Exemplo de explicação local fornecida por uma DT



Fonte: Os autores

Assim, o que o LIME faz para gerar explicação para uma determinada saída de um método ML caixa-preta sobre um ponto de dados x é minimizar a perda de reconhecimento de localidade, medindo o quanto o modelo a ser explicado pode ser aproximado por meio de modelo linear, na vizinhança de x, ao mesmo tempo que mantenha a complexidade do modelo baixo. A Figura 6 apresenta um exemplo de aplicação deste explicador para a base de regressão "*Wine-quality Red*"disponível no repositório de '*Machine Learning* da '*UCI - University of California Irvine* (Asuncion e Newman, 2007) para uma predição realizada pelo por um modelo *Random Forest* aplicado a uma instância individual escolhida aleatoriamente.

Conforme pode ser observado, o LIME fornece uma medida de importância para cada um dos atributos (por meio do valor dos pesos associados a eles), informa se essa medida é positiva ou negativa em relação ao valor predito, ou seja, se a predição aumenta

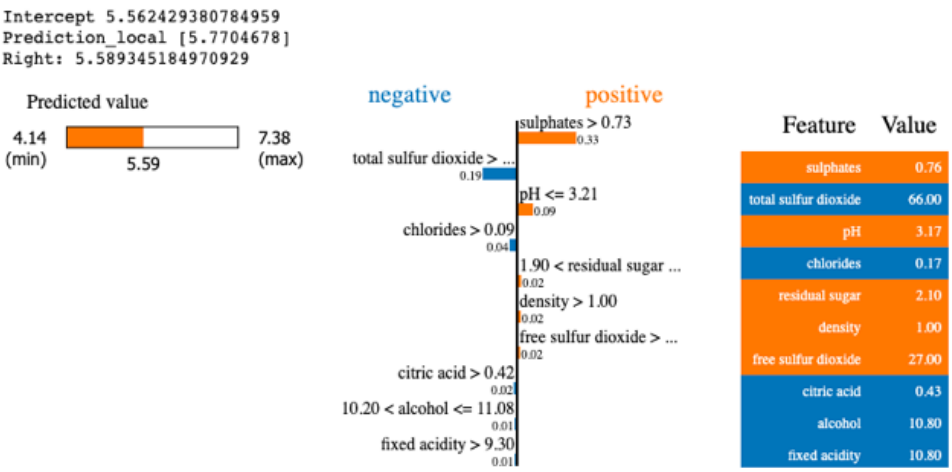


Figura 6 – Exemplo de explicação local fornecida pelo LIME

Fonte: Kuo (2020)

o reduz de acordo com o incremento de cada recurso, os limites de decisão sobre cada um dos atributos, o valor predito, o valor real gerado pelo modelo RF, e o limite inferior e superior do valor de predição.

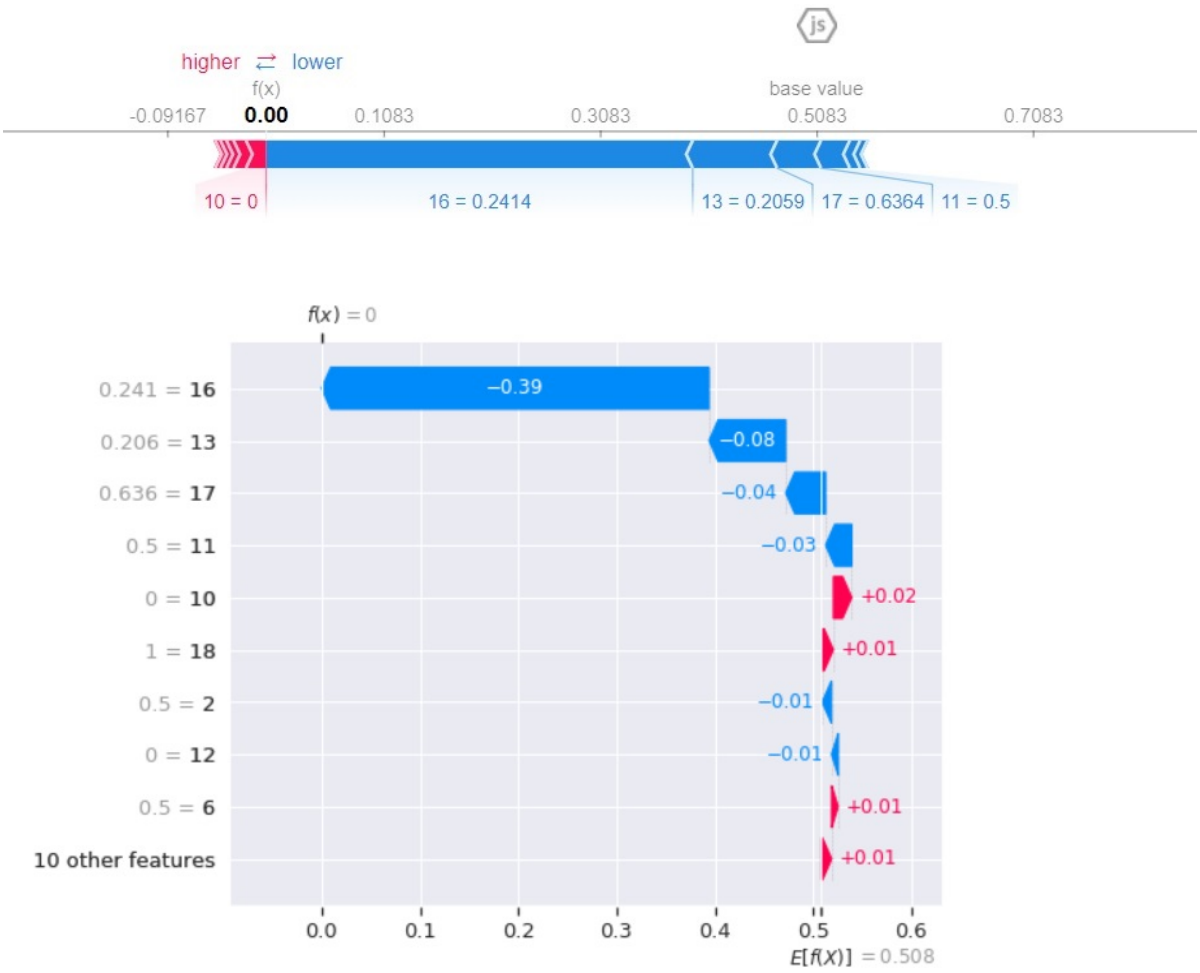
A principal vantagem deste explicador é a possibilidade de ser aplicado para diferentes tipos de dados, como tabular, texto e imagens. Porém, um dos problemas associado a ele, é que apenas modelos lineares foram usados para aproximar o comportamento local. Em contextos em que a região em torno da amostra a ser explicada é pequena, e pode ser aproximada utilizando um modelo simples, a técnica pode funcionar bem. Contudo, para casos em que as regiões possuem alta complexidade, o modelo linear pode não ser um bom aproximador do caixa-preta. Para essas situações, a utilização deste modelo não é interessante.

2.6.2 SHAP

O SHAP (Lundberg e Lee, 2017) faz as explicações da predição de uma instância, verificando as contribuições individuais de cada um dos atributos da base de dados. Este método utiliza o conceito de valor Shapley (Shapley, 1953) baseado em teoria dos jogos, em que cada jogador deve ser recompensado de acordo com a contribuição individual dada para o alcance de determinada recompensa. Nessa analogia, jogadores equivalem aos atributos da base de dados, enquanto o pagamento é calculado a partir da diferença entre a previsão média da instância pela previsão média de todas as instâncias. Considerando isso, os valores Shapley conseguem explicar as contribuições de cada variável do problema para a predição. Este método produz explicações locais e globais. A Figura 7 apresenta um exemplo de uma explicação local gerada por este método para a predição realizada por um modelo RF para uma amostra individual escolhida aleatoriamente a partir da base de

dados “hepatitis"disponível no UCI (Asuncion e Newman, 2007). Ambas imagens trazem explicações para a mesma instância, usando diferentes formatos de apresentação.

Figura 7 – Exemplo de uma explicação local realizada pelo SHAP



Fonte: Os autores

2.6.3 PDP

O PDP (Partial Dependence Plot, em português: Gráfico de Dependência Parcial) (Greenwell, 2017) é um pacote em R que exibe, graficamente, a representação de um ou dois atributos em relação à predição realizada. Nesse caso, é possível verificar se a relação parcial entre determinada característica e a saída é linear, monolítica ou mais complexa (Molnar, 2019). Ou seja, PDPs ajudam a visualizar a relação média parcial entre a resposta predita para uma ou mais características (Goldstein et al., 2015). Este tipo de estudo é interessante para problemas pequenos, já que a visualização é limitada a duas dimensões.

2.6.4 ICE

Já no ICE (Individual Conditional Expectation, Expectativa Condicional Individual) (Goldstein et al., 2015), que também é um método de visualização gráfica, é utilizado para instâncias individuais, pois mantém-se todos os atributos fixos, exceto um e modifica-se esta característica individualmente, e gera-se um conjunto de pontos onde é observado o impacto dessas mudanças na predição.

2.6.5 EXPLAIN

No método EXPLAIN (Robnik-Šikonja e Kononenko, 2008) é computado o quanto determinada característica influencia a saída do sistema. Nesse caso, o método considera que quanto maior a mudança na saída, mais importante é o atributo na predição. Todavia, essa abordagem leva em consideração somente um único recurso por vez, e consequentemente, não pode detectar certas dependências de ordem superior (em particular disjunções) e redundâncias no modelo (Robnik-Šikonja e Bohanec, 2018).

2.6.6 Outros

Sabendo da dificuldade em se obter explicabilidade e interpretabilidade a partir de modelos de Aprendizagem Profunda, Samek et al. (2017) apresentam fundamentos que justificam a explicabilidade na IA, a saber: Verificação, melhora, e aprendizado do sistema, bem como a conformidade com a legislação. Tais argumentos justificam-se na premissa de que ao conhecer e entender o modelo adotado, assim como suas vantagens e desvantagens, pode-se empreender mudanças e melhorias nele. Tendo isso em vista, os autores apresentaram duas abordagens no que tange a visualização, interpretação e explicabilidade de modelos de aprendizagem profunda. A primeira, denominada Análise de Sensibilidade, consiste em fazer pequenas mudanças nas entradas, e verificar o efeito sobre a saída. O segundo, por sua vez, conhecido como Propagação de Relevância com Camada (*Layer-Wise Relevance Propagation*) trata-se de um *framework* que faz a decomposição das decisões em termos das entradas recebidas. A partir dessa avaliação, os autores reforçam que o entendimento dos modelos de IA podem tanto ajudar na adoção da sociedade por esses modelos mais amplamente, além de ajudar a detectar problemas do modelo ou da base de dados utilizada, ajudando na evolução dos mesmos.

E no que se refere aos jogos computacionais, categoria na qual a IA é largamente utilizada, Van Lent et al. (2004) reconhecem que a compreensão do porquê determinadas ações são tomadas torna-se cada vez mais difícil, à medida que o sistema fica mais complexo. Nesse contexto, a explicabilidade tem grande importância, tendo em conta que muitas das avaliações da jogabilidade desses games estão diretamente associadas com o comportamento

da IA. Pensando nisso, o trabalho do autor apresenta a descrição da IA utilizada pela *Full Spectrum Comman* (FCS), bem como a explicabilidade agregada ao comportamento dela a partir de uma determinada ação, em algumas fases analisadas. A partir disso, os autores relatam que é possível prover ao usuário o entendimento concernente às perguntas deste a respeito da IA do sistema estudado.

2.7 Estado da Arte

Os explicadores mais populares, como o LIME e o SHAP possuem algumas limitações que fizeram com que outros métodos fossem propostos para supri-las. Ambos os explicadores produzem incerteza nas explicações quando há interações entre os recursos (Gosiewska e Biecek, 2019), de forma que essa correlação deve ser considerada para reduzir este problema.

No que se refere à aplicação para problemas de classificação e regressão, alguns autores citam a baixa fidelidade do LIME quanto as predições do método de ML caixa-preta como a principal desvantagem desse método (Molnar, 2019). Uma vez que o LIME utiliza um modelo linear para fazer a aproximação local, se a região de interesse tiver muitas não linearidades, o modelo linear não consegue ajustar bem a função de interesse. Além disso, o modelo sofre também com falta de estabilidade local, já que é gerada uma vizinhança aleatória baseada em distância euclidiana. Para resolver o problema da estabilidade, foi proposto o GALE (Xenopoulos et al., 2022). Trata-se de um *framework* que transforma explicações locais de um conjunto de dados em uma representação topológica que pode ser aplicado para comparar métodos de explicação heterogêneos. Isso é feito a partir da modelagem da relação entre o espaço de explicação e as previsões do modelo como uma função escalar. Em seguida, o esqueleto topológico dessa função é obtido. Este esqueleto topológico atua como uma assinatura para tais funções, que é utilizada para comparar diferentes métodos de explicação. Já Staniak e Biecek (2018) propuseram o *live* para lidar com o problema da estabilidade, alterando a amostragem realizada para uma perturbação de um recurso por vez, e as variáveis originais são usadas como entradas interpretáveis. Também para resolver este problema, uma outra abordagem de vizinhança foi avaliada para o DLIME (Zafar e Khan, 2019). Neste caso, ela é definida a partir do agrupamento das amostras semelhantes da base de treino em clusteres usando o algoritmo *K-Nearest Neighbour* (KNN) em conjunto com o *Agglomerative Hierarchical Clustering* (AHC). Embora isso consiga resolver o problema mencionado, gera-se uma dependência da disponibilidade dos dados de treinamento para utilização deste método. Ademais, se a base de dados não contiver muitas amostras na região de interesse, a aproximação ficará ruim, resultando em baixa fidelidade. Para lidar com o problema da baixa fidelidade, Ferreira et al. (2020) propôs o *Genetic Programming Explainer* (GPX), que é um explicador baseado em programação genética e agnóstico em relação ao modelo, que ao invés de

utilizar um modelo linear, faz uso de programação genética para gerar expressões que descrevam o comportamento do método opaco para uma instância.

Em relação ao SHAP, uma das limitações identificadas é que quando há duas variáveis com alto grau de multicolinearidade, os valores de SHAP atribuídos a elas serão altos para uma delas e zero ou muito baixo para a outra. Considerando que uma das propostas deste sistema é fornecer o grau de importância de cada atributo, nesse caso, a importância atribuída a uma dessas características estará incorreta. Além disso, as desvantagens dos valores Shapley também se aplicam ao SHAP: os valores Shapley podem ser mal interpretados, e é necessário recalculá-los para novos dados (exceto para TreeSHAP), aponta Molnar (2019).

Isso posto, o presente trabalho tem o intuito de propor um novo explicador local, baseado em PDT, para problemas de classificação e regressão, agnóstico em relação ao modelo, que consiga superar as principais limitações identificadas nesses métodos mais populares, e fornecer ao especialista diferentes opções de visualizações para compreender e avaliar as explicações geradas pelo modelo. Haja vista que as PDTs são descritíveis e interpretáveis (Zázvorka, 2020), com maior potencial de aproximação local que modelos lineares, pode-se fornecer explicações mais precisas ou equiparáveis aos modelos atuais. Além disso, essa estrutura permite que diferentes informações possam ser obtidas, como o grau de importância de cada atributo, os limites de decisões associados a eles, bem como as regras seguidas para obtenção da saída.

Considerando-se que muitas PDTs da literatura foram evoluídas por meio de meta-heurísticas (Rivera-Lopez et al., 2022), e que o algorítimo de Evolução Diferencial é uma dessas abordagens que desde que foi proposto apresentou boas soluções aplicadas à problemas contínuos (Yang et al., 2011), e os resultados apresentados pela aplicação desta técnica foram promissores em relação às DTs (Lopes et al., 2012), considera-se que a aplicação de uma versão otimizada deste método possa gerar bons resultados para essa tarefa. Tendo isso em vista, neste trabalho é proposto um novo processo para obtenção dos pesos dos hiperplanos associados a essa árvore, e uma vez evoluída para um modelo com alta fidelidade em relação ao caixa-preta, o processo de extração de explicações pode ser realizado. Todas as etapas consolidadas juntamente com as saídas fornecidas são denominadas de *Perceptron Decision Tree Explainer* (PDTX). Para que este método possa ser usado para extração de explicações é fundamental que se comporte de maneira similar ao caixa-preta, dentro da localidade que se deseja explicar, e para isso deve ser feito o processo de indução dos pesos dessa árvore, de forma que ela seja treinada a imitar tal comportamento. Em virtude disso, no próximo capítulo é tratado como a técnica de otimização utilizada para a indução das PDTs funciona, e como se deu a evolução dessa heurística – desde a Evolução Diferencial até o jSO (versão otimizada escolhida) – onde várias melhorias foram incorporadas desde a versão inicial, e a tornam adequada para

realização desta tarefa.

2.8 Considerações do capítulo

1. O avanço da capacidade computacional trouxe um incremento vultoso de sistemas que são capazes de tomar decisões a partir de experiências anteriores. Esses sistemas, que tem o funcionamento fundamentado em métodos conhecidos como ML, podem ser compreensíveis ou de difícil interpretação. No primeiro caso, são referidos como caixa-branca, enquanto no último geralmente são chamados caixa-preta;

2. Esses sistemas são assim chamados pois mesmo que todos os parâmetros do modelo sejam conhecidos, não é possível descrever como se deu a tomada de decisão;

3. Existe um *trade-off* entre a capacidade de explicação dos modelos (interpretabilidade) e o desempenho. Os métodos que possuem os melhores resultados normalmente são complexos de entender;

4. Diversos são os motivos pelos quais se justifica a necessidade de compreender o funcionamento interno de sistemas opacos, bem como das regras que produziram determinadas saídas. Dentre eles, cita-se a aceitação social desses sistemas, a detecção de vieses contidos nos dados de treino, a necessidade de garantir a segurança dos usuários, além da possibilidade de melhor depurar e auditar esses sistemas;

5. Quanto à taxonomia, esses modelos podem ser divididos em intrínseco ou *post-hoc*, específico ou agnóstico em relação ao modelo, e o escopo das explicações pode ser local e/ou global;

6. Em relação aos métodos que são naturalmente interpretáveis destaca-se a regressão linear e logística e algumas DTs. Para modelos que são caixa-preta uma das opções é a utilização de técnicas *post-hoc* de interpretabilidade, como o LIME, SHAP, PDP, ICE, Explain e outros.

7. A simplicidade contida em métodos globalmente lineares, como o LIME, faz com que o ajuste realizado por este método ao fazer a aproximação com o caixa-preta não seja de boa qualidade em contextos em que a aproximação é feita para sub-regiões complexas. Dessa forma, a aplicação de alternativas que não sejam tão simples, que consigam fazer uma boa aproximação local, mas que mantenham o potencial de explicabilidade, faz-se uma opção interessante para realização dessa tarefa.

8. A técnica proposta neste trabalho, o PDTX, tem o intuito de reduzir as principais desvantagens conhecidas dos explicadores mais populares atualmente (como LIME e SHAP) e é estudada para produzir diferentes tipos de explicações a partir de PDTs,

como: a importância dos atributos, os limites de decisões, as regras utilizadas para se chegar às respostas, etc. Referente às classificações estudadas, o PDTX produz interpretabilidade local, é agnóstico quanto ao modelo, e produz explicações *post-hoc*.

Capítulo 3

Evolução Diferencial

Um problema de otimização pode ser definido como aquele cujo objetivo é encontrar a melhor dentre todas as possíveis soluções, que possui o valor mínimo ou máximo de função objetivo, dentro da região factível (Karger et al., 2010). Dada uma função genérica $f(x)$, um problema de otimização irrestrito pode ser definido matematicamente como:

$$x^* = \arg\min f(x), \quad x \in K \subset \mathbb{R}^n$$

em que x^* é o argumento que minimiza $f(x)$ para todo o domínio de x, K é a região factível para o problema proposto, e é subconjunto de $\mathbb{R}^n$, onde n é a dimensão do problema trabalhado.

Nesse sentido, os algoritmos de otimização são importantes estratégias para solução de problemas complexos, conforme apontado por Eltaeib e Mahmood (2018). Algumas razões podem torná-los difíceis de resolver, como enumeradas pelos autores. A primeira, é quando o espaço de busca é muito grande, dificultando a realização de uma busca abrangente; a segunda, são situações em que a função de avaliação pode ter ruídos ou variar de acordo com o tempo, levando a mais de um valor de solução; e por último, quando as restrições impedem que se chegue a uma solução factível.

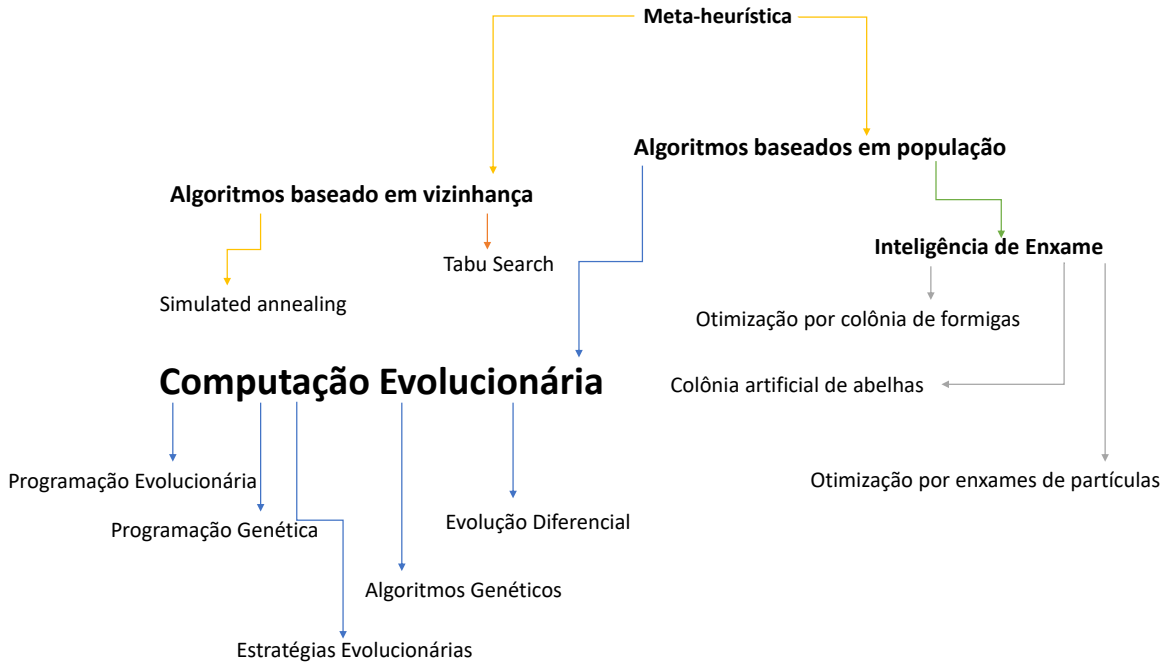
Diante dessas e outras dificuldades, muitos pesquisadores se concentram na proposição e investigação de algoritmos genéricos, conhecidos como meta-heurísticas, que podem ser explorados para resolver vários desses problemas (Pant et al., 2020). Uma meta-heurística pode ser definida como uma estrutura algorítmica geral a ser aplicada a diferentes problemas de otimização com relativamente poucas modificações que possam ajustá-la a um problema específico (Levy Boccato, 2014).

São inclusas nessa classe estratégias inteligentes de alto nível, na maioria das vezes de comportamento estocástico, e que imita algum processo biológico ou físico, e que são capazes de obter boas soluções, com custo computacional viável, independente da natureza do problema, sem garantia de viabilidade ou otimalidade (Beheshti e Shamsuddin, 2013).

Apesar disso, essas soluções podem ser aplicadas em diversos campos, onde ainda não foram concebidos algoritmos exatos de solução ou mesmo heurísticas específicas, além do fato de, alguns casos, os métodos convencionais que garantem a localização da melhor solução serem infactíveis. Nestas situações, as meta-heurísticas se tornam candidatas interessantes (Levy Boccato, 2014).

As meta-heurísticas podem ser globalmente divididas em duas categorias, conforme Pant et al. (2020): Algoritmos baseados em vizinhança ou baseados em população. Dois métodos populares que encontram-se na primeira subdivisão são: *Simulated Anneling* e *Tabu Search*, enquanto a segunda compreende métodos inspirados na natureza, podendo ser subdivididos em dois grandes grupos: Inteligência de Enxame e Algoritmos Evolucionários. A Figura 8 sumariza as principais categorias e sub-categorias dentro das meta-heurísticas computacionais.

Figura 8 – Meta-heurísticas



Fonte: Pant et al. (2020), adaptado.

Conforme observa-se na Figura 8, dentro do grupo de Algoritmos Evolucionários encontra-se a Evolução Diferencial (*Differential Evolution, DE*), que embora não seja inspirada em um processo biológico específico, faz uso de conceitos comuns dessa classe de métodos, como mutação, recombinação e seleção para gerar e avaliar novos indivíduos, em busca de desenvolver uma coleção de soluções candidatas em direção ao ótimo (Price, 2013). A versão canônica deste algoritmo foi proposta na década de 90 por Storn e Price (1997) e apresentou bons resultados, preliminarmente para funções contínuas (Storn e Price, 1997; Rönkkönen et al., 2009), e mais adiante para problemas combinatórios (Onwubolu

e Davendra, 2009). Esse algoritmo e suas variantes apresentaram bons resultados em diversas aplicações (Pant et al., 2020; Deng et al., 2021; Pishchalnikov, 2018), inclusive para problemas multiobjetivo (da Silva Santos et al., 2021; Liang et al., 2021; Li et al., 2021). Tendo isso em vista, as seções seguintes explicitam o funcionamento dessa versão (Storn e Price, 1997) e citam alguns dos avanços deste algoritmo desde que foi proposto.

3.1 Versão básica

O DE é um algoritmo baseado em população, em que cada um dos N indivíduos é um vetor n-dimensional. Assim, X_t representa a população composta pelos indivíduos $x_{t,i}$ da geração t.

$$X_t = (x_{t,1}, x_{t,2}, x_{t,3}, ..., x_{t,N}),\ i = 1,2,3, ..., N \tag{3.1}$$

O algoritmo inicializa o espaço de busca gerando indivíduos em posições definidas aleatoriamente ou de maneira uniforme que abranja todo o espaço, e usa os operadores de mutação em conjunto com a recombinação (ou cruzamento) para gerar novos indivíduos. Com o operador de seleção, define-se os melhores indivíduos para prosseguir no processo evolucionário. Para isso, é preciso definir previamente a probabilidade de cruzamento, CR, e o peso diferencial F, que pondera a mutação. O processo iterativo básico é descrito na Figura 3.1.

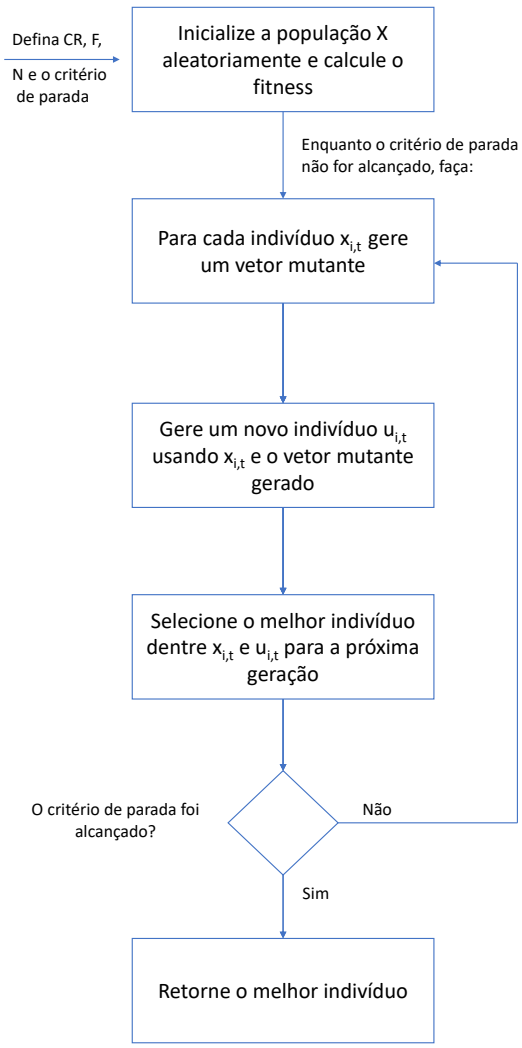
Para melhor compreensão, encontra-se no Algoritmo 1 o pseudocódigo deste otimizador. Como mostrado na linha 4 deste pseudocódigo, o DE é iterativo. Após o processo de inicialização, o algoritmo repete o processo de mutação, recombinação e seleção, até que a condição de parada seja alcançada. Compreende-se que a complexidade do problema reflete no custo computacional depreendido para solução desse. Em virtude disso, diferentes estratégias de mutação e recombinação podem ser usadas para obtenção de melhores indivíduos.

3.1.1 Inicialização

Na inexistência de conhecimento prévio quanto ao espaço de busca (regiões promissoras ou soluções parciais), no processo de inicialização é gerada a população inicial $X_{t=1}$, com indivíduos em todo o espaço K, seguindo uma distribuição uniforme, e predefinindo o limite inferior l_f e superior l_s, conforme mostrado abaixo, caso os mesmos valores sejam usados para todas as dimensões:

$$x_{1,i,j} = l_f + (l_s - l_f) \times \mathcal{U}_{[0,1]}$$

Figura 9 – Fluxograma do DE



Ou seja, são gerados vetores n-dimensionais com valores entre 0 e 1, seguindo uma distribuição uniforme, e estes vetores são multiplicado pela diferença entre o limite superior, e o resultado somado ao limite inferior. Esse processo garante que todos os valores gerados obedecerão os parâmetros de limites pré-definidos.

3.1.2 Mutação

Nessa etapa, para cada um dos indivíduos x_i da população, também chamados de *target vector*, gera-se um vetor mutante u_i (*trial vector*), que é a combinação de três indivíduos da população x_{r1}, x_{r2} e x_{r3}, escolhidos aleatoriamente e que são diferentes do indivíduo x_i. Essa combinação é ponderada pelo peso F, conforme a definição da Equação 3.1.2:

$$u_{t,i,j} = x_{t,r_1,j} + F(x_{t,r_2,j} - x_{t,r_3,j}) \tag{3.2}$$

Algorithm 1 Pseudocódigo da versão básica do DE

```
1▸ t ← 1
2▸ Gerar a população inicial X_t = {x_{t,i}; i = 1,...,N}
3▸ enquanto condição de parada é falsa faça
4▸     para i = 1 até N faça
5▸         Escolha aleatoriamente r_1, r_2, r_3 ∈ {1,...,N}
6▸         Escolha aleatoriamente k_i ∈ {1,...,n}
7▸         para j = 1 até n faça
8▸             se U_[0,1] < CR ∨ j = k então
9▸                 u_{t,i,j} = x_{t,r_1,j} + F(x_{t,r_2,j} − x_{t,r_3,j})
10▸            senão
11▸                u_{t,i,j} = x_{t,i,j}
12▸            fim se
13▸        fim para
14▸        se f(u_{t,i}) ≤ f(x_{t,i}) então
15▸            x_{t+1,i} ← u_{t,i}
16▸        senão
17▸            x_{t+1,i} ← u_{t,i}
18▸        fim se
19▸    fim para
20▸    t ← t + 1
21▸ fim enquanto
```

Com intuito de simplificar a apresentação das variantes mais comuns foi apresentada a notação $DE/a/b/c$, em que:

1. **a** representa o vetor que sofrerá a mutação;
2. **b** define a quantidade de vetores de diferença usados no processo de mutação;
3. **c** especifica o processo de recombinação usado.

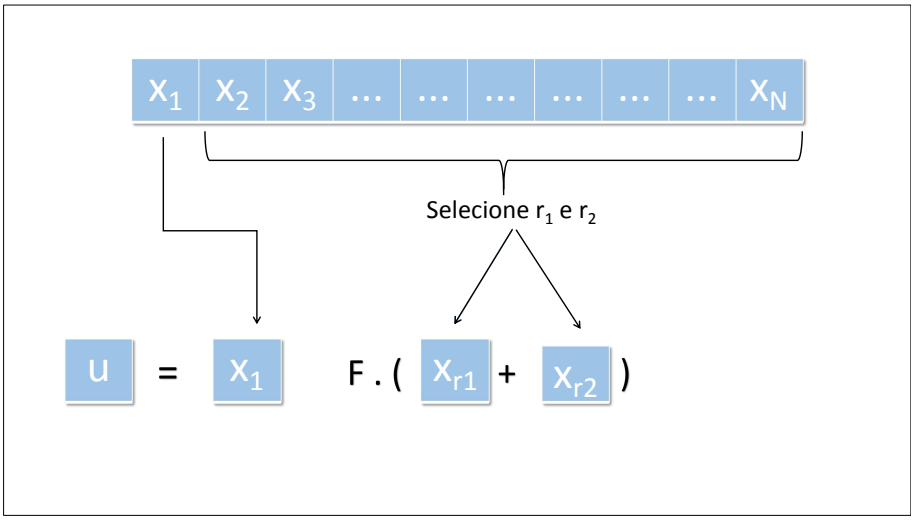
A estratégia de mutação usada na versão clássica é apresentada na bibliografia como $DE/rand/1/bin$. Posteriormente outras estratégias de mutação foram sugeridas, sendo que algumas das mais populares são sumarizadas na Tabela 2.

3.1.3 Recombinação

Na recombinação, também chamada de cruzamento, é gerada a população descendente a partir da combinação de cada indivíduo $x_{t,i}$ com o respectivo vetor mutante $u_{t,i}$, gerando assim um conjunto de soluções candidatas U_t a permanecerem no processo evolucionário.

Para isso, para cada dimensão $x_{t,i,j}$ gera-se aleatoriamente um número entre 0 e 1, e utiliza-se a variável de recombinação CR para determinar quais posições receberão os

Figura 10 – Mutação



Fonte: Os autores

Tabela 2 – Alguns operadores de mutação do DE

Notação	Mutação
DE/rand/1/bin	$u_{g,i} = x_{t,r1} + F(x_{t,r2} - x_{t,r3})$
DE/best/1/bin	$u_{t,i} = x_{t,best} + F(x_{t,r2} - x_{t,r3})$
DE/mean/1/bin	$u_{t,i} = \frac{1}{N}\sum_{k=1}^{N} x_{t,k} + F(x_{t,r2} - x_{t,r3})$
DE/rand-to-best/1/bin	$x_{t,r1} + \lambda(x_{t,best} - x_{t,r1}) + F(x_{t,r2} - x_{t,r3})$
DE/current-to-best/1/bin	$x_{t,i} + \lambda(x_{t,best} - x_{t,i}) + F(x_{t,r2} - x_{t,r3})$
DE/rand/2/bin	$x_{t,r1} + F_1(x_{t,r2} - x_{t,r4}) + F_2(x_{t,r3} - x_{t,r5})$

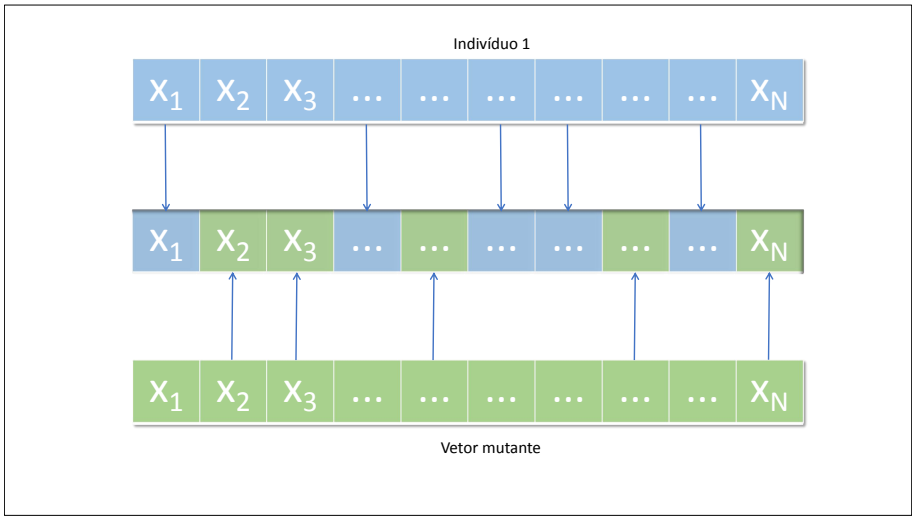
valores de $u_{t,i}$, caso os valores gerados sejam menores que CR, e quais receberão os valores originais de $x_{t,i}$, caso contrário.

Para garantir que pelo menos uma das posições necessariamente sofrerá a mutação, utiliza-se ainda a condição $j = k$, onde k é uma posição da população, escolhida aleatoriamente, que será verdadeira pelo menos uma vez.

3.1.4 Seleção

Após a geração da população de indivíduos candidatos U_t calcula-se o fitness de cada um dos indivíduos gerados pelos processos anteriores. O processo de seleção é utilizado é elitista, visto que cada um dos elementos $x_{t,i}$ da população P_t é comparado com o respectivo indivíduo $u_{t,i}$ de U_t tendo como parâmetro o valor de *fitness* de cada um deles. Os melhores indivíduos prosseguem no processo evolutivo. Em um problema de otimização de minimização, por exemplo, seleciona-se os elementos que possuem os menores *fitness*.

Figura 11 – Recombinação



Fonte: Os autores

3.2 Mudanças e melhorias do DE

Desde que foi proposto, o DE passou por muitas modificações e melhorias, em busca de torná-lo aplicável a problemas de otimização complexos. Essas modificações vão desde o processo de inicialização, bem como nas demais etapas que compõe o algoritmo. Além de técnicas de controle adaptativo dos parâmetros de cruzamento e mutação.

A Tabela 3 sumariza algumas das modificações propostas na literatura, apresentadas em Pant et al. (2020).

Muitas dessas mudanças foram incorporadas em diversos dos algoritmos que participaram em algumas das sessões da competição do *IEEE Congress on Evolutionary Computation*. Um desses algoritmos é o jSO, que é uma versão evoluída de várias outras, e por meio dessas alterações, apresentou resultados promissores (Brest et al., 2017). Considerando o bom desempenho para problemas contínuos, esse método foi adotado neste trabalho para evoluir os pesos da PDT a ser utilizada para geração de explicações local.

3.3 Variantes da Evolução Diferencial

Conforme mostrado na seção anterior, várias melhorias foram incorporadas ao DE para ganho de desempenho e ampliação da aplicabilidade, inclusive para problemas multiobjetivo (Adeyemo e Otieno, 2009) e/ou combinatórios (Onwubolu e Davendra, 2009). Nesta seção é apresentado o conjunto de variantes que influenciaram na proposição do algoritmo jSO, utilizado neste trabalho. Com isso, o leitor conseguirá identificar quais modificações foram incorporadas a cada variante, até a versão aqui utilizada. Ao final, na Figura 12 apresenta-se a linha do tempo com todas as versões citadas, bem como a

Tabela 3 – Modificações realizadas no DE

Etapa	Principais alterações
Inicialização	Baseada em oposição (Rahnamayan et al., 2008); Via interpolação quadrática (Pant et al., 2009); Por meio do método de otimização Simplex (Ali e Pant, 2011); Abordagem descentralizada (Tanabe e Fukunaga, 2013b); Compilação de diferentes métodos (Ali et al., 2013); Amostragem inteligente por meio de ML (de Melo e Delbem, 2012);
Mutação	As mencionadas na Tabela 2; Adaptação automática de parâmetros (Zhang e Sanderson, 2009); Mutação adaptativa em torno do ótimo local (Yu et al., 2015); Utilização dos p melhores indivíduos (Bujok e Tvrdík, 2015); Mutação triangular via combinação convexa de vetores (Mohamed, 2015); Mutação baseada em algoritmo heurístico topográfico (Sacco e Henderson, 2014); Operador de mutação de classificação não dominada (Wei et al., 2015); Aplicação do conceito de atração e repulsão (Fan et al., 2010); Operador de mutação seguindo a distribuição de Cauchy (Ali e Pant, 2011); E várias outras abordagens (Pant et al., 2020).
Cruzamento	Cruzamento binomial (Islam, 2012; Zaharie, 2007); Cruzamento exponencial (Zaharie, 2007); Cruzamento centrado nos pais (Pant et al., 2008); Cruzamento aritmético epistático (Fister et al., 2016); Regra de cruzamento preferencial (Ali, 2007); Cruzamento adaptativo (Fan e Zhang, 2016); Cruzamento ortogonal (Wang et al., 2012) Cruzamento baseado em localidade (Mukherjee et al., 2013); Taxa de cruzamento via estratégia linear crescente Dexuan e Liqun (2012); Parâmetros de controle baseado em experiências anteriores (Qin e Suganthan, 2005).
Seleção	Mecanismo próprio de seleção que mantém a diversidade populacional.
Outros	Redução adaptativa no tamanho populacional; Arquivo histórico opcional (Zhang e Sanderson, 2009).

identificação das melhorias implementadas em cada uma delas.

3.3.1 jDE

O DE padrão inclui parâmetros que são mantidos fixos ao longo de todo o processo evolutivo, e que precisam ser ajustados para cada problema de otimização. Esse processo é árduo (Georgioudakis e Plevris, 2020), mas condiciona diretamente a eficiência do algoritmo. Com isso, surgiram versões com intuito de que essas variáveis sejam mais bem ajustadas. O JDE é uma variante que apresenta configurações de parâmetros de controle auto-adaptativos e apresentou bom desempenho em problemas de *benchmark* numérico (Brest et al., 2006).

Este algoritmo foi desenvolvido por Brest et al. (2006), com objetivo de atualização dos parâmetros $F_{t,i}$ e $Cr_{t,i}$, em nível individual, e em tempo de execução. Se um indivíduo melhor for produzido, esses parâmetros são mantidos, visto que ajudam a gerar melhores descendentes que são mais propensos a sobreviver e, portanto, propagar esses melhores parâmetros; caso contrário, eles serão ajustados conforme mostrado na sequência.

$$F_{t+1,i} = \begin{cases} F_l + rand_1 \times F_u & \text{se } rand_2 < \tau_1 \\ F_{t,i} & \text{caso contrário} \end{cases} \tag{3.3}$$

$$CR_{t+1,i} = \begin{cases} rand_3 & \text{se } rand_4 < \tau_2 \\ CR_{t,i} & \text{caso contrário} \end{cases} \tag{3.4}$$

onde

- $rand_j, j \in \{1,2,3,4\}$, são valores uniformes $\in [0,1]$;
- τ_1 e τ_2 são as probabilidades de ajuste de F e CR, respectivamente. Os autores usaram nos experimentos $\tau_1 = \tau_2 = 0{,}1$ e $F_l = 0{,}1$ e $F_u = 0{,}9$. Novos F recebem valores entre $[0{,}1; 0{,}9]$, aleatoriamente, enquanto novos valores de CR são atribuídos entre $[0,1]$.

Os valores de $F_{t+1,i}$ e $CR_{t+1,i}$ são calculados antes do processo de mutação, e consequentemente influenciam diretamente nesta etapa e nas posteriores. Dessa forma, com este algoritmo já não é mais necessário um extenso empenho para descobrir bons valores de F e CR. Além disso, as regras para a auto-adaptação dos dois parâmetros são simples e fáceis de implementar, sem que aumente significativamente a complexidade ou o esforço computacional em comparação com a versão clássica, apresentando resultados superiores (Liu e Lampinen, 2005).

3.3.2 saDE

A versão canônica do DE é altamente dependente dos valores atribuídos aos operadores de mutação e recombinação, o que pode requerer alto esforço na tentativa de definir manualmente esses parâmetros (Gämperle et al., 2002). Para resolver este problema foi proposto o *Self-adaptive DE algorithm* ou saDE (Qin e Suganthan, 2005) com o intuito de adaptação automática dessas variáveis.

Nessa implementação existem duas possíveis mutações que podem ser selecionadas: *rand/1/bin* e *current-to-best/2/bin*, que são aplicadas de acordo com duas variáveis p_1 e p_2, respectivamente. O valor inicial de $p_1 = p_2 = 0{,}5$. É gerado um vetor $Prob$ do tamanho da população com elementos gerados aleatoriamente a partir de uma distribuição uniforme

entre [0,1]. Se $Prob_i \leq p_1$ então aplica-se a mutação *rand/l/bin*, ou *current-to-best/2/bin*, caso contrário.

A quantidade de vezes que *rand/l/bin* e *current-to-best/2/bin* é escolhida como mutação é armazenada em ns_1 e ns_2, respectivamente. E o número de vezes que os vetores de mutação são descartados para cada uma das mutações é nf_1 e nf_2, respectivamente. Esse número é acumulado dentro de uma quantidade de gerações pré-definida. A atualização das probabilidades ocorre conforme abaixo:

$$p_1 = \frac{ns1 \times (ns2 + nf2)}{ns2 \times (ns1 + nf1) + ns1(ns2 + nf2)}$$

$$p_2 = 1 - p_1$$

Os valores referente ao operador de mutação F são gerados dentre o intervalo de (0,2], aleatoriamente, para cada geração de acordo com uma distribuição normal com média 0,5 e desvio padrão 0,3. Quanto ao operador de cruzamento, os valores são gerados aleatoriamente a partir de distribuição normal independente com valor de média CRm e desvio padrão 0,1, denotado por $N(CRm; 0{,}1)$, sendo CRm inicializado com 0,5.

Acredita-se que o procedimento descrito possa evoluir gradualmente a estratégia de mutação mais adequada em diferentes estágios de aprendizagem para o problema em consideração. Posteriormente, esse algorítimo foi melhorado em Qin et al. (2008) para trabalhar com quatro estratégias de mutação, ao invés de duas.

3.3.3 JADE

Conforme mencionado, os valores atribuídos aos operadores de mutação e recombinação podem afetar significativamente o desempenho do DE. De forma que, a definição deles por tentativa e erro, além de exaustiva, gera resultados específicos quanto ao problema a ser resolvido. Em virtude disso, vários autores propuseram mecanismo para definição dessas variáveis, independentemente do problema de aplicação (Brest et al., 2006).

O JADE foi publicado em 2007 por Zhang e Sanderson (2007) com a proposta de melhorar a taxa e confiabilidade de convergência por meio da implementação de uma nova estratégia de mutação denominada *DE/current-to-p-best* e pelo controle adaptativo dos parâmetros de mutação e cruzamento, com um arquivo externo opcional e que pode ser vista como uma generalização da técnica *DE/current-to-best*.

O JADE utiliza não somente a melhor solução, mas um conjunto de boas soluções, que podem ser escolhidas aleatoriamente para participar do processo de mutação. Nesse caso, qualquer uma das $100 \times p\%$ principais, $p \in (0, 1]$, soluções podem ser selecionadas para a aplicação da técnica *DE/current-to-best*. Isso é interessante pois a utilização somente

da melhor solução corrente pode levar à convergência prematura, pois pode levar à baixa diversidade populacional.

A versão sem utilização de arquivo externo o vetor mutante é gerado da seguinte maneira:

$$u_{t,i} = x_{t,i} + F_i(x^p_{t,best} - x_{t,i}) + F_i(x_{t,r1} - x_{t,r2}) \tag{3.5}$$

onde $x^p_{t,best}$ é escolhido aleatoriamente dentre o conjunto das $100p\%$ melhores soluções da população atual, com $p \in (0,1]$ e F_i é o fator de mutação associado ao vetor x_i. Por outro lado, na presença deste arquivo, há uma pequena mudança no último termo da fórmula, como se segue:

$$u_{t,i} = x_{t,i} + F_i(x^p_{t,best} - x_{t,i}) + F_i(x_{t,r1} - \tilde{x}_{t,r2}) \tag{3.6}$$

Assim, $x_{t,i}$, $x^p_{t,best}$ e $x_{t,r1}$ são selecionados da maneira tradicional, enquanto $\tilde{x}_{t,r2}$ é escolhido a partir da união de $P \cup A$, em que A é o arquivo que armazena um conjunto de soluções que falharam no processo de seleção. Caso o tamanho do arquivo alcance um valor pré-definido (normalmente o mesmo tamanho da população ($|A| = |P| = N$), as soluções são removidas do arquivo aleatoriamente. Os resultados numéricos mostraram que o método em questão apresentou performance superior ou pelo menos comparável a outros algoritmos de DE padrão ou adaptativos (Zhang e Sanderson, 2009).

A rotina utilizada pelo JADE é mostrada no Algoritmo 2, adaptada de Zhang e Sanderson (2007). Para cada geração, os valores de CR_i e F_i que consigam gerar um *trial vector* $u_{t,i}$ melhor do que o indivíduo progenitor $x_{t,i}$ são armazenados em S_{CR} e S_F, e ao final da geração t, os valores de μ_{CR} e μ_F são atualizados conforme mostrado nas linhas 26 e 27. Feito esse passo, na geração seguinte os operadores CR e F são gerados a partir dos valores atualizados de μ_{CR} e μ_F.

O parâmetro de controle c é chamado de taxa de aprendizado, e recebe o valor de $c = 0{,}1$, conforme sugerido pelos autores. A função $mean(.)$ calcula a média aritmética do argumento enquanto L calcula a média de Lehmer.

3.3.4 SHADE

O SHADE (Tanabe e Fukunaga, 2013b) foi proposto como uma melhoria do JADE sendo essa a principal base dele, em conjunto com algumas características do saDE. Enquanto o JADE usa somente μ_{CR} e μ_F para fazer a adaptação e controle de parâmetros, o SHADE usa ainda uma memória histórica M_{CR} e M_F, de tamanho H, em cada geração para armazenar a média de S_{CR} e S_F. Os valores a serem usados dessas memórias são definidos aleatoriamente tendo como base o tamanho da memória. Ou seja, se r_i é o índice escolhido aleatoriamente da memória de tamanho $|H|$:

Algorithm 2 Pseudocódigo JADE

Inicialize a população $x_{0,i}$ e defina $\mu_{cr} \leftarrow 0{,}5$ e $\mu_F \leftarrow 0{,}6$
para $t = 1$ até G **faça**
  $CR_i \leftarrow \mathcal{N}(\mu_{CR}; 0{,}1)$, para $i = 1{,}2{,}...{,}N$
  $F_i \leftarrow \mathcal{N}(0; 1{,}2)$ para $\frac{1}{3}$ dos indivíduos e
  $F_i \leftarrow \mathcal{N}_i(\mu_F; 0{,}1)$ para os demais.
  $S_F \leftarrow \emptyset$, $S_{CR} = \emptyset$
  para $i = 1$ até N **faça**
    Escolha aleatoriamente $x^p_{t,best}$ dentre os $100p\%$ melhores indivíduos
    Escolha aleatoriamente $r_1 \neq r_2 \neq i$ do conjunto de $1{,}2{,}...{,}N$
    $u_{t,i} = x_{t,i} + F_i(x^p_{t,best} - x_{t,i}) + F_i(x_{r1,t} - x_{r2,t})$
    Gere j_{rand} ou $\mathcal{U}_j(0{,}1) \leq CR_i$
    para $j = 1$ ate D **faça**
      se $j = j_{rand}$ *ou* $\mathcal{U}_j(0{,}1) < CR_i$ **então**
        $u_{j,t,i} \leftarrow u_{j,t,i}$
      senão
        $u_{j,t,i} \leftarrow x_{j,t,i}$
      fim se
    fim para
    se f$(x_{t,i})$ < f$(u_{t,i})$ **então**
      $x_{t+1,i} \leftarrow x_i$
    senão
      $x_{t+1,i} \leftarrow u_{t,i}$
      $S_{CR} \leftarrow S_{CR} \cup CR_i$, $S_f \leftarrow S_f \cup F_i$
    fim se
  fim para
  $\mu_{CR} \leftarrow (1 - c).\mu_{CR} + c.mean(S_{CR})$
  $\mu_F \leftarrow (1 - c).\mu_F + c.L(S_F)$
fim para

$$CR_i = \mathcal{N}_i(M_{CR,r_i}, 0{,}1)$$

$$F_i = \mathcal{C}_i(M_{F,r_i}, 0{,}1)$$

As memórias são atualizadas elemento a elemento, de acordo com a ordem em que eles foram inseridos na memória. Isto é, o primeiro elemento que foi inserido é também o primeiro elemento a ser atualizado, desde que $S_{CR} \neq \emptyset$ e $S_F \neq \emptyset$.

Outra diferença do SHADE se dá em relação ao parâmetro p usado pela técnica *current-to-pbest/1*, que no JADE é definido manualmente, e no SHADE conforme mostrado na sequência:

$$p_i = \mathcal{N}(p_{min}, 0{,}2) \tag{3.7}$$

em que p_{min} é o conjunto de onde o p-melhor indivíduo é selecionado, sendo que pelo

Algorithm 3 Pseudocódigo SHADE

Inicialize a população $x_{0,i}$ e defina $\mu_{cr} \leftarrow 0{,}5$ e $\mu_F \leftarrow 0{,}5$
Arquivo $A \leftarrow \emptyset$
$k \leftarrow 1$, $t \leftarrow 1$
enquanto O critério de parada não for alcançado **faça**
    $S_F \leftarrow \emptyset$, $S_{CR} \leftarrow \emptyset$
    para $i \leftarrow 1$ até N **faça**
        $r_i \leftarrow$ Escolha de $[1, H]$ aleatoriamente
        $CR_{t,i} \leftarrow \mathcal{N}_i(M_{CR,r_i}, 0{,}1)$
        $F_{t,i} \leftarrow \mathcal{C}_i(M_{F,r_i}, 0{,}1)$
        Escolha aleatoriamente $p_{t,i}$ dentre $[p_{min}, 0{,}2]$
        Gere o trial vector $u_{t,i}$ usando a técnica $current - to - p - best/1/bin$
    fim para
    para $i \leftarrow 1$ até N **faça**
        se $\mathrm{f}(u_{t,i}) < \mathrm{f}(x_{t,i})$ **então**
            $x_{t+1,i} \leftarrow u_{t,i}$
        senão
            $x_{t+1,i} \leftarrow x_{t,i}$
        fim se
        se $\mathrm{f}(u_{t,i}) < \mathrm{f}(x_{t,i})$ **então**
            $x_{t,i} \rightarrow A$
            $CR_{t,i} \rightarrow S_{CR}, F_{t,i} \rightarrow S_F$
        fim se
    fim para
    Se o tamanho do arquivo exceder $|P|$, escolha aleatoriamente indivíduos
    para deleção, até que $|A| \leq |P|$
    se $S_{CR} \neq \emptyset$ e $S_F \neq \emptyset$ **então**
        Atualize $M_{CR,k}, M_{F,k}$ baseado em S_{CR}, S_F
        $k \leftarrow k + 1$
        se $k > H$ **então**
            $k \leftarrow 1$
        fim se
    fim se
    $t = t + 1$
fim enquanto

menos 2 indivíduos são selecionados, $p_{min} = 2/N$. O valor máximo de p sugerido por Zhang e Sanderson (2007) é 0,2.

Segundo os autores, uso da memória histórica do SHADE é semelhante ao saDE, embora difira deste por não armazenar diretamente os valores reais de CR e F como o saDE, pois somente as médias dos parâmetros que geram descendentes melhores que os progenitores dos conjuntos S_{CR} e S_F para cada geração são armazenadas.

O Algoritmo 3 apresenta o pseudocódigo do SHADE. Em 2014 foi publicada uma melhoria do SHADE, também conhecida como SHADE 1.1 (Tanabe e Fukunaga, 2013a), que realiza a redução do tamanho da população linearmente.

3.3.5 L-SHADE

Na sequência, foi proposto o L-SHADE (Tanabe e Fukunaga, 2014), que combina as vantagens do SHADE 1.1, e fornece uma estratégia para redução populacional com base em uma função linear, sem perda de performance, o que se traduz em um menor custo computacional. O mecanismo de atualização de memória usado pelo SHADE 1.1 e também pelo L-SHADE é mostrado no Algoritmo 4.

Algorithm 4 Mecanismo de atualização de memória do SHADE 1.1

se $S_{CR} \neq \emptyset$ e $S_F \neq \emptyset$ **então**
  se $M_{CR,t,k} = \perp$ ou $max(S_{CR}) = 0$ **então**
    $M_{CR,t+1,k} \leftarrow \perp$
  senão
    $M_{CR,t+1,k} \leftarrow mean_{WL}(S_{CR})$
  fim se
  $M_{F,t+1,k} \leftarrow mean_{WL}(S_F)$
  $k = k + 1$
  se $k > H$ **então**
    $k \leftarrow 1$
  fim se
senão
  $M_{CR,t+1,k} \leftarrow M_{CR,t,k}$
  $M_{F,t+1,k} \leftarrow M_{F,t,k}$
fim se

A variável k refere-se à posição da memória a ser atualizada, sendo inicializada em $k = 1$ e é incrementada sempre que um novo elemento é adicionado a ela. Caso o valor de k ultrapasse o tamanho da memória (H), ele é reiniciado para $k = 1$. Semelhantemente ao que acontece nos demais algoritmos que usam os operadores S_{CR} e S_F, caso na t-ésima geração nenhum elemento melhor seja produzido considerando todos os indivíduos da população, a memória não é atualizada, visto que ambos conjuntos são vazios.

A grande novidade do L-SHADE é a redução linear do tamanho da população à medida que o processo evolucionário acontece. Sendo N^{init} o tamanho da população inicial e N^{min} o menor tamanho que a população pode assumir, a diminuição acontece a cada geração seguindo a seguinte fórmula:

$$N_{t+1} = \left[\left(\frac{N^{min} - N^{init}}{MAX\_NFE} \right) NFE + N^{init} \right] \tag{3.8}$$

onde NFE e MAX_NFE são os números atual e máximo de avaliações da função objetivo, respectivamente. Dessa forma, enquanto $N_{t+1} < N_t$ os indivíduos excedentes são deletados da população. Além disso, pelo mecanismo de mutação usado nesse algoritmo, o menor número que N^{min} poderá assumir é 4. O Algoritmo 5 apresenta o pseudocódigo completo do L-SHADE.

Algorithm 5 Pseudocódigo L-SHADE

$t \leftarrow 1, N_t \leftarrow N^{init}$
Inicialize a população $x_{0,i}$ e defina $\mu_{cr} \leftarrow 0{,}5$ e $\mu_F \leftarrow 0{,}5$
Arquivo $A \leftarrow \emptyset$
enquanto O critério de parada não for alcançado **faça**
    $S_F \leftarrow \emptyset$, $S_{CR} \leftarrow \emptyset$
    para $i \leftarrow$ até N **faça**
        $r_i \leftarrow$ Escolha de $[1, H]$ aleatoriamente
        se $M_{CR,r_i} = \perp$ **então**
            $CR_{t,i} = 0$
        senão
            $CR_{t,i} = \mathcal{N}_i(M_{CR,r_i}, 0{,}1)$
        fim se
        $F_{t,i} \leftarrow \mathcal{C}_i(M_{F,r_i}, 0{,}1)$
        Gere o trial vector $u_{t,i}$ usando a técnica *current-to-p-best/1/bin*
    fim para
    para $i \leftarrow$ até N **faça**
        se $\mathrm{f}(u_{t,i}) < \mathrm{f}(x_{t,i})$ **então**
            $x_{t+1,i} \leftarrow u_{t,i}$
        senão
            $x_{t+1,i} \leftarrow x_{t,i}$
        fim se
        se $\mathrm{f}(u_{t,i}) < \mathrm{f}(x_{t,i})$ **então**
            $x_{t,i} \rightarrow A$
            $CR_{t,i} \rightarrow S_{CR}, F_{t,i} \rightarrow S_F$
        fim se
    fim para
    Se o tamanho do arquivo exceder $|P|$, escolha aleatoriamente indivíduos
    para deleção, até que $|A| \leq |P|$
    Atualize as memórias M_{CR} e M_F (Algoritmo 4)
    Calcule N_{t+1} de acordo com a Equação 3.8.
    se $N_t < N_{t+1}$ **então**
        Ordene os indivíduos de P com base no *fitness*;
        Delete os piores indivíduos dentre N_t e N_{t+1}
        Redimensione o arquivo A de acordo com o novo $|P|$
    fim se
    $t = t + 1$
fim enquanto

3.3.6 iL-SHADE

Na sequência, o il-SHADE (Brest et al., 2016) evoluiu o L-SHADE ao modificar os valores de inicialização de algumas variáveis e alterar o procedimento de atualização de memória. Os autores sumarizam as diferenças nos itens abaixo:

- O mecanismo de atualização de memória armazena valores históricos da memória da geração anterior e os usa para calcular os novos valores de memória para a próxima

geração, (linha 5 do Algoritmo 6);

- Cada memória possui H registros, e um deles contém valores que são fixos. Esta entrada não é atualizada, mas valores dela são usados para gerar os parâmetros de controle CR_i e F_i;
- Todos os valores históricos da memória no M_{CR} são inicializados em 0,8, e não com 0,5, além disso há mecanismos para que valores altos de CR seja propagado durante o processo de otimização;
- Se a M_{CR} for atribuído o valor terminal $\perp$, então M_{CR} é redefinido para 0,0;
- Também não é permitido altos valores de F e baixos valores de CR no início do processo evolucionário.

Algorithm 6 Mecanismo de atualização de memória do iL-SHADE 1.1

1▸ **se** $S_{CR} \neq \emptyset$ e $S_F \neq \emptyset$ **então**
2▸ **se** $M_{CR,t,k} = \perp$ ou $max(S_{CR} = 0$ **então**
3▸ $M_{CR,t+1,k} \leftarrow \perp$
4▸ **senão**
5▸ $M_{CR,t+1,k} \leftarrow ((mean_{WL}(S_{CR}) + M_{CR,t,k}/2)$
6▸ **fim se**
7▸ $M_{F,t+1,k} \leftarrow ((mean_{WL}(S_F) + M_{F,t,k})/2$
8▸ $k++$
9▸ **se** $k > H$ **então**
10▸ $k \leftarrow 1$
11▸ **fim se**
12▸ **senão**
13▸ $M_{CR,t+1,k} \leftarrow M_{CR,t,k}$
14▸ $M_{F,t+1,k} \leftarrow M_{F,t,k}$
15▸ **fim se**

Essas mudanças tornaram o desempenho do il-SHADE melhor ou competitivamente semelhante ao do L-SHADE, para o *benchmark* testado, conforme os resultados apresentados pelos autores.

3.3.7 jSO

Finalmente, no algoritmo utilizado neste trabalho, jSO (Brest et al., 2017), foram feitas pequenas mudanças, mas significativas em termos de performance, em relação ao il-SHADE. Nesse contexto, houve alteração quanto ao valor atribuído a $CR_{t,i}$ e $F_{t,i}$ ao longo do processo evolutivo, como mostrado nas linhas 16, 19, 21 e 23 do Algoritmo 7. Além disso, a Equação 3.6 sofre uma pequena alteração para Equação 3.9.

$$u_{t,i} = x_{t,i} + F_w(x^p_{t,best} - x_{t,i}) + F_i(x_{t,r1} - \tilde{x}_{t,r2}) \tag{3.9}$$

onde F_w é obtido da seguinte maneira:

$$F_w = \begin{cases} 0{,}7 \times F & \text{se } nfes < 0{,}2 \times max\_nfes \\ 0{,}8 \times F & \text{se } nfes < 0{,}4 \times max\_nfes \\ 1{,}2 \times F & \text{caso contrário} \end{cases} \tag{3.10}$$

Dessa maneira, o termo em que $x^p_{t,best}$ aparece tem menor influência nas gerações iniciais, visto que é ponderado por um peso pequeno, o que diminui as chances de o algoritmo ficar preso em um mínimo local. O pseudocódigo do jSO pode ser visto no Algoritmo 7.

O mecanismo utilizado para mutação no jSO é denominado *current-to-p-best-w/1/bin*, o qual encontra-se na Equação 3.11.

$$u_{t,i} \leftarrow x_{t,i} + F_w(x_{t,pBest} - x_{t,i}) + F(x_{t,r1} - x_{t,r2}) \tag{3.11}$$

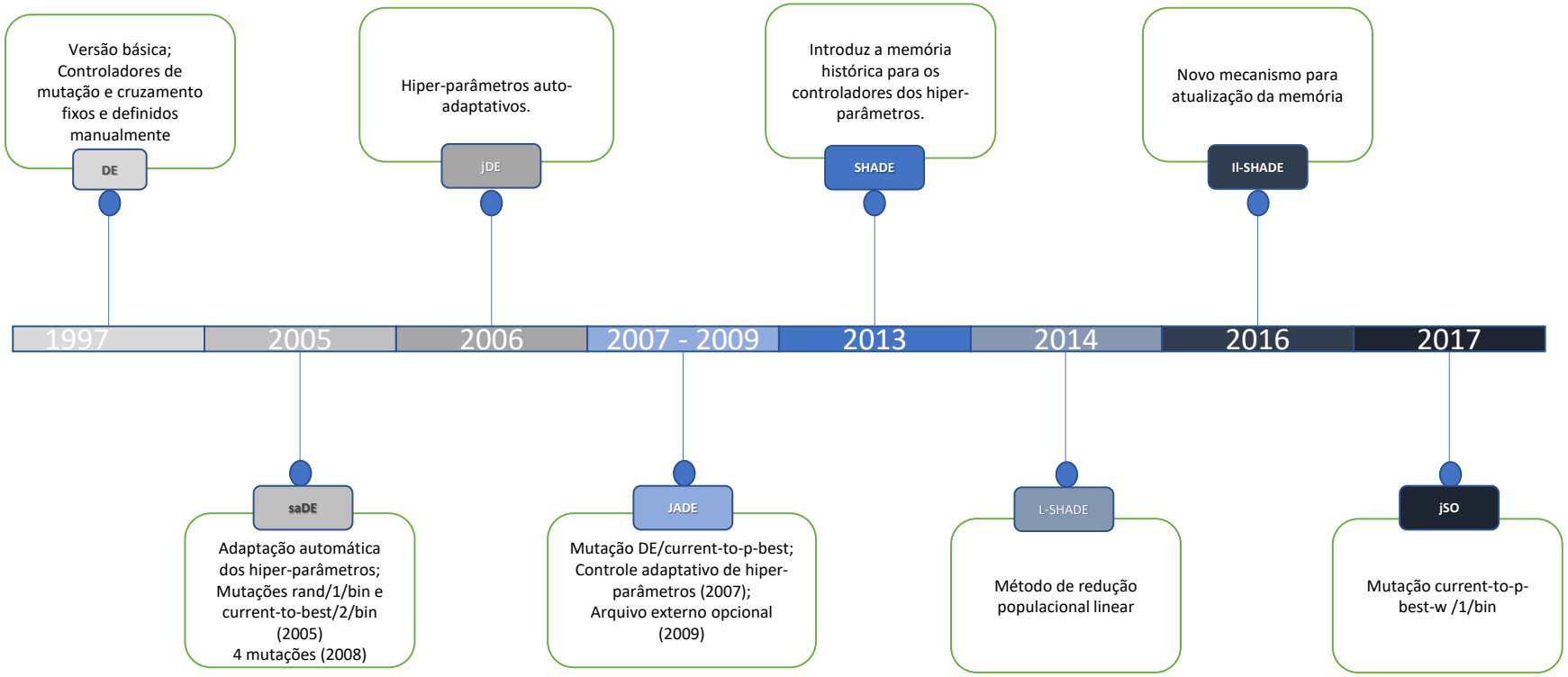
A Figura 12 apresenta a linha do tempo com as principais melhorias identificadas, onde destaca-se:

- Na versão inicial do DE não haviam controladores dos operadores de mutação e cruzamento automáticos. Assim, os valores previamente definidos eram mantidos durante todo o processo evolutivo, e influenciavam diretamente na performance do algoritmo;
- No saDE, em 2005, foi proposta a adaptação automática desses hiper-parâmetros, e a utilização de dois tipos de mutação, Mutações *rand/1/bin* e *current-to-best/2/bin*, de acordo com uma taxa de probabilidade. Mais adiante, em uma evolução desta proposta, em 2008, utiliza-se 4 tipos de mutação e não 2;
- No jDE foi proposto a auto-adaptação dos hiper-parâmetros, que tem os valores são alterados a medida que o processo evolucionário ocorre;
- Já no JADE, proposto em 2007, além do controle adaptativo de hiper-parâmetros, foi proposto também um novo tipo de mutação, chamado *DE/current-to-p-best*, em que os p melhores individuos identificados até o momento são armazenados e utilizados para geração de novos indivíduos, ajudando o algorítmo a não ficar preso em ótimos locais. Na versão proposta em 2009 também há opção da utilização de um arquivo externo opcional, que armazena os indivíduos ruins, que foram descartados na fase de seleção, mas que podem contribuir para variedade populacional;
- No SHADE, de 2013, implementa-se o conceito de memória histórica para os controladores dos hiper-parâmetros, e assim, vários valores dos controladores que tiverem produzido indivíduos ótimos são armazenados;

Algorithm 7 Pseudocódigo jSO

$t \leftarrow 1, N_t \leftarrow N^{init}$
Inicialize a população $x_{0,i}$ e defina $\mu_{cr} \leftarrow 0{,}8$ e $\mu_F \leftarrow 0{,}5$
Arquivo $A \leftarrow \emptyset$
$k \leftarrow 1$
enquanto O critério de parada não for alcançado **faça**
    $S_F \leftarrow \emptyset$, $S_{CR} \leftarrow \emptyset$
    para $i \leftarrow$ até N **faça**
        $r_i \leftarrow$ Seleciona de $[1, H]$ aleatoriamente
        se $r_i = H$ **então**
            $M_{F,r_i} \leftarrow 0{,}9$
            $M_{CR,r_i} \leftarrow 0{,}9$
        fim se
        se $M_{CR,r_i} < 0$ **então**
            $CR_{t,i} \leftarrow 0$
        senão
            $CR_{t,i} \leftarrow \mathcal{N}_i(M_{CR,r_i}, 0{,}1)$
        fim se
        se $t < 0{,}7t_{MAX}$ **então**
            $CR_{t,i} \leftarrow max(CR_{t,i}, 0{,}70)$
        senão se $t < 0{,}5t_{MAX}$ **então**
            $CR_{t,i} \leftarrow max(CR_{t,i}, 0{,}6)$
        fim se
        $F_{t,i} \leftarrow C_i(M_{F,r_i}, 0{,}1)$
        se $t < 0{,}6t_{MAX}$ ou $F_{t,i} > 0{,}7$ **então**
            $F_{t,i} \leftarrow 0{,}7$
        fim se
        Gere o trial vector $u_{t,i}$ usando a Equação 3.11.
    fim para
    para $i \leftarrow$ até N **faça**
        se $\mathrm{f}(u_{t,i}) < \mathrm{f}(x_{t,i})$ **então**
            $x_{t+1,i} \leftarrow u_{t,i}$
        senão
            $x_{t+1,i} \leftarrow x_{t,i}$
        fim se
        se $\mathrm{f}(u_{t,i}) < \mathrm{f}(x_{t,i})$ **então**
            $x_{t,i} \rightarrow A$
            $CR_{t,i} \rightarrow S_{CR}, F_{t,i} \rightarrow S_F$
        fim se
        Se o tamanho do arquivo exceder $|P|$, escolha aleatoriamente indivíduos para deleção, até que $|A| \leq |P|$
        Atualize as memórias M_{CR} e M_F (Algoritmo 6)
        Aplique a estratégia LPSR
        Atualize p usando a Equação 3.7.
    fim para
    $t = t + 1$
fim enquanto

Figura 12 – Linha do tempo variantes entre o DE e o JSO



1

Fonte: Os autores

- No L-SHADE, de 2014, utiliza-se o conceito de redução populacional linear, que propõe um mecanismo para diminuição do tamanho da população ao longo do processo evolucionário, mas que mantém o bom desempenho do algorítimo, o que se traduz em redução do custo computacional;

- No il-SHADE, proposto em 2016, foi incorporado um novo mecanismo de atualização da memória, proposto originalmente no SHADE;

- E finalmente, o jSO, versão escolhida para ser utilizada neste trabalho, foi implementada todas as melhorias citadas anteriormente, além de ter sido proposta um novo tipo de mutação, chamado *current-to-p- best-w /1/bin*. Essa versão foi aplicada para um conjunto de 30 funções contínuas distintas,rotacionadas e deslocadas do eixo original, para as dimensões $D = 10$, $D = 30$, $D = 50$ e $D = 100$, e apresentou resultados superiores às versões anteriores neste contexto. Tendo em vista que a proposta deste trabalho fundamenta-se em otimizar os pesos de uma árvore oblíqua, os resultados apresentados pra este algoritmo reforçam sua capacidade em gerar bons resultados para problemas contínuos complexos.

Uma versão mais recente do DE (Stanovov et al., 2020) revela que o jSO se manteve como um algoritmo de alta performance para problemas contínuos, já que no contexto em que foi testado para 30 funções de *benchmark* das competições do IEEE CEC 2017 e 2020 não teve o desempenho superado para a grande maioria das funções. No trabalho mencionado foi proposto uma técnica chamada *Linear Bias Reduction* (ou Redução Linear de Bias, em tradução livre), inspirada no mecanismo de redução do tamanho populacional. Essa técnica ajuda na redução de viés dos hiper-parâmetros de mutação e recombinação ao longo do processo evolucionário, por meio da atualização dos parâmetros $p_{\min}$ e $p_{\max}$. Pelo apresentado, essa abordagem trouxe até três melhorias para 15 dimensões, e por isso foi incorporada à meta-heuristica aqui utilizada.

3.4 Considerações do capítulo

1. Neste capítulo foi apresentada a versão canônica do DE, em conjunto com as principais características desta meta-heurística. Trata-se de um algoritmo evolutivo baseado em população, que embora não tenha inspiração em um processo natural, como outros métodos dessa classe, utiliza-se de operadores de mutação, recombinação e seleção para conduzir o processo de otimização. Diferentemente dos algoritmos genéticos que usam uma seleção etilista, este método consegue fazer a seleção de bons indivíduos mantendo a diversidade populacional, ao comparar os candidatos a prosseguirem no processo evolucionário par a par - entre o indivíduo da geração corrente e o novo indivíduo gerado a partir desse;

2. Foi apresentado um conjunto de modificações e melhorias que foram apresentadas em extensões deste método ao longo das últimas duas décadas, desde que foi proposto, em conjunto com as referências, caso o leitor desejar aprofundar em alguma dessas técnicas;

3. Na sequência é construída uma linha do tempo que apresenta as versões intermediárias entre o DE e o jSO, utilizado neste trabalho. A primeira variante mencionada é o jDE que propôs configuração de hiper-parâmetros auto-adaptativo, que dispensa a fixação dos parâmetros que controlam a mutação e a recombinação. Já o saDE, além de propor uma nova adaptação automática dos hiper-parâmetros, alterna entre dois tipos de mutação *rand/1/bin* e *current-to-best/2/bin* a serem aplicadas de acordo com uma determinada taxa de probabilidade. Outra versão deste algoritmo também foi proposta para trabalhar com 4 versões de mutação, ao invés de duas. Na sequência, o JADE introduziu modificações mais substanciais ao propor uma nova estratégia de mutação chamada *DE/current-to-p-best*, que utiliza um grupo de melhores indivíduos para gerar novos indivíduos, e não apenas um. O que reduz as chances de o algorítimo ficar preso em um ótimo-local. Foi implementado também o controle adaptativo dos parâmetros de mutação e cruzamento, com um arquivo externo opcional, que armazena indivíduos ruins, mas que podem ser usados ocasionalmente para melhoria da diversidade populacional. Além dessas melhorias, o SHADE propôs também a utilização de uma memória histórica para os controladores dos hiper-parâmetros. O L-SHADE, por sua vez, fornece um método de redução populacional linear ao longo do processo evolucionário, que reduz o custo computacional associado à otimização realizada pelo algoritmo. O il-SHADE evoluiu o anterior ao propor um novo mecanismo para atualização da memória. Finalmente, o jSO incorpora praticamente todas essas melhorias, além de alterar os valores inicialmente atribuídos aos parâmetros de controle, e propôs a mutação *current-to-p-best-w*/1/*bin*.

4. O jSO apresentou boa performance dentro de um grande conjunto de *benchmark* onde avalia-se o desempenho para funções contínuas de forma padrão, rotacionadas e deslocadas, para várias dimensões. Considerando que os pesos a serem otimizados são contínuos, esse algoritmo constitui uma boa opção para evoluir PDTs, motivo pelo qual foi escolhido para esta função no presente trabalho.

Capítulo 4

Metodologia

4.1 Perceptron Decision Tree

As árvores de decisões tradicionais (Decision Tree, ou DT) foram largamente utilizadas em diversos trabalhos, especialmente devido a representação facilmente interpretável (Breiman et al., 1984; López-Chau et al., 2013). Muitos trabalhos que adotaram essa técnica, os quais incluem aplicações no mercado financeiro (Wu et al., 2006), na identificação de fatores associados à hipertensão (Tayefi et al., 2017), na personalização de anúncios via internet (Kim et al., 2001), dentre outros.

Todavia, muitos dos problemas reais são representados como vetores de dimensões elevadas, e a DT sofre com a conhecida "maldição de dimensionalidade", pois quanto mais recursos houver na base de dados, maior deve ser a base de dados para que o algoritmo obtenha um bom desempenho. Além disso, Liu e Tsang (2016) apontam que muitos recursos em um espaço de grande dimensão geralmente não são informativos ou são ruidosos, o que reduz o desempenho de generalização da DT e prejudica os resultados promissores para lidar com dados separáveis não lineares.

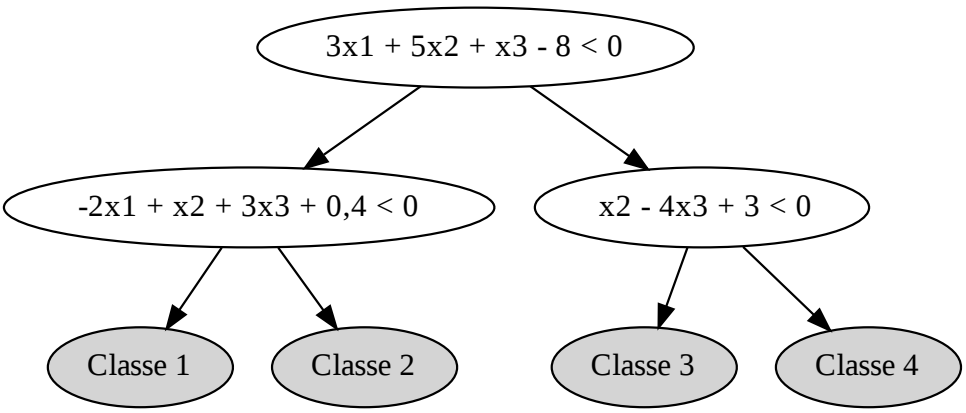
Com isso em vista, as Árvores de Decisão Perceptron (PDT), também conhecida na literatura como Árvores de Decisões Oblíquas, foram introduzidas com diferentes nomes por vários autores (Murthy et al., 1994; Lopes et al., 2012). A principal característica desta variante é que cada nó interno está associado a um hiperplano em posição geral no espaço de entrada. Essa abordagem apresentou bons resultados na classificação de problemas do mundo real (Bennett e Mangasarian, 1994; Bennett et al., 1998). Por serem paralelas em relação ao eixo, a DT é interessante quando os limites de decisão estão alinhados com os eixos dos recursos. Por outro lado, as divisões oblíquas são hiperplanos definidos por uma combinação linear dos atributos, e são mais atraentes quando os limites de decisão não estão alinhados com os eixos dos dados (Wickramarachchi et al., 2016), comum em problemas complexos. Matematicamente, cada nó de uma PDT pode ser definida conforme

a Equação 4.1.

$$\sum_{1}^{N} a_i x_i + \theta \tag{4.1}$$

onde $x = [x_1, x_2, ..., x_i, ...x_N]$ é um vetor de atributos de tamanho N, $a = [a_1, a_2, ..., a_i, ...a_N]$ são os pesos que pondera cada característica, θ é uma constante, e $x, a, \theta \in \mathbb{R}$. No que se refere aos nós folhas, esses indicam as classes do problema. A Figura 4.1 apresenta um exemplo meramente ilustrativo de uma PDT de altura $h = 3$, para melhor compreensão do leitor quanto à estrutura da árvore, e cada um dos seus componentes.

Figura 13 – Exemplo de uma PDT de classificação



Fonte: Os autores

Nesta PDT, o nó raiz é representado pelo hiperplano $3x_1 + 5x_2 + x_3 - 8 < 0$, sendo que $x = [x_1, x_2, x_3]$ é o vetor de características, $[3, 5, 1]$ é o vetor de pesos w_1 associado a x, e -8 é o valor de θ_1. Quando esta inequação resultar em $Verdadeiro$, prossegue-se pelo nó da esquerda, que tem o seguinte vetor de pesos e θ associado a ele: $w_2 = [-2, 1, 3]$ e $\theta_2 = 0{,}4$, respectivamente. Caso contrário, prossegue-se para o lado direito da árvore, onde o hiperplano correspondente tem $w_3 = [0, 1, 4]$ e $\theta_3 = 3$. O processo é feito até que um nó folha tenha sido alcançado, que corresponde à classe predita para x. Outro exemplo de PDT com as respectivas partições do plano pode ser encontrado em Santos et al. (2021).

Dessa forma, uma PDT utiliza todos os atributos em concomitância para obter a predição para uma determinada instância. Para que as predições estejam corretas, é necessário que ela consiga determinar os hiperplanos que realizem a melhor divisão do espaço de busca. Assim sendo, determinar os valores de w e θ para cada nó consiste no

problema de otimização de encontrar os melhores hiperplanos que fazem as respectivas divisões de classes, de forma a minimizar o erro de classificação de uma dada PDT. Ou seja:

$$w^*,\theta^* = \arg\min \sum_{s_i \in \eta} err(s_i), \tag{4.2}$$

onde

$$err(s_i) = \begin{cases} 1 & \text{if } y^{'} \neq y \\ 0 & \text{if } y^{'} = y \end{cases} \tag{4.3}$$

em que y é a classe esperada e y' a classe obtida pela PDT, em um contexto de classificação.

Para obtenção de w e θ, várias técnicas de otimização podem ser utilizadas. No trabalho de Bot e Langdon (2000) foi proposto a utilização de Programação Genética (GP) para a evolução de árvores de decisão oblíquas. Nesse contexto, cada indivíduo da GP é codificado como uma árvore com nós e terminais funcionais. Em Vukobratović e Struharik (2015) foi proposto o algoritmo denominado EFTI, que é baseado no algoritmo HereBoy para solução deste problema. No trabalho de Rivera-Lopez et al. (2017) foi proposto a aplicação de uma abordagem baseada em Evolução Diferencial, denominada OC1-DE, para induzir árvores de decisão oblíquas utilizando particionamento recursivo. O algoritmo utiliza a estratégia de evolução *DE/rand/1/bin*, e é aplicado com o objetivo de encontrar hiperplanos quase ótimos que serão usados como condições de teste de uma árvore de decisão oblíqua.

Considerando que no trabalho de Lopes et al. (2012) uma aplicação da versão básica do DE apresentou resultados promissores para evolução das variáveis mencionadas; e que outras versões que evoluem o DE foram apresentadas nos últimos anos, conforme detalhado no Capítulo 3, é possível inferir que soluções baseadas neste método mostram-se interessante para obtenção de w e θ.

Conforme explanado, o algoritmo jSO é uma versão que possui configuração automática de hiper-parâmetros, redução do tamanho da população de soluções ao longo do processo evolucionário, dentre diversas outras melhorias, e por isso foi proposto a utilização deste otimizador para evolução dos pesos de uma PDT, neste trabalho. Para tal, foi feita a adequação da estrutura de dados que representa um indivíduo, que nesse caso corresponde a uma PDT, de forma que o jSO é utilizado conforme proposto pelos autores, com pontuais ajustes, para evoluir os pesos contínuos da PDT, e as classes contidas nos nós folhas, que são discretas, foram determinadas via *random-setting*. Ou seja, para cada nó folha de uma PDT, é escolhido aleatoriamente uma classe dentre as possíveis. A utilização de busca local também foi considerada para otimização da parte discreta.

4.1.1 PDT para classificação

Neste trabalho, considerou-se a PDT uma árvore completamente binária balanceada. Sendo h a altura pré-definida de uma árvore, a quantidade de nós-internos e folhas pode ser obtida em função de h, correspondendo a $2^{h-1}-1$ e 2^{h-1} nós, respectivamente.

Diferentes estruturas de dados podem ser utilizadas para representar uma PDT. Neste trabalho foi adotado o formato matricial sugerido por Lopes et al. (2012), por ser um formato de simples reprodutibilidade. Nessa abordagem, três estruturas de dados são utilizadas para representar cada PDT:

1. Uma matriz de ordem $n \times 2^{(h-1)}-1$, em que o primeiro termo corresponde a quantidade de atributos da base de dados; enquanto o segundo diz respeito a quantidade de nós internos. Assim, w_{ij} é o peso associado à característica x_i, e cada coluna j corresponde a um nó interno da árvore;

2. Além disso, utiliza-se um vetor de tamanho $1 \times 2^{h-1}-1$, em que cada coluna j corresponde ao θ_j de cada hiperplano;

3. E outro vetor de mesma dimensão com as possíveis classes preditas, correspondendo aos nós folhas.

Assim, a primeira coluna da matriz de pesos, w_1, somada ao respectivo θ_1 corresponde ao nó raiz. Se a aplicação da Equação 4.1 ao vetor de atributos resultar em um valor inferior a 0, seleciona-se o nó esquerdo da árvore, que corresponde à posição $2 \times j$ da matriz de pesos e vetor de θ's. Caso contrário, caminha-se para o nó direito da árvore, posição $2 \times j + 1$ das estruturas de dados, onde j é o índice que identifica a coluna de w e θ. O nó folha correspondente encontra-se na posição p, calculada pela Equação 4.4.

$$p = \begin{cases} (k - 2^{h-2}) \times 2 & \text{se } w_k x + \theta_k < 0 \\ (k - 2^{h-2}) \times 2 + 1 & \text{caso contrário.} \end{cases} \tag{4.4}$$

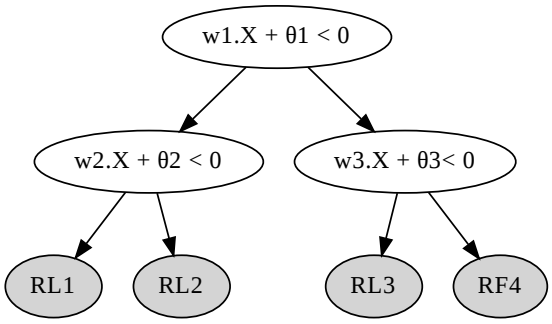
onde k é a posição na matriz w do nó imediatamente anterior ao nó folha, que corresponde ao j-ésimo nó interno da PDT, e i,$j \in \mathbb{N}^*$.

4.1.2 PDT para regressão

A proposição de PDT para problemas de regressão não parece ter sido extensivamente abordada na literatura, como o foi no contexto de classificação (Wickramarachchi et al., 2016; Barros et al., 2011, 2014). Tendo isso em vista, neste trabalho foi proposto um novo método de regressão baseado em PDT. O método proposto assemelha-se ao de classificação mencionado, exceto que cada nó folha corresponde à uma regressão linear

sobre os dados filtrados até aquela folha. Como ocorre nos problemas de classificação, é esperado que o desempenho dessa versão seja superior aos métodos de regressão tradicionais, especialmente àqueles que são transparentes e podem ser usados como intermediário para provê explicações para este tipo de problema. A Figura 14 apresenta um exemplo visual da PDT para regressão proposta.

Figura 14 – Exemplo visual da PDT proposta para problemas de regressão



Fonte: Os autores

O problema de evoluir uma PDT para regressão também pode ser modelado como um problema de otimização, cujo objetivo é minimizar o erro da predição realizada. Como os nós folhas possuem uma regressão, ao invés de uma classe, obtém-se um valor contínuo, e a partir deste, o erro quadrático médio (Mean Squared Error, ou MSE) pode ser calculado, por meio da Equação **??**. Neste caso, $err(s_i)$ pode ser reescrito como:

$$MSE = \quad \frac{1}{N}\sum_{1}^{N}(y' - y)^2 \tag{4.5}$$

em que N é a quantidade de amostras da base de dados.

Sabendo, portanto, que a PDT é um método de ML interpretável, e que possui desempenho geralmente melhor do que as tradicionais DT para problemas de classificação (Lopes et al., 2012), por fazer divisões que não são necessariamente paralelas aos eixos dos recursos, investiga-se o processo de construção de um explicador para modelos de ML caixa-preta a partir da indução de PDTs, tanto no contexto de classificação, como no de regressão. É sabido de localmente, pode-se utilizar um modelo simples para simular o comportamento de um método complexo (Ribeiro et al., 2016; Ferreira et al., 2020), como ocorre com explicadores como LIME, DT e outros. Todavia, como a PDT é um modelo transparente que normalmente apresenta melhor desempenho quando comparada a DTs tradicionais (Murthy et al., 1994; Lopes et al., 2012), e métodos lineares, como Regressão linear (), ela pode ser usada como intermediário para gerar explicações mais fidedignas

que estes primeiros. Sendo assim, a próxima seção apresenta a justificativa para utilização deste método como explicador local.

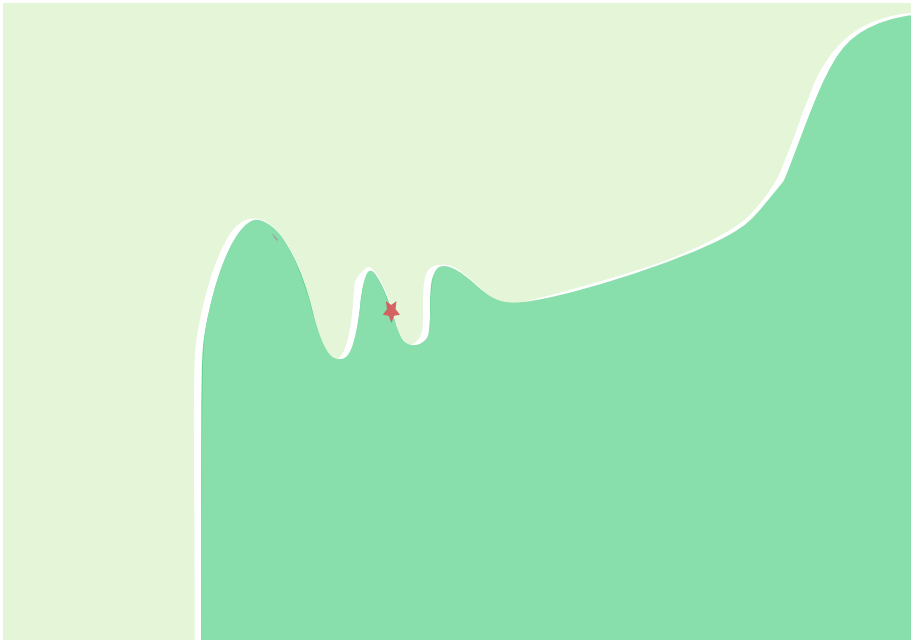
4.2 Justificativa para utilização da PDT como explicador local

A proposta dos explicadores locais é provê explicações para uma determinada instância, a qual não necessariamente será válida em um contexto global. Nesse caso, uma sub-região é determinada na vizinhança da amostra de interesse, também chamada de conjunto ruído. Muitos trabalhos, embora proponham explicações locais, não implementam critérios práticos para determinar o tamanho da sub-região em volta de x, especialmente que garanta que a sub-região poderá ser aproximada com o modelo transparente utilizado. O LIME (Ribeiro et al., 2016), por exemplo, que atualmente possui quase 10 mil citações, segundo o *Google Scholar*, não faz a implementação de mecanismo para esta finalidade. O problema associado a isso é que, métodos que são globalmente lineares, só conseguirão fornecer explicações precisas se dentro da sub-região de interesse, o caixa-preta se comportar também de maneira linear. Caso contrário, o modelo linear não conseguirá fornecer explicações precisas sobre o método opaco. As próximas figuras exemplificam este problema. Na Figura 15 encontra-se a representação de uma região fictícia gerada por um método caixa-preta e a instância de interesse em destaque, representada por uma estrela vermelha. O problema possui duas classes, sendo uma delas a região superior e mais à esquerda - em verde claro, e a segunda, a região inferior e mais à direita - em verde escuro. Na sequência, a Figura 16 apresenta a aproximação ideal por um método transparente linear, geralmente apresentadas como exemplo de como a explicação local ocorre, com uma sub-região pequena e bem definida em torno de x, e com uma reta que se ajusta muito bem à separação de classes na localidade onde x se encontra.

Entretanto, se não existir um controle de que a sub-região em torno de x é de fato linear, mesmo que haja uma função a ser aproximada muito mais complexa localmente, como apresentada neste exemplo, o método ainda tentará fazer essa aproximação por meio de um modelo muito mais simples. Com isso, ocorre uma situação de *underfitting*, e consequentemente a aproximação do método em relação ao caixa-preta tem baixa fidelidade local, ou seja, o método transparente não é capaz de realmente descrever o comportamento do método mais complexo. Na Figura 17 é mostrado aproximadamente como um modelo linear se comportaria neste exemplo, considerando a sub-região maior e onde o comportamento da função é muito complexo para ser aproximado por um modelo linear.

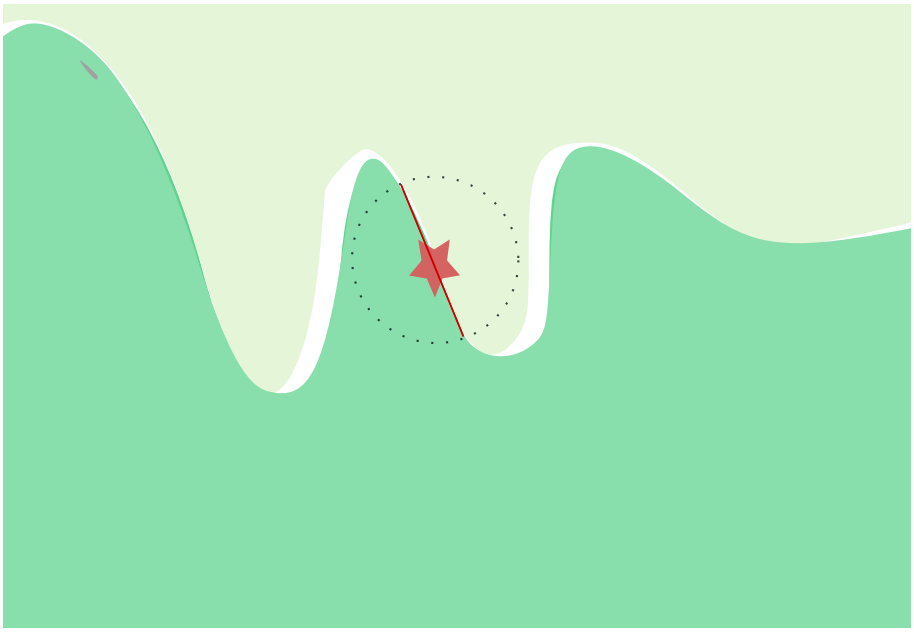
Uma maneira de resolver isso, seria a utilização de métodos que não sejam globalmente lineares e que consigam fazer essa aproximação mesmo que a sub-região

Figura 15 – Região fictícia gerada por um método caixa-preta, e em destaque a instância a ser explicada



Fonte: Os autores

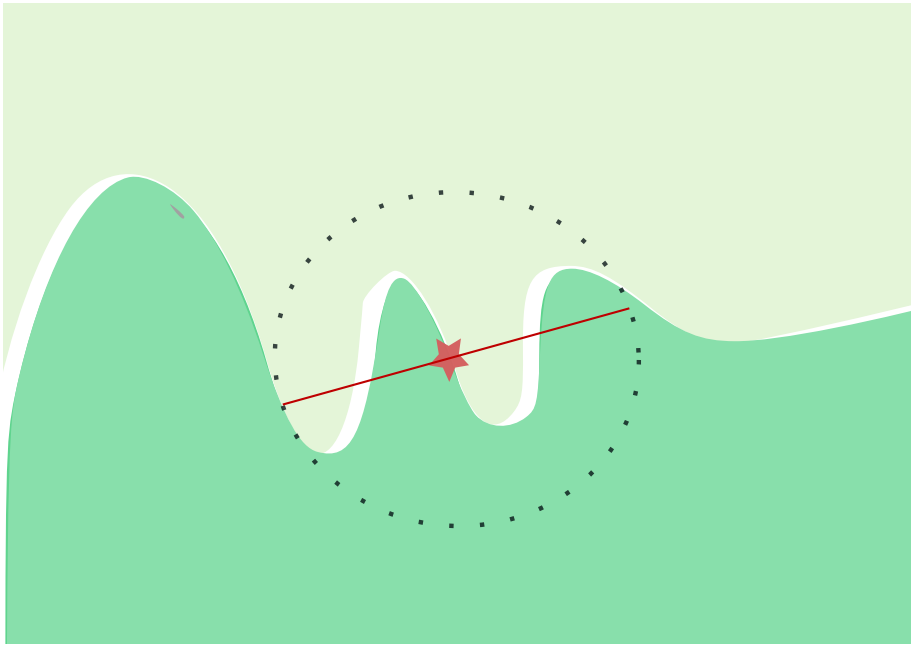
Figura 16 – Aproximação local idealizada a ser realizada por alguns métodos lineares - Vizinhança pequena e sub-região que permite aproximação via métodos globalmente lineares



Fonte: Os autores

gerada possua muitas não linearidades, como é o caso da PDT. A PDT consegue lidar muito bem com este problema pois faz várias partições no espaço de busca e, ao minimizar o erro de aproximação em relação ao caixa-preta, consegue identificar sub-regiões que possam

Figura 17 – Representação mais próxima da realidade da aproximação linear que de fato é realizada por alguns métodos lineares para a sub-região pré-definida



Fonte: Os autores

Figura 18 – Representação de uma possível aproximação gerada por uma PDT

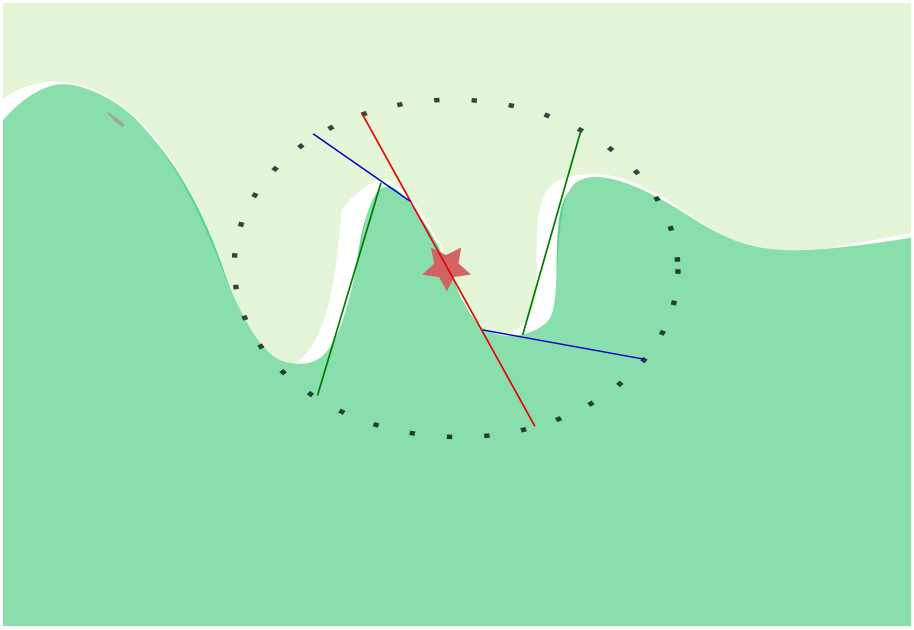


Figura 19 – Fonte: Os autores

ser interpretadas por meio dos hiperplanos oblíquos. A Figura 19 traz uma representação de uma PDT de altura $h = 4$ para subdivisão dessa mesma região. Os ramos que não fazem separação de classes foram ocultados dessa representação, pois na prática, são removidos pela operação de poda, explicada na subseção 4.3.4. Nessa representação, o segmento

de reta vermelho representa a subdivisão realizada pelo nó raiz, as retas azuis mostram a subdivisão realizada pelos nós internos, imediatamente abaixo do nó raiz. E as retas verdes mostram o último grupo de subdivisões realizadas, imediatamente anterior aos nós folhas. Uma vez que a reta vermelha é a mais próxima de x, ela é utilizada na geração das explicações locais.

Além disso, o formato da PDT permite concatenar as informações presentes tanto em modelos lineares, como a Regressão Linear, que é formada utilizando o mesmo princípio que as partições oblíquas, como aquelas presentes numa DT, produzindo portanto uma explicação mais completa para o usuário. Com base nesse princípio, na próxima seção apresenta-se o PDTX, que é um explicador local que gera explicações utilizando as PDTs evoluídas com o jSO.

4.3 PDTX: um novo explicador baseado na indução de PDTs

A forma de gerar explicações locais utilizada pelo PDTX é semelhante ao de outras abordagens da literatura (Ferreira et al., 2020), (Ribeiro et al., 2016) em que um método transparente é utilizado para fornecer informações sobre o processo de tomada de decisão de sistemas caixa-preta.

4.3.1 Aproximação local

Seja um método opaco g pré-treinado, o qual deseja-se obter a interpretação do processo de tomada de decisão realizado para se obter a saída y associada a x, em que $x \in \mathbb{R}^n$ é um vetor de características, o qual deseja-se explicar o processo de tomada de decisão de $g(x)$, em que $g : \mathbb{R}^n \to \mathbb{R}$. O processo consiste em aproximar localmente um método transparente p, onde $p : \mathbb{R}^n \to \mathbb{R}$, nas redondezas de x, de forma que seja possível simular o comportamento do caixa-preta por meio de um método caixa-branca. Uma vez que isso é feito, é possível gerar explicações para $g(x)$ por meio de $p(x)$. O processo é realizado da seguinte maneira. Gera-se um conjunto ruído η de tamanho m em torno de x. A partir do caixa-preta pré-treinado, obtém-se as respectivas saídas para o conjunto ruído definido. Em seguida, utiliza-se p para gerar as mesmas saídas que g para essa vizinhança. Assim, o objetivo do PDTX é encontrar uma função $f^* : \mathbb{R}^n \to \mathbb{R}$, tal que f^* é o argumento que minimiza a distância entre as previsões realizadas por por g e p para η. Ou seja:

$$f^* = \arg\min_{s_i \in \eta} \sum_{i}^{N} D[p(s_i) - g(s_i)] \tag{4.6}$$

Onde,

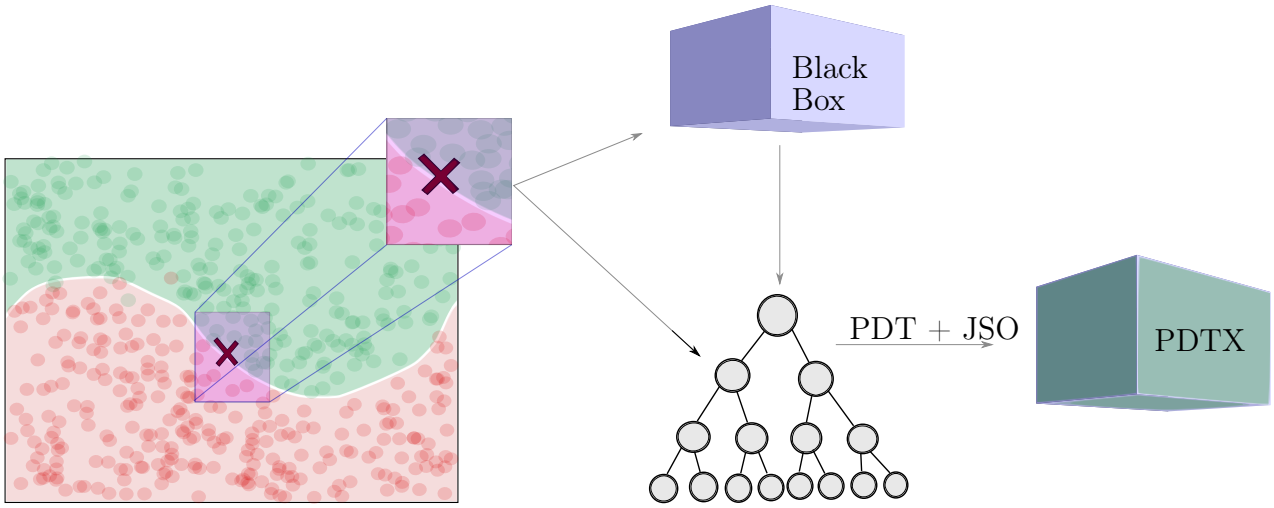
- s_i é uma amostra i do conjunto ruído η;
- $p(s_i)$ é a predição realizada por p para a amostra s_i;
- $g(s_i)$ é a predição realizada por g para a amostra s_i;
- D é a diferença entre a predição realizada por p e g;
- E f^* a função que minimiza o somatório das distâncias entre as predições de p e g para todo o conjunto η.

Explicadores como o LIME utilizam modelos lineares simples, como Regressão Linear, com base na distância euclidiana entre as amostras de η e x, para fazer essa aproximação aplicada a dados tabulares. Na GPX foi utilizado programação genética para esse mesmo propósito. Aqui, sabendo que a PDT é completamente descritível e interpretável (Zázvorka, 2020), e que apresentou bons resultados quando aplicada em problemas de classificação (Zhang e Suganthan, 2014), foi proposta a sua utilização como método transparente p.

Utilizando-se da nomenclatura adotada pela xAI, o processo de aproximar as predições de g por meio de p pode ser chamado de fidelidade (Ver seção 5.1.3). No contexto de classificação, essa métrica pode ser interpretada como a medida de acurácia de $p(\eta)$ em relação a $g(\eta)$, conforme Equação 4.2. Já para regressão, diferentes métricas de minimização de erro podem ser utilizadas para mensurar a fidelidade. Nesse trabalho, adotou-se o Erro quadrático médio (Equação 4.5) como medida de aproximação para problemas de regressão.

Assim, no PDTX, p é obtido a partir da evolução de uma PDT com a utilização do jSO, pois dado que g produz um conjunto de predições para η, utiliza-se este otimizador para que a PDT resultante consiga gerar essas mesmas predições, ou seja, para minimizar o erro de predição da PDT em relação ao caixa-preta proposto. Uma vez que p consiga simular g, em torno de x, pode-se utilizar p para extrair explicações para g. A visão geral deste processo pode ser vista na Figura 20. Essa figura traz uma visão geral do PDTX em que apresenta-se (i) a geração de uma vizinhança em torno de uma amostra a ser explicada, (ii) a obtenção das previsões do modelo de caixa preta para o conjunto de ruído e (iii) aproximação das previsões da PDT para essas predições com o jSO. Com a árvore gerada, é possível explicar o caso de teste.

Figura 20 – Visão geral do PDTX



Fonte: Santos et al.(2021, adaptado)

Para maior clareza, considere o seguinte exemplo. Seja uma base de dados formada por dois atributos x_1 e x_2 que juntos mapeiam duas classes, Classe 1 - representada na Figura 21 pelas bolinhas roxas e Classe 2 representadas pelos quadradinhos azuis. E seja x a instância de interesse cuja predição deve ser explicada, escolhida propositalmente na região de interseção das duas classes. Na Classe 1 encontram-se pontos gerados aleatoriamente seguindo uma distribuição normal com média μ 0,0 e desvio padrão σ igual a 0,4, tanto pra x_1, como para x_2; e na Classe 2, foram gerados dados seguindo as mesmas características dos dados da Classe 1, mas com $\mu = 1.0$ e $\sigma_{x_1} = 0{,}3$ para x_1 e $\sigma_{x_2} = 0{,}5$ para x_2. No total foram geradas 1000 amostras para representar a região mapeada pelo caixa-preta. [1]

O ruído η, em destaque na Figura 21, é gerado localmente em torno da amostra de interesse. Utilizando g, obtém-se as saídas para $g(\eta)$, conforme apresentado na Figura 21, as quais são utilizadas pelo jSO para evoluir uma PDT para que ela consiga produzir as mesmas saídas que g nessa vizinhança. Neste exemplo, g é um Perceptron de Múltiplas Camadas (MLP ou Multilayer Perceptron), com 100 neurônios na camada interna. [2]

4.3.2 Geração das explicações

Embora seja completamente descritível e interpretável, a PDT sozinha não consegue fornecer tantas informações caso procedimentos adicionais não sejam realizados. A Figura 21 apresenta a PDT gerada para o exemplo de classificação aqui tratado. Como pode ser observado, todas as informações que podem ser obtidas por meio dela não estão disponíveis por simples inspeção visual. Entretanto, por meio dessa mesma árvore

[1] Dados gerados por meio da biblioteca Numpy da linguagem de programação Python 3. Gráficos gerados via bibliotecas *Seaborn* e *Matplotlib*, dessa mesma linguagem.

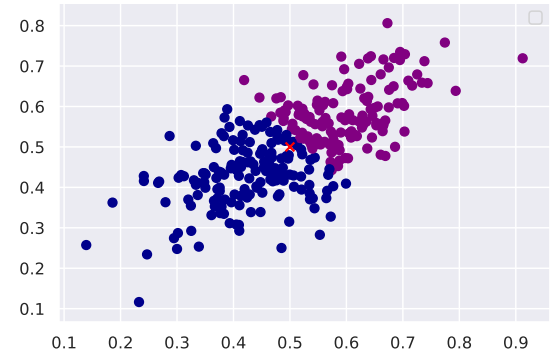
[2] O modelo g foi gerado utilizando a biblioteca *Sklearn* do *Python 3*.

Figura 21 – Exemplo do processo de indução de uma árvore PDT para explicações locais para uma base fictícia

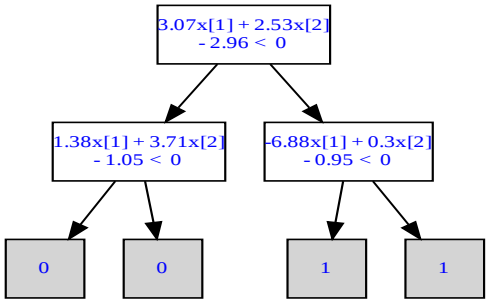
(a) Base de dados original mapeada pelo caixa-preta

(b) Destaque para conjunto ruído e amostra x juntos à base original

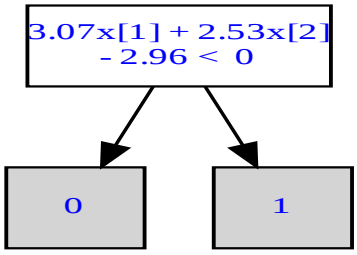
(c) Classificação de η gerada por g



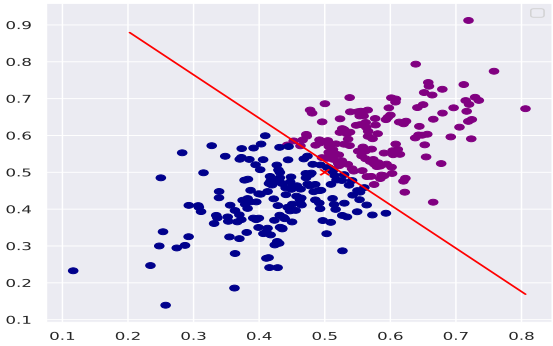
(d) PDT resultante pela otimização feita pelo JSO



(e) PDT após simplificação (operação de poda)



(f) Representação da partição realizada pela PDT ao simular o MLP em η



Fonte: Os autores.

é possível auferir detalhes sobre a importância local das características, os limites de decisões associados a cada uma delas, quais foram os atributos mais importantes para cada divisão realizada por ela, e as regras geradas para a predição em específico, além de modificações locais que podem ser realizadas dentro dos limites de decisões para se obter outras classificações, se isso for possível. Assim, o seguinte processo para geração das explicações pode ser realizado:

1. Obtém-se o caminho gerado do nó raiz até a folha, seguido por x para obtenção da respectiva saída. Nesse processo, todos os hiperplanos percorridos da raiz aos nós folhas são armazenados, e a indicação para qual dos ramos de cada nó houve prosseguimento para a predição;

2. Realiza-se a poda do caminho para remover hiperplanos que não são relevantes para a geração da saída (Ver subseção 4.3.4);

3. Para problemas de classificação, obtém-se o hiperplano divisor de classes mais próximo de x.

   Ao encontrar os hiperplanos de separação de classes, os quais delimitam a região onde x se encontra, é possível não somente rastrear um caminho semelhante ao realizado pelo caixa-preta para realização das predições, mas também obter informações importantes sobre a sub-região de interesse, as quais são detalhadas nas próximas subseções. Neste contexto, avaliar o hiperplano mais próximo de x é similar a "dar um zoom"na sub-região em que x se encontra. E dessa forma, semelhante ao que ocorre com modelos lineares, é possível extrair informações sobre x por meio dos coeficientes deste hiperplano.

4. Para problemas da regressão, os coeficientes da regressão são utilizados nesse processo.

4.3.3 Obtenção do hiperplano mais próximo

O hiperplano mais próximo da instância de interesse é obtido com base no seguinte passo-a-passo:

1. Cada hiperplano utilizado no caminho para obtenção predição da amostra x é obtido ao percorrer a PDT final e fazer a predição de x. Dada uma árvore de tamanho h, haverá no máximo $h - 1$ hiperplanos do nó raiz até a folha;

2. Conforme definido na Equação 4.1, cada hiperplano é uma combinação linear de todos os atributos, podendo ser reescrito como $\sum_{i=1}^{N} a_i \times x_i + \theta$;

3. Para cada um desses hiperplanos, calcula-se o ponto Q mais próximo de x, em que Q o ponto que encontra-se no hiperplano de dimensão $N - 1$ que liga o hiperplano j à instância x. Q é obtido por meio da Equação 4.11. Em detalhes:

Seja o ponto $P = (x_{P_1}, x_{P_2}, ..., x_{P_N})$ e o hiperplano $\alpha : a_1x_1 + a_2x_2 + ... + \theta = 0$ sobre o mesmo espaço cartesiano. A menor distância possível entre P e α é dada por r, em que r é a reta que passa por P e é perpendicular a α. Assim sendo, são válidas as equações paramétricas de forma:

$$x_i = x_{P_i} + a_i t \tag{4.7}$$

para algum escalar t. Se Q é o ponto de interseção entre r e α, com $Q \in r$ então:

$$Q = (x_{P_1} + a_1 t, x_{P_2} + a_2 t, x_{P_3} + a_3 t, ...) \tag{4.8}$$

Como Q também pertence a α, então a seguinte relação também se aplica:

$$\sum_{i=1}^{N} a_i(x_{P_i} + a_i t) + \theta = 0 \tag{4.9}$$

$$t = -\frac{\sum_{i=1}^{N} a_i x_{P_i} + \theta}{\sum_{i=1}^{N} a_i^2} \tag{4.10}$$

Portanto, tem-se que cada coordenada i de Q é definida por:

$$Q_i = \left(x_{Pi} - \frac{\sum_{i=1}^{N} a_i x_{P_i} + \theta}{\sum_{i=1}^{N} a_i^2} \times a_i \right) \tag{4.11}$$

Hiperplanos que não são divisores de classes são descartados por meio da operação de poda, pois não oferecem informações suficientes para explicar o comportamento da instância de interessante. Com base nos hiperplanos remanescentes, e na Equação 4.12, calcula-se qual o Q_j com menor distância da amostra x. A partir daí, utiliza-se o hiperplano j correspondente para gerar as explicações a respeito de x.

$$d(Q,\alpha) = \sqrt{\left(x_{Pi} - \frac{\sum_{i=1}^{N} a_i x_{P_i} + \theta}{\sum_{i=1}^{N} a_i^2} \times \sum_{i=1}^{N} a_i^2 \right)} \tag{4.12}$$

Na sequência, gera-se novos pontos, resultante da soma de um ϵ positivo a Q_i, e obtém-se a predição destes novos pontos. Verifica-se se a predição encontrada para cada um desses novos pontos é distinta da predição de x, a fim de se determinar se a contribuição de cada recurso é positiva ou negativa em relação à predição inicial.

4.3.4 Operação de poda

Para árvores de classificação, verifica-se se todos os hiperplanos deste caminho são de fato importantes para a predição. Considerando-se que o jSO é heurístico, a árvore

gerada pode ter hiperplanos sub-utilizados, os quais são removidos com a operação de poda. São considerados irrelevantes aqueles hiperplanos que quando removidos da PDT retornada, não gere efeito sobre a saída gerada.

Para cada hiperplano contido no caminho obtido, troca-se o sinal de todos os coeficientes deste hiperplano e calcula-se a fidelidade obtida com o novo hiperplano. Essa alteração faz com que o lado oposto da árvore seja seguido em relação ao caminho normal. Se ao fazer isso, mesmo assim não houver redução da fidelidade local, significa que este hiperplano está separando a mesma classe, e se for removido, não alterará a predição. Dessa forma, todos os hiperplanos que se mostrarem não essenciais são removidos do caminho obtido e não desconsiderados para geração das explicações locais. Na Figura 21 esta operação ocorre sobre a árvore resultante pela aplicação do JSO, apresentada na Figura 21d, resultando na árvore mostrada na Figura 21e.

4.3.5 Obtenção da importância das características

Conforme Zázvorka (2020), os coeficientes de um hiperplano de uma PDT fornece informações relevantes sobre a importância de cada atributo. Com base neste conceito, e no apresentado em Nathans et al. (2012); Kasza e Wolfe (2014); Tompkins (1993), construiu-se a seguinte estratégia para definição dessa importância:

1. Obtém-se os valores em módulo dos coeficientes do hiperplano mais próximo de x, no caso da classificação, e os coeficientes da regressão linear, para problemas de regressão;

2. Para determinar se o impacto da característica é positivo ou negativo, nos problemas de classificação, soma-se um $\epsilon > 0$ a cada atributo de x, e realiza-se a predição utilizando a PDT evoluída para verificar se a classificação original foi alterada. Se houver alteração, o impacto observado é negativo. Caso contrário, é positivo ou nulo, a depender do valor do módulo do coeficiente associado à característica observada. Nos problemas de regressão, a avaliação do coeficiente original traz a relação direta da amplitude de importância de cada atributo, e o sinal associado cada uma delas.

4.3.6 Obtenção dos limites de decisão

Também é possível a obtenção dos limites de decisão de cada atributo por meio da PDT. Ou seja, as condições mínimas que devem ser superadas para cada atributo para que a predição mude. A partir do hiperplano mais próximo da amostra de interesse, considerando o contexto de classificação, obtém-se os limites de decisão para cada dimensão

de x mantendo fixo os valores das demais dimensões e isolando x_i. Reescrevendo a equação geral do hiperplano mais próximo, tem-se que:

$$x_i < \frac{-\left(\sum_{k=1,k\neq i}^{N} a_k x_k + \theta\right)}{a_i} \tag{4.13}$$

O mesmo raciocínio é aplicado à Regressão Linear resultante do processo de aplicação da PDT para problemas de regressão.

A Figura 22 apresenta a importância dos atributos, calculados conforme subseção 4.3.5, juntamente com os limites de decisões de cada uma delas, mostrados nesta seção para o problema de classificação apresentado na Figura 21. Por meio deste gráfico é possível notar que a classificação gerada denota a Classe 0, mas ambas as características possuem peso negativo para esta classe. Ou seja, ao se aumentar o valor de cada uma dessas características, a probabilidade em se classificar essa classe são reduzidas. Além disso, a amplitude de x_0 é maior do que a amplitude de x_1, indicando que esse atributo impacta mais as probabilidades de saídas, que x_1. Os limites de decisões para a Classe 0 também são dados para cada uma dessas características. Fora deste intervalo a classificação muda para Classe 1, ou estão fora dos limites da vizinhança local pré-definida.

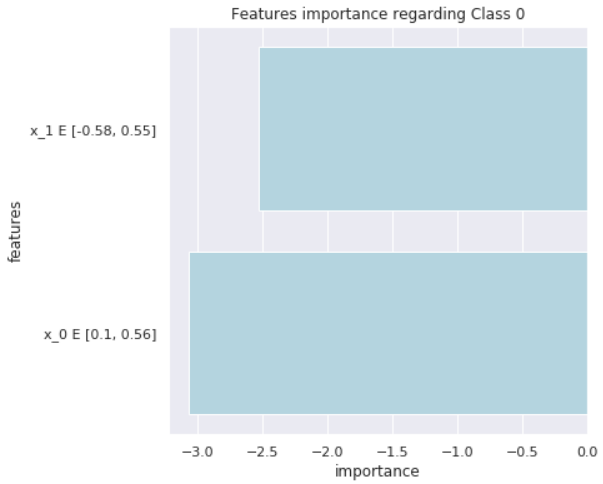


Figura 22 – Importância das características e limites de decisão

4.3.7 Obtenção das regras de classificação e regressão

Tanto para problemas de classificação, como para aqueles de regressão, obtém-se o caminho do nó-raiz ao nó-folha para geração das explicações. A diferença principal entre metodologia aplicada em relação a esses dois tipos de problema, é que no caso de classificação, as explicações são geradas a partir do hiperplano mais próximo de x, enquanto na regressão utiliza-se a função de regressão correspondente ao nó folha da predição. Dessa

forma, a obtenção deste caminho permite averiguar todas as regras utilizadas em conjunto. Obtendo-se os limites de decisão para todos os hiperplanos nesse caminho, a região de interesse é delimitada. Neste exemplo, como houve somente um hiperplano remanescente, a avaliação é feita sobre ele. Caso haja vários hiperplanos, as informações sobre um mesmo atributo são condensadas. A regra gerada para classificação gerada foi:

$$\textbf{IF } (x_1 \leq 0.55 \textbf{ AND } x_2 \leq 0.56) \textbf{ THEN } \textit{Class 0} \tag{4.14}$$

Além disso, o PDTX fornece informações gerais sobre os atributos em relação a outras sub-regiões dentro vizinhança de x mapeadas pela PDT. Essas informações ajudam o especialista na tomada de decisão, já que além de conseguir entender o caminho usado para a predição, consegue-se ainda visualizar outras alternativas que permitiriam chegar às demais predição no raio da vizinhança local.

Figura 23 – Regras geradas pelo PDTX - Classificação

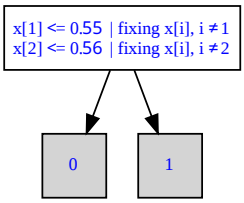
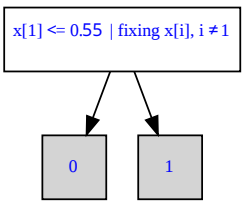
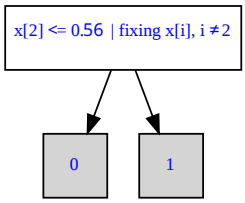


Figura 24 – Regras geradas para x_1



No apresentado, caso desejasse alterar a predição de Classe 0 para Classe 1, se o problema real permitir a manipulação das características envolvidas, pode-se manter x_2 fixo em 0,5, e alterar o x_1 para um valor maior que 0,55, o que faria com que a condição apresentada na primeira linha seja falsa, e assim o ramo direito da árvore seria escolhido. Alternativamente, pode-se manter x_1 fixo em 0,5, e atribuir a x_2 um valor superior a 0,56,

Figura 25 – Regras geradas para x_2



o que também selecionaria o ramo direito deste nó. Ou seja, pode-se olhar individualmente para cada recurso a alteração possível para percorrer todos os ramos da árvore. A Figura 23 sumariza essas alternativas para todas as variáveis, enquanto as Figuras 24 e 25 tratam esses caminhos individualmente para cada preditor.

Para obter todas as regras, para cada um dos hiperplanos contidos na PDT, calcula-se todos os limites de decisão para cada uma das variáveis, conforme abordado na subseção 4.3.6. Observe que nessa abordagem são utilizados não somente os hiperplanos contidos no caminho das predições, mas todos. Dessa forma, além de obter as informações individuais para a instância de interesse, é possível também obter informações sobre toda a vizinhança, e pode-se visualizar outros caminhos que podem ser seguidos ao se fazer modificações sobre cada uma das características.

Embora o exemplo apresentado seja simples, e passível de aproximação por um modelo globalmente linear, esse tipo de informação fornecida pelas Figuras 23, 24 e 25, é especialmente útil para problemas complexos, pois, além de conseguir informar quais as variáveis mais importantes e os limites associados a elas, como outros métodos como LIME o fazem, apresenta ainda variações presentes na subregião de interesse, que não são mostradas por este outro explicador. Ou seja, o tipo de informação apresentada extrapola as regras apresentadas em 4.3.7. De fato, o PDTX consegue fornecer informações semelhantes às apresentadas pelo LIME e por uma DT, em conjunto.

Para problemas de regressão, o PDTX gera regras a partir de uma PDT que contém Regressões Lineares nas folhas. Dessa maneira, a estrutura da árvore, que vai do nó raiz ao nó que antecede a folha, é utilizada para delimitar a região em torno de x, de forma que a Regressão Linear consiga aproximar as predições de uma sub-região complexa, usando uma estrutura muito mais simples. As regras resultantes do processo são geradas a partir da Equação 4.3.6 concatenadas às regras geradas pela Equação 4.3.7.

4.4 Estudo da vizinhança local

Para que explicadores locais baseados em métodos transparentes consigam gerar explicações corretas, confiáveis e robustas, é necessário que ele simule o comportamento do método original em volta da instância de interesse (explicação local). Uma alternativa comumente utilizada para entender este comportamento é gerar uma vizinhança local em torno da amostra a ser explicada e obter as saídas correspondentes geradas pelo ML e utilizar um modelo interpretável que reproduza este mesmo comportamento. São exemplos de explicadores que usa essa técnica o LIME, e as extensões provenientes dele (Botari et al., 2020; Zafar e Khan, 2019), a GPX (Ferreira et al., 2020), e a própria PDTX (Santos et al., 2021). Com isso, é preciso que a vizinhança seja representativa o suficiente para capturar as especificidades da região. Isso significa que amostras semelhantes, escolhidas na mesma região, produzirão explicações semelhantes, conceito conhecido como robustez ou estabilidade. Na subseção 5.1.3 detalha-se como isso é avaliado quantitativamente. Ao mesmo tempo, o método transparente utilizado para explicação, mesmo que tenha baixa complexidade, deve ser capaz de reproduzir o comportamento estudado e capturar as minúncias da subregião de interesse. Conceito conhecido como fidelidade (Molnar, 2019).

Vários explicadores fazem uso de uma vizinhança local, neste trabalho também referida como conjunto ruído ou perturbações da instância de interesse, comumente gerada aleatoriamente para ajustar o modelo transparente e assim provê as explicações sobre o caixa-preta (Santos et al., 2021; Ribeiro et al., 2016; Ferreira et al., 2020; Goodfellow et al., 2014; Jia et al., 2019). Todavia, essa técnica pode gerar instabilidade nas explicações, já que são gerados pontos diferentes a cada execução do explicador. Essa divergência pode fazer com que as explicações geradas utilizando as mesmas configurações de hiper-parâmetros, para uma mesma instância, sejam distintas, o que não é desejável, pode reduzir a confiabilidade do explicador gerado, e, consequentemente, impedir a implantação desses sistemas para aplicações críticas, como da área de saúde. No trabalho de Visani et al. (2020b) foram propostos dois indicadores que avaliam a estabilidade do LIME, nomeados pelos autores como Índice de estabilidade de variáveis (VSI) e Índice de estabilidade de coeficientes (CSI). O VSI mede a concordância dentre o subconjunto de variáveis classificadas como mais importantes, enquanto o CSI testa a similaridade entre os coeficientes para um mesmo atributo, em chamadas repetidas de LIME. Esses dois estimadores em conjunto permitem avaliar a estabilidade desse método.

Diferentes estratégias foram propostas para resolver o problema de estabilidade nas extensões do LIME. No trabalho de Zafar e Khan (2019), os autores substituíram a perturbação aleatória, por um cluster hierárquico aglomerativo para agrupar os dados de treinamento, e em seguida, aplicaram o método de clustering K-Nearest Neighbor (KNN) para determinar o cluster relevante da nova instância que está sendo explicada. Na sequência, um modelo linear é treinado sobre o cluster selecionado para gerar as explicações.

Embora esta técnica possa reduzir o problema da instabilidade das explicações, a aplicação do explicador limita-se a sistemas que a base utilizada para treinamento esteja disponível. Além disso, é necessário que haja uma quantidade significante de amostras na base de treino, que estejam próximas à instância de interesse.

Outrossim, Zhou et al. (2021) propuseram o S-LIME, que utiliza uma estrutura de teste de hipóteses baseado no teorema do limite central para determinar o número de pontos de perturbação necessários para garantir a estabilidade do explicação. Este trabalho, junto ao trabalho de Botari et al. (2020), indica que uma amostragem aleatória suficientemente grande pode reduzir o problema da instabilidade das explicações.

Com tudo isso em vista, este trabalho traz o estudo de três diferentes métodos para definição de uma vizinhança local para ser utilizada pelo PDTX. Considera-se que um conjunto de pontos adequado para este propósito, deve ser representativo, no sentido de conseguir mapear adequadamente o comportamento apresentado na vizinhança de interesse, bem como possuir aspectos de estabilidade que, mesmo que possua aleatoriedade, ainda consiga descrever o comportamento de interesse.

Foram consideradas as seguintes técnicas: aleatória baseada em uma distribuição normal ou gaussiana, utilizada pelo LIME (Ribeiro et al., 2016; Zafar e Khan, 2019); amostragem baseada na sequência de Hammersley (Wong et al., 1997) e amostragem por hipercubo latino (Loh, 1996) . Tais distribuições foram pré-selecionadas empiricamente a partir de um grupo maior de amostragens: Hipercubo Latino, Hipercubo latino centralizado, Hipercubo latino máximo optimizado, Amostragem de hipercubo otimizada por correlação, Amostragem Halton e Amostragem via sequencia de Hammersley.

Além da amostragem gaussiana, foi observado que a amostragem por hipercubo latino tradicional aparece frequentemente como opções em alguns trabalhos (Aldeia e de França, 2021, 2022), em vista disso, ela também foi considerada como opção para aplicação neste trabalho. Além das mencionadas, as sequências de Halton e Hammersley costumam ser mencionadas na literatura como duas sequências de baixa discrepância bem conhecidas, que foram usados para integração quase-Monte Carlo em alguns trabalhos (Wong et al., 1997), sendo a Hammersley derivada da sequência de Halton, e testes empíricos mostram que ambas possuem comportamento próximo de determinístico. De fato, elas costumam ser chamadas de amostragem "quase-aleatória"por alguns autores(Wong et al., 1997). Uma vez que a segunda evolui da primeira, ela também foi considerada neste trabalho como opção de amostragem a ser utilizada para treino do PDTX. Com base nesse estudo, é possível definir qual a estratégia de vizinhança mais adequada a ser adotada para geração de explicações robustas pelo PDTX.

A PDT obtida nesse estudo é a melhor árvore resultante da aplicação do algoritmo evolutivo jSO sobre a vizinhança local. Uma vez que este método não é determinístico, diferentes árvores podem ser geradas em diferentes execuções. Entretanto, para uma

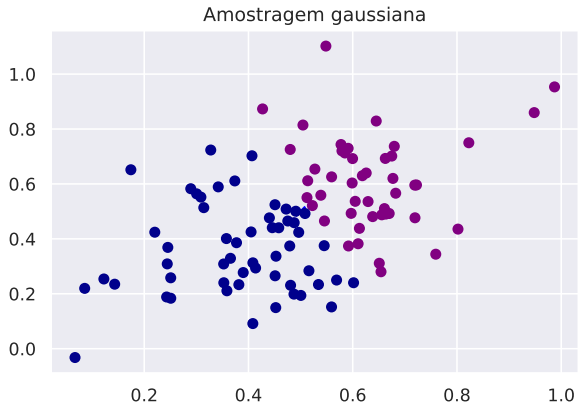
vizinhança com características determinísticas, executando-se uma quantidade de gerações tal que o permita convergir para o ótimo global, ou para as proximidades desse, faz com que diferentes execuções geram resultados muitos semelhantes. Dessa maneira, mesmo sendo empregada uma estratégia meta-heurística para evolução dos pesos da PDT, pode-se gerar resultados estáveis a partir desta técnica por meio de uma vizinhança estável. Assim, esse estudo permite definir a estratégia de amostragem e o conjunto de configurações adequadas para que as explicações geradas sejam coerentes para diferentes execuções.

4.4.1 Amostragem aleatória a partir de uma distribuição normal

Nesse tipo de amostragem, amostras aleatórias são escolhidas a partir de uma distribuição normal (ou gaussiana). Este processo é muito utilizado na prática, pois vários eventos da natureza podem ser representado por essa distribuição, que também pode ser chamada de curva de sino por causa de sua forma. O LIME (Ribeiro et al., 2016) e várias extensões desse explicador (Botari et al., 2020) utilizam essa distribuição para determinar a vizinhança local, para a partir dessa, produzir as explicações. Em Santos et al. (2021) a amostragem foi realizada a utilizando.

No $\mathbb{R}^2$, a aparência dessa distribuição é semelhante a um sino, e aproximadamente 68% dos dados dessa distribuição estarão a uma proximidade de um desvio padrão σ da média μ, 95% deles estarão a uma distância de até 2σ, e 99% dos dados a uma distância de cerca de 3σ de μ (Farber, 2015). Isso significa, que a grande maioria dos pontos amostrados utilizando essa técnica estarão necessariamente próximos da média, ou do ponto de interesse. A Figura 26 permite visualizar um exemplo de pontos gerados aleatoriamente seguindo esta distribuição.

Figura 26 – Amostragem gaussiana



Fonte: Os autores

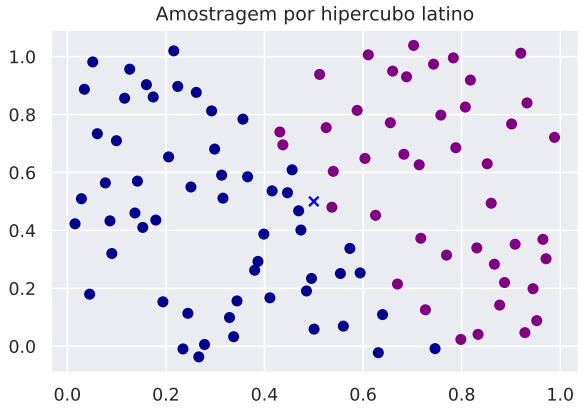
Como a maioria dos pontos estão muito próximos de x, para que se consiga cobrir bem toda a sub-região de interesse, é necessário que a quantidade de pontos gerados seja

suficiente para tal, caso contrário, a utilização de poucas amostras podem fazer com que o explicador utilizado não consiga perceber todas as variações contidas na vizinhança adotada.

4.4.2 Amostragem por Hipercubo Latino

De acordo com Hora e Helton (2003), a técnica de amostragem conhecida como hipercubo latino (ou Latin Hypercube Sampling – LHS), foi proposta por McKay, Beckman e Conover em 1979 , como uma alternativa à amostragem aleatória simples em experimentos computacionais. Essa técnica difere da amostragem aleatória simples, especialmente porque ela estratifica simultaneamente todas as dimensões de entrada. Segundo Shields et al. (2015), a amostragem por hipercubo latino é provavelmente o método de amostragem aleatória, baseado em Monte Carlo, mais utilizado para quantificação de incerteza e análise de confiabilidade, empregado em quase todos os campos da ciência computacional, engenharia e matemática. Os autores ressaltam ainda que este método de amostragem é especialmente poderoso e útil graças principalmente às propriedades identificadas por Stein (Shields e Zhang, 2016) que mostraram que o LHS tem o efeito de filtrar a variância associada aos componentes aditivos de uma transformação. Embora seja uma técnica de amostragem aleatória, essa técnica é significativamente representativa, já que o espaço amostral de cada uma das dimensões do problema é dividido em m intervalos equiprováveis, e um ponto é escolhido em cada um destes intervalos. A Figura 49 apresenta um exemplo de amostragem seguindo essa técnica no $\mathbb{R}^2$.

Figura 27 – Amostragem por Hibercubo Latino



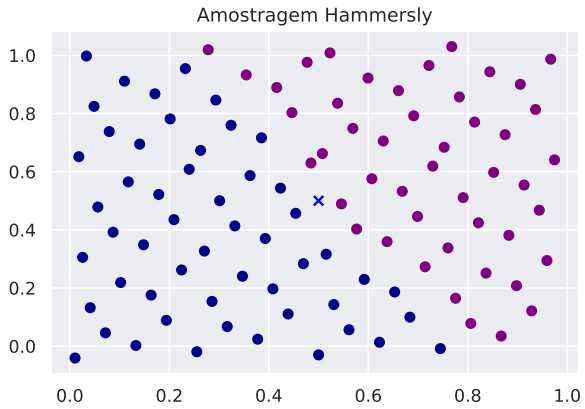
Fonte: Os autores

4.4.3 Amostragem Hammersley

O conjunto de pontos de Hammersley (Hammersley, 1960), ou sequência de baixa discrepância, é outra opção a ser considerada neste contexto no que diz respeito à

estabilidade. Trata-se de uma fórmula determinística que gera um padrão de amostragem uniformemente distribuído e de aparência estocástica, com baixo custo computacional (Wong et al., 1997). Esse tipo de sequência, também é chamada de sequência quase aleatória, pois comumente são aplicadas para substituir números aleatórios uniformemente distribuídos. Segundo Cauwet et al. (2020), técnicas de amostragem como Halton (Wong et al., 1997), Hammersley (Wong et al., 1997) ou amostragem de hipercubo latino (Loh, 1996), visam distribuir os pontos de forma mais diversificada do que a amostragem aleatória independente. Além disso, Kalagnanam e Diwekar (1997) mostraram que os pontos da sequência de Hammersley têm boa uniformidade multidimensional.

Figura 28 – Amostragem Hammersley



Fonte: Os autores

As Figuras 26, 49 e 28 apresentam um exemplo visual bidimensional de cada uma dessas distribuições, respectivamente.

4.4.4 Adequação do tamanho sub-região em torno de x

Um detalhe que pode afetar o PDTX e outros métodos de explicação local é quando dentro da sub-região ao redor de x não há alteração da saída gerada pelo caixa-preta. Ou seja, todas as amostras dessa sub-região próxima de x geram a mesma resposta. Nesse caso, não é possível determinar a importância dos atributos, pois o hiperplano gerado terá os coeficientes associado a x zerados, e o termo independente não consegue fornecer informações suficientes para essa análise. De fato, quando isso ocorre, pequenas mudanças realizadas sobre a amostra não gerarão efeito na saída.

Porém, com o intuito de ainda sim conseguir fornecer explicações que direcionem a tomada de decisão pelo usuário, para definição da vizinhança local do PDTX foi adotado a seguinte definição: O conjunto ruído deve ser gerado em torno da instância de interesse, considerando um tamanho de 10% do desvio padrão da base de treino, ou deve ser igual a 0,1 caso a base de treino não esteja disponível. Verifica-se se dentro dessa sub-região

as saídas geradas pelo caixa-preta são iguais. Caso seja, o tamanho da sub-região é acrescido em 5%, aumentando para 15% do desvio padrão da base de treino, ou 0,15. E assim sucessivamente. Esse processo ocorre até que pelo menos duas saídas geradas sejam diferentes, ou que se alcance no máximo 30% do desvio padrão original (valor definido empiricamente, considerando-se que deseja-se fazer uma explicação local, e que parece não haver uma definição na literatura que limite o tamanho dessa). A ocorrência dessa última situação, sem que haja saídas distintas, implica que não há mudanças locais que alterem a saída localmente, e que todos os resultados têm a mesma importância, ou possuem os coeficientes nulos. A Figura 4.4.4 apresenta um exemplo fictício deste processo para a mesma região fictícia apresentada nos exemplos anteriores. Nessa figura, x encontra-se em outra localidade, e o tamanho da sub-região original não alcança outra classe, de forma que nenhuma alteração dentro dessa sub-região alteraria a saída gerada pelo caixa-preta. Sendo assim, a sub-região foi aumentada, até que a classe adjacente fosse alcançada, e assim a explicação pudesse ser gerada.

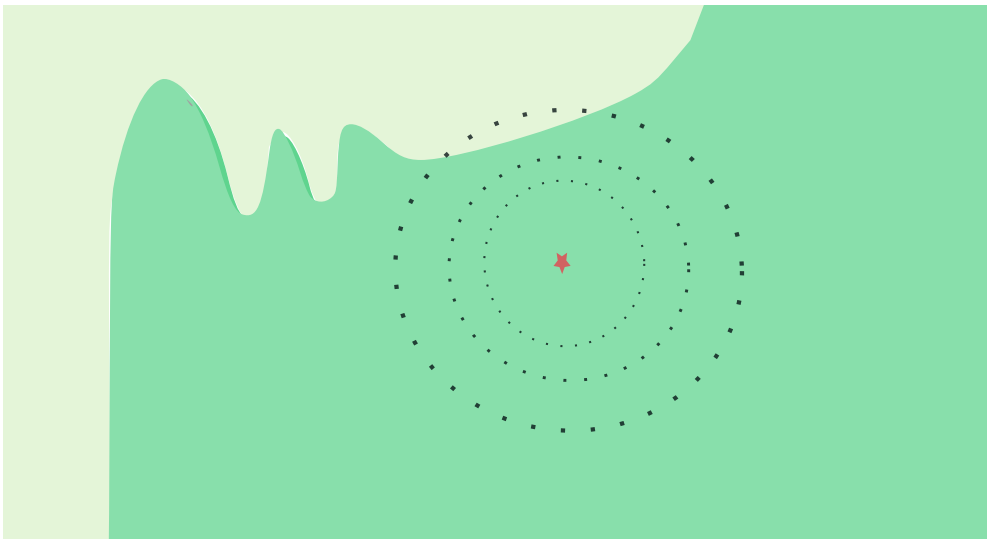


Figura 29 – Adequação do tamanho da vizinhança de x

Esse tipo de alteração é facilmente tratado pelo PDTX, pois mesmo que o aumento da sub-região implique em englobar regiões complexas, o formato da PDT permite lidar com esse tipo de complexidade, que é muito difícil de ser atendido por métodos globalmente lineares.

4.5 Seleção de características

O processo heurístico pode ser custoso computacionalmente, já que várias soluções candidatas são testadas e evoluídas em busca da melhor solução possível. Embora não haja garantias de se alcançar este ótimo global, o processo normalmente produz soluções boas. Porém, dependendo da aplicação, o desempenho computacional deve ser considerado. Nesse

contexto de explicação local, onde identificar as variáveis preditivas mais importantes é essencial, técnicas de seleção de atributos podem contribuir para redução do custo computacional associado a esta identificação.

Sabe-se que no processo de desenvolvimento de modelos de ML, o pré-processamento dos dados é uma importante etapa na construção do modelo, pois nem todas as características contribuem positivamente para o bom desempenho do método de aprendizagem automática. Contudo, ainda que o pré-processamento dos dados tenha ocorrido de forma apreciável, é possível que, para avaliação local de uma amostra de interesse, nem todas as variáveis contribuam para a predição realizada, e a remoção dessas variáveis possam melhorar a fidelidade do modelo transparente em relação à predição realizada pelo caixa-preta. A seleção de características consiste em determinar os atributos que são mais relevantes para a classificação ou regressão realizada pelo método de ML e descartar àquelas que possuem baixa importância, ou mesmo influência negativa.

Pensando nisso, foi proposta a avaliação comparativa entre 5 (cinco) métodos de seleção de características quanto à fidelidade do PDTX. Avalia-se, portanto, qual dos métodos de seleção estudado produz melhor fidelidade média para o *benchmark* de classificação escolhido. No caso do PDTX, ao descartar atributos irrelevantes, pode-se reduzir o custo computacional associado à evolução da PDT, caso a definição das características seja realizada à priori ou no início da aplicação da abordagem evolutiva.

4.5.1 Boruta

O Boruta é um método de seleção de características originalmente proposto por Rudnicki et al. (2006) e melhorado em Kursa et al. (2010). Nesse método, para cada atributo original, um atributo 'sombra' é gerado pela permutação aleatória de seus valores. Então, avalia-se se a importância atribuída a cada característica é superior à importância máxima alcançada por um atributo de contraste. Para obter resultados estatisticamente significativos este procedimento é repetido várias vezes, com variáveis de contraste geradas independentemente para cada iteração. Após cada iteração, o algoritmo verifica quantas vezes a importância dos atributos testados é maior (ou menor) do que a variável de contraste mais bem classificada. Uma vez que este número é significativamente maior do que o permitido sob hipótese de igualdade com importância de maior contraste aleatório, o atributo é considerado relevante e não testado posteriormente. Muitos trabalhos mostram resultados promissores encontrados por este método (Yahaya et al., 2020; Nahar et al., 2021), o que o permite ser considerado para este tipo de tarefa .

Existem vários métodos na literatura que podem ser usados na tarefa de seleção de características (Cai et al., 2018). Dentre as vantagens apresentadas pelo Boruta, destaca-se o bom funcionamento para problemas de classificação e regressão; o fato de ele considerar os relacionamentos multivariáveis; além de ser uma melhoria do método de *Random Forest*,

que é comumente utilizado para esta tarefa. Considerando que, a maioria dos outros algoritmos de seleção de variáveis segue um método ótimo mínimo (Kursa e Rudnicki, 2010), onde se depende de um pequeno subconjunto de recursos que produz um erro mínimo em um classificador escolhido, o Boruta é capaz de lidar com interações entre variáveis e com a natureza aleatória de importância gerada pelo *Random Forest*.

4.5.2 RFE

O RFE (*Recursive Feature Elimination* ou Eliminação de Recurso Recursivo, em tradução livre), é uma metodologia de seleção de atributos que utiliza um algoritmo de ML interno para realizar a tarefa de redução do número de características. Nesse contexto, o método de ML classifica os recursos de acordo com a importância de cada um deles, e descarta aqueles com menor relevância, recursivamente, até que o número de atributos pré-definido seja alcançado. O RFE busca melhorar o desempenho da generalização removendo as características menos importantes cuja a exclusão terá o menor efeito sobre os erros de treinamento (Chen e Jeong, 2007). Tal técnica pode ser usada tanto no contexto de classificação, como no de regressão.

4.5.3 Seleção por X^2

A seleção por X^2, ou *Chi-squared*, é feita via método estatístico que utiliza operações aritméticas simples para avaliar quais recursos devem ser selecionados para o processo de aprendizado supervisionado. A cada característica é atribuída uma pontuação, a qual é utilizada para ordenar por importância os preditores da base de dados. A seleção se baseia no teste de mesmo nome, o qual mede a dependência entre variáveis estocásticas, portanto, usar essa função "elimina" as características que são mais prováveis de serem independentes da classe e, portanto, irrelevantes para a classificação (Nassar et al., 2019).

4.5.4 Seleção por Regularização - LR

Explicadores que utilizam modelos lineares, como Regressão Linear, podem fazer uso de regularização para fazer essa seleção. A regularização Lasso, também chamada de *L1 Norm*, por exemplo, zera os coeficientes dos atributos irrelevantes, enquanto a regularização Ridge reduz os coeficientes para valores próximos de zero, sem zerá-los. Essa segunda é interessante em contextos que as características são independentes (não-correlacionadas) e que a maioria é útil na predição da resposta. O objetivo da regularização é adicionar uma penalidade aos diferentes parâmetros do modelo com a finalidade de reduzir o grau de liberdade do modelo em busca de reduzir o *overfitting*. Neste trabalho, esse tipo de seleção também foi avaliado

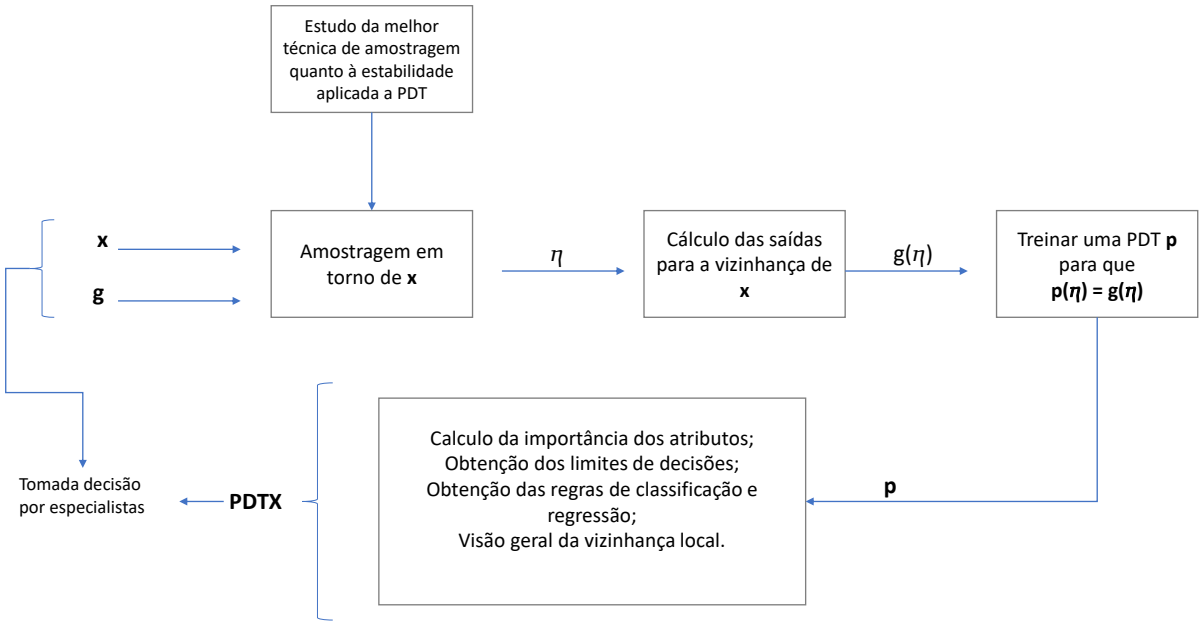
4.5.5 Seleção por Extremely Randomized Trees Classifier

O Extremely Randomized Trees Classifier (Classificador de Árvores Extremamente Randomizadas) também pode ser utilizado nessa tarefa, visto que agrega os resultados de várias DTs não correlacionadas coletadas em uma "floresta" para produzir o resultado de classificação. O processo de seleção de característica utilizando essa técnica é feito da seguinte maneira: durante a construção da floresta, para cada atributo, a redução total normalizada nos critérios matemáticos usados para a partição do nó, por exemplo o índice de Gini, é calculado. Para realizar a seleção de recursos, cada atributo é ordenado em ordem decrescente de acordo com a importância Gini e o usuário seleciona as k principais características de acordo com sua escolha (Beat, 2020).

A proposta é verificar se alguma dessas técnicas tem melhor resultado quanto à fidelidade quando aplicada em conjunto com o PDTX. Sabe-se que a redução de dimensionalidade normalmente se traduz em redução do custo computacional. Tendo isso em vista, a aplicação de uma dessas técnicas ajuda na utilização de bases grandes, já que torna o processo de geração de explicação mais rápido. Ressalta-se que no contexto da PDTX, o próprio explicador consegue estabelecer a importância dos atributos avaliados. Todavia, métodos de ML que foram treinados com muitos atributos irrelevantes, demandam do explicador maior esforço computacional para que a aproximação local seja bem sucedida. Além disso, conforme mencionado anteriormente, ainda que o processo de pré-processamento de dados tenha removido atributos irrelevantes, é possível que localmente nem todos atributos tenha relevância para a predição. Tendo isso em vista, pode-se opcionalmente realizar o processo de seleção de atributos, para que o explicador gere as regras, determine a hierarquia de importância entre os recursos e determine os limites de decisão somente sobre os atributos que de fato tenham importância no processo de classificação ou regressão.

Tendo em vista o apresentado até aqui, a Figura 30 apresenta o *pipeline* com as fases do PDTX. Dado um método caixa preta g e uma amostra x, a qual deseja-se explicar a predição de $g(x)$, gera-se uma vizinhança η em torno de x, utilizando-se a técnica de amostragem que se mostrar mais estável quando aplicada a PDT, dentre as amostragens aleatória, a partir da distribuição gaussiana, amostragem via hipercubo latino e via sequencia de Hammesley. Uma vez gerada essa vizinhança, calcula-se as saídas de g para esses pontos. Evolui uma PDT p usando o jSO de forma que as saídas da PDT sejam próximas às saídas do método caixa-preta, ou seja $p(\eta) \approx g(\eta)$. Uma vez que isso ocorrer, gera-se as explicações por meio de p, que juntamente com os dados iniciais, podem ser usadas para tomada de decisão pelos especialistas.

Figura 30 – Pipeline PDTX



Fonte: Os autores

4.6 Considerações do capítulo

1. Neste capítulo foi detalhado como se dá o funcionamento de uma PDT tradicional, que é a base do explicador proposto, PDTX. Esse tipo de árvore mostra-se interessante por utilizar hiperplanos oblíquos para fazer a divisão do espaço de busca, considerando todos os atributos em simultâneo, diferente do que ocorre em uma DT. Pela boa capacidade de reprodução do comportamento local de métodos caixa-preta ela pode ser utilizada para fornecer explicações sobre eles.

2. Tendo em vista que alguns algorítimos evolucionários mostraram boa capacidade em evoluir este tipo de árvore, foi escolhido o algoritmo populacional jSO, que é uma evolução do DE. Nessa versão estão disponíveis adaptação automática de parâmetros, adequação do tamanho da vizinhança ao longo do processo evolucionário, estratégia de mutação que considera um histórico de valores que apresentaram bons resultados em gerações anteriores, bem como o armazenamento de um grupo de melhores indivíduos para aplicação da mutação, e opcionalmente dos piores indivíduos, por meio de um arquivo externo. Ressalta-se ainda os bons resultados apresentados por este método para solução de funções contínuas rotacionadas e deslocadas.

3. Foi mantida a estrutura de dados proposta em Lopes et al. (2012) para geração das PDTs. E um novo método para regressão baseado em PDT foi proposto, que

diferencia da PDT para classificação por conter regressões lineares nos nós folhas, ao invés de classes.

4. Descreveu-se como a PDT é evoluída para que consiga reproduzir as predições realizadas pelo caixa-preta, e como são retiradas informações a partir de uma PDT para gerar explicações locais. O processo descrito permite obtenção da importância local dos atributos, a determinação dos limites de decisões para cada variável, quando as demais são mantidas fixas; a obtenção do caminho seguido até a predição, com destaque dos atributos mais relevantes para cada divisão realizada pelos nós internos, e uma visão geral das regras geradas pela PDT, tendo em conta toda a vizinhança local avaliada, e não somente o caminho seguido até x.

5. Realizou-se a descrição dos principais modelos de vizinhança local abordado na literatura, propôs-se o estudo de três métodos de geração de amostragem: aleatória a partir de uma distribuição gaussiana, amostragem por hipercubo latino, e amostragem baseada na sequência Hammersley. Além disso, foi explicado um problema comum na definição do tamanho da sub-região em torno de x, que é um desafio para muitos explicadores que se baseiam em modelos globalmente lineares. Mostrou-se como a PDT lida naturalmente com este problema, e foi proposto ainda a flexibilização do tamanho da vizinhança local, para maior capacidade em fornecer explicações em contextos que outros métodos podem falhar em fazê-lo.

6. Foram apresentados algumas opções de seleção de características que podem ser utilizadas em conjunto com o PDTX. Tendo em vista que a PDT é um método de ML, é possível aplicar a essa seleção no pré-processamento de dados de forma a melhorar a fidelidade local. Isso é relevante pois, ainda que o método caixa-preta tenha sido pré-treinado somente com atributos relevantes, é possível que nem todos os atributos sejam localmente importantes. Sabendo disso, no próximo capítulo é apresentado o comparativo entre essas técnicas, no sentido de verificar se alguma delas se destaca em termos da fidelidade local, para problemas de classificação;

Capítulo 5

Resultados e Discussões

Neste capítulo a avaliação dos resultados alcançados é realizada. Primeiramente, faz-se um estudo quantitativo dos resultados obtidos pela metodologia aqui proposta, comparativamente a técnicas comuns na literatura, as quais são abordadas na subseção 5.1.4. Foram definidas três métricas para avaliação, conhecidas como fidelidade, estabilidade e simplicidade. Todos os detalhes de configuração é documentado na seção de 5.1. Na sequência, é feita a avaliação qualitativa dos resultados visuais apresentados por vários explicadores, e tais informações são comparadas entre si, para verificação se há coerência em especial entre o grau de importância atribuído por cada explicador a cada atributo.

5.1 Procedimento Experimental

Na sequência aborda-se os procedimentos experimentais adotados para cada um dos estudos a serem realizados. Conforme detalhado no Capítulo 4, o primeiro passo consiste em determinar qual o tipo de amostragem gera resultados mais estáveis quando aplicada ao PDTX, antes mesmo de prosseguir para as demais avaliações. Uma vez que esse resultado tenha sido obtido, os demais experimentos são feitos considerando a técnica que tiver se mostrado mais estável.

5.1.1 Amostragem da vizinhança local

Foram consideradas as técnicas de amostragem abordadas na seção 4.4 para os métodos caixa-preta considerados na subseção 5.1.4, para problemas de regressão. Para estes experimentos considerou-se o total de 10 base de dados $\times$ 30 execuções cada $\times 3$ métodos caixa-preta $\times 3$ métodos de amostragem, totalizando 2700 execuções.

- Cada uma das bases trabalhadas foi divida em treino e teste, na proporção 80/20. Cada método caixa-preta foi treinado utilizando-se o conjunto de dados de treino, e a partir da base de teste foi obtida a instância de interesse aleatoriamente;

- Foram gerados $30 \times D$ pontos para cada técnica de amostragem centralizados em x, onde D é a dimensão da base de dados, utilizando o *skopt* do módulo *scikit-optimize 0.8.1*, no Python 3;
- Os dados foram normalizados utilizando a normalização *Min-Max*, onde os dados são ajustados entre o intervalo de [0,1]. Tal adequação foi feita utilizando o pacote *preprocessing* do módulo *Sklearn* (Buitinck et al., 2013);
- O PDTX foi ajustado para gerar as mesmas previsões dos caixa-pretas para cada uma das vizinhanças estudadas;
- A PDT usada pelo PDTX é o melhor indivíduo resultante da execução do jSO, que foi configurado considerando 100 indivíduos e $D \times 5000$ avaliações da função objetivo, e com limites inferiores e superiores em $[-100,100]$;
- Os métodos caixa-preta utilizados foram executados a partir do módulo *scikit-learn 1.1.1* (Buitinck et al., 2013), considerando as seguintes funções e hiper-parâmetros:
  - MLPRegressor: configuração padrão da biblioteca, com função de ativação *relu*;
  - RandomForestRegressor: configuração padrão da biblioteca, exceto a quantidade de árvores, que foi alterada para 1000;
  - SVR: configuração padrão, exceto o Kernel, que foi definido como *rbf*.

O processo de avaliação quantitativo a ser realizado para avaliar o explicador proposto e os concorrentes é detalhado na sequência. Foram utilizadas três métricas de avaliação, a saber: fidelidade, estabilidade e simplicidade. Três métodos caixa-preta foram determinados para mensurar esses indicadores: Redes Neurais Artificiais, Máquinas de Vetores de Suporte e *Random Forest*. Todos esses métodos possuem versão para classificação e para regressão, e por isso foram adotados para ambos os tipos de problemas. Assim, 10 bases de dados foram escolhidas para cada classe de problema, ou seja, 10 bases para classificação e 10 para regressão. A fim de se fazer a avaliação estatística dos experimentos, foram coletadas as saídas para 30 execuções de cada explicador, sobre cada método caixa preta, em cada base de dados, totalizando 10 *datasets* $\times 3$ métodos caixa-preta $\times 30$ execuções $\times 2$ (regressão e classificação) $= 1800$ execuções para cada explicador – 5400 execuções, no total.

- Para comparação entre os explicadores o mesmo procedimento experimental descrito anteriormente foi considerado, e assim o PDTX, o LIME e a DT foram ajustados para obterem as previsões do caixa-preta para cada amostra, levando em conta as especificidades de cada um deles, descritas nas documentações, e a partir disso as métricas avaliadas são calculadas para cada um deles;

- A quantidade de avaliações da função objetivo realizada pelo jSO para ajuste do PDTX foi ajustada para $D \times 10000$ nos procedimentos de classificação;
- As DT de Classificação e Regressão seguiram as configurações padrões da biblioteca Buitinck et al. (2013), com ajuste de altura máxima igual a 5;
- O PDTX foi ajustado para altura dinâmica para procedimentos de classificação, que considera $h = 2^{\lceil a \rceil + 1}$, onde a é dado pelo quociente da quantidade de classes na vizinhança de x por 2; e altura fixa igual a 2 para procedimentos de regressão.
  - MLPClassifier: configuração padrão da biblioteca, com função de ativação *relu*;
  - RandomForestClassifier: configuração padrão da biblioteca, exceto a quantidade de árvores, que foi alterada para 1000;
  - SVC: configuração padrão, exceto o *Kernel*, que foi definido como *sigmoid*.

Por fim, para experimentos comparativos quanto à fidelidade média obtida pelas técnicas de seleção de características, a quantidade de amostras na vizinhança local ficou definido como $20 \times D$ e a quantidade de avaliação da função objetivo realizada pelo jSO ficou definida em $D \times 1000$. A quantidade de atributos utilizado nessa avaliação foi $\lceil D/2 \rceil$. Esse experimento foi utilizado considerando 10 bases de classificação $\times 3$ métodos de avaliação caixa preta $\times 30$ execuções $\times$ 5 métodos de redução de características, totalizando 4500 execuções. O principal objetivo desse experimento foi verificar se algumas das técnicas estudadas produz maior fidelidade quando a quantidade de atributos é reduzida pela metade. Por ter configuração experimental distinta da adotada no comparativo entre os explicadores, o resultado não é diretamente comparável ao obtido pelo PDTX naquele contexto.

5.1.2 Base de dados

Muitos dos trabalhos desenvolvidos mais recentemente utilizam poucas bases de dados para avaliação dos indicadores de qualidade, o que pode produzir conclusões equivocadas. Aqui considera-se que a utilização de várias bases de dados e vários métodos ML é essencial para averiguação da qualidade dos resultados. As bases de dados foram selecionadas a partir do repositório *Penn Machine Learning Benchmarks* (PMLB) (Olson et al., 2017), que fornece um conjunto de dados de referência para avaliar e comparar os algoritmos de ML supervisionados. Essa seleção foi baseada na variabilidade do número de características, amostras e classes. As Tabelas 4 e 5 apresentam a descrição do conjunto de dados utilizados.

Tabela 4 – Bases de dados para problemas de classificação

Base de dados	Atributos	Amostras	Classes
Adult	14	48842	2
Analcatdata_bankruptcy	6	50	2
Breast_w	9	699	2
Cleveland_normal	7	303	5
Heart_starlog	13	270	2
Hepatitis	19	155	2
Irish	5	500	2
Tic_tac_toe	9	958	2
ThreeOf	9	512	2
Vowel	13	990	11

Tabela 5 – Bases de dados para problemas de regressão

Base de dados	Atributos	Amostras
227_cpu_small	12	8192
229_pwLinear	10	200
230_machine_cpu	6	209
522_pm10	7	500
593_fri_c1_1000_10	10	1000
620_fri_c1_1000_25	25	1000
621_fri_c0_100_10	10	100
623_fri_c4_1000_10	10	1000
627_fri_c2_500_10	10	500
627_fri_c2_500_10	10	500
659_sleuth_ex1714	7	47
706_sleuth_case1202	6	93

5.1.3 Métricas de avaliação

Para quantificar a **estabilidade** das explicações geradas será utilizada a métrica *Kendall tau distance* (Distância de Tau)(Cicirello, 2019), que avalia a hierarquia de importância entre dois conjuntos. Essa é uma métrica que quantifica o número de desacordos entre duas listas. Quanto maior a distância, mais as duas listas diferem entre si. A opção por avaliar a hierarquia e não os coeficientes puramente deve-se ao fato de que, em termos práticos, os coeficientes obtidos são utilizados para dimensionar o grau de importância entre os recursos. Dessa forma, tanto para o LIME, como pra PDTX, pequenas mudanças dos coeficientes que não gerem alterações na hierarquia obtida mantém a estabilidade da

explicação obtida. A *Kendall tau distance* pode ser obtida da seguinte forma:

$$K_d(\tau_1, \tau_2) = \sum_{\{i,j\} \in P} \bar{K}_{i,j}(\tau_1,\tau_2) \tag{5.1}$$

onde

- P é o conjunto de pares não ordenados de elementos distintos em τ_1 e τ_2;
- $\bar{K}_{i,j}(\tau_1,\tau_2) = 0$ se i e j estão na mesma ordem em τ_1 e τ_2;
- $\bar{K}_{i,j}(\tau_1,\tau_2) = 1$ se i e j estão em ordens opostas em τ_1 e τ_2;

O objetivo é, portanto, minimizar a *Kendall tau distance* , cujo valor ótimo é 0, o que significa que as duas listas de ranks comparadas são iguais. Para cada base de dados, e cada método caixa-preta, serão coletadas as *Kendall tau distance* resultantes para duas execuções distintas considerando os mesmos hiper-parâmetros. Cada experimento será executado 30 vezes para comparação estatística, para problemas de classificação e regressão. As amostras a serem explicadas foram escolhidas aleatoriamente.

No que diz respeito à **fidelidade** local, para os problemas de classificação, os resultados serão avaliados de acordo com as Equações 4.2 e 4.3, cujo intuito é maximizar a fidelidade local, calculado por meio da acurácia entre o valor real predito pelo ML e o gerado pelo explicador, e para os problemas de regressão a Equação 4.5 será aplicada, onde a finalidade é minimizar o erro médio associado à aproximação dos explicadores com o caixa-preta, mensurado pelo EQM.

Outra métrica importante de ser avaliada é a **simplicidade** das explicações geradas. Essa métrica pode ser entendida como a capacidade de entender e auditar a previsão facilmente, e em modelos baseados em árvore ou em regras pode ser mensurada por meio do índice de interpretabilidade definido em Margot et al. (2021), sendo que um conjunto pequeno de regras e com pequenos comprimentos é preferível em relação ao oposto. Este índice é definido como a soma dos comprimentos das regras do modelo gerado. Assim, o índice de interpretabilidade de uma coleção de regras C é definido como:

$$Int(C) := \sum_{r \in C} c(r), \tag{5.2}$$

em que c é o comprimento de r. Margot et al. (2021) definem o índice de interpretabilidade do estimador g_n gerado por um conjunto de regras C_n como $Int(g_n) = Int(C_n)$. Para normalização entre 0 e 1, a simplicidade pode ser computada como a razão entre o mínimo dos índices de interpretabilidade dos algoritmos considerados dividido pelo índice de interpretabilidade desse algoritmo (Margot e Luta, 2021).

Tendo em vista que o PDTX produz mais de um tipo de regra, essa métrica pode ser avaliada de três formas distintas. Considerando regras IF/ELSE(SE-ENTÃO) em que as condições são equivalentes aos hiperplanos, e o predicado corresponde à classe predita, como apresentado na Figura 21. Ou seja:

$$\textbf{IF}(ax + \theta < 0)\textbf{THEN}\, Class\ a \tag{5.3}$$

Nesse formato, cada hiperplano produz somente uma regra ($ax + \theta < 0$), e a quantidade de regras geradas será equivalente ao número de hiperplanos entre o nó raiz e o nó folha. Este formato traduz em uma relação matemática entre entrada e saída, o que propicia ao usuário entender o porquê de obtenção de uma saída, e não outra. Ressalta-se, todavia, que embora este formato permita predizer qualquer nova amostra nas redondezas de x considerada na formulação da vizinhança, e de acordo com a definição de simplicidade apresentada por Margot et al. (2021), é simples por possuir poucas regras, é de interpretação mais difícil que o formato decomposto das saídas, o qual é explicitado mais adiante. Assim, a decomposição dessa estrutura, em várias pequenas regras, como foi apresentado, é mais direto e intuitivo ao usuário.

A PDTX também gera regras semelhantes ao LIME quanto aos limites de decisão de cada recurso, decompondo cada hiperplano que encontra-se entre o nó raiz e a predição em sub-regras, e eliminando as redundâncias. O LIME utiliza somente o modelo linear, em conjunto com os limites superiores e inferiores de cada atributo para definir os limites de decisão associados a eles. Por outro lado, o PDTX primeiro isola uma sub-região de interesse, para só assim calcular tais limites, utilizando o hiperplano mais próximo da amostra de interesse, no caso de aplicações de classificação, e uma RL, contida nos nós folhas, se tratando de problemas de regressão. Sendo assim, o número de regras para os dois explicadores será, em média, igual ou levemente superior para o PDTX. A forma como esses limites são gerados são mostradas na subseção 4.3.6. Essa avaliação é comparável com as regras geradas pelo LIME e por aquelas geradas por uma DT e será utilizada para comparar a simplicidade das explicações junto ao LIME e DT.

O terceiro formato, exemplificado na Figura 23, apresenta todas as minúcias contidas na vizinhança de x, e é encorajada a ser utilizada estudando um atributo por vez, como apresentado nas Figuras 24 e 25, já que, conhecendo a hierarquia de importância das características estudadas, interessa observar aquelas que possuem maior impacto na predição final. Sendo assim, para avaliação da simplicidade dessa explicação, considera-se todos os hiperplanos gerados para toda a PDT, isolando-se uma variável de interesse por vez. Essa avaliação, que é comparável com uma DT completa, não será explorada neste trabalho.

De acordo com a definição da Equação 5.2 é possível inferir que controlar o tamanho

da árvore, para modelos que utilizam essa estrutura, gera explicações mais simples que árvores longas. Tratando-se da DT, as regras são obtidas diretamente considerando o caminho da predição.

5.1.4 Métodos caixa-preta avaliados

Para avaliação e comparação com os métodos concorrentes foram definidos três métodos caixa-preta populares, que podem ser usados tanto nos contextos de regressão, como de classificação. A saber: Redes Neurais Artificiais, por meio da utilização do Perceptron Multicamadas (MLP), Máquinas de Vetores de Suporte (Support Vector Machines, SVM) e Random Forest (RF).

O MLP é sistema de neurônios simples interconectados, cujo modelo representa um mapeamento não linear entre um vetor de entrada e um vetor de saída (Gardner e Dorling, 1998). Essas estruturas são conectados por pesos e sinais de saída que é uma função da soma das entradas do nó modificado por uma simples função de transferência ou ativação não linear. Essa rede possui uma camada de entrada, uma ou mais camadas ocultas, e uma camada de saída. A alimentação da rede é para frente, popularmente chamada de "*feed-forward*", e utiliza-se o método *Backpropagation* para atualização dos pesos internos. Por possuir várias camadas interconectadas, ainda que os valores obtidos para os pesos estejam disponíveis, este método é considerado opaco, já que esses dados não conseguem fornecer informações sobre o processo de tomada de decisão realizado.

O SVM é um método tradicional de ML que é capaz de lidar com problemas não lineares ao mapear o conjunto de treinamento do espaço original, referenciado como de entradas, para um outro espaço de maior dimensão, denominado espaço de características (*feature space*), de forma que neste novo espaço os dados possam ser separados linearmente (Lorena e De Carvalho, 2007). Para problemas complexos, este mapeamento é feito por meio de *kernels* não lineares.

E, por fim, as RF são uma extensão da ideia de *bagging*, que pode ser usada predição de problemas de classificação e de regressão. As árvores usadas em Random Forests uma são DT, e como tal, particionam o espaço do preditor usando uma sequência de partições binárias em um atributo individual por vez. O nó "raiz" da árvore compreende todo o espaço do preditor. Os nós que não são divididos são chamados de "nós terminais" e formam a partição final do espaço preditor. Para formação de cada árvore,cada nó não terminal se divide em dois nós descendentes, um à esquerda e outro à direita, de acordo com o valor de uma das variáveis preditoras. Para uma variável preditora contínua, uma divisão é determinada por um ponto de divisão; pontos para os quais o preditor é menor que o ponto de divisão vão para a esquerda, o resto vai para a direita (Cutler et al., 2012). Assim, as RF utilizam um conjunto com muitas dessas árvores para realização das predições, e são consideradas caixa-preta por gerar regiões muito mais complexas do que

as árvores individuais, sendo, portanto de difícil interpretação.

Isto posto, o intuito é pré-treinar cada um desses métodos utilizando a base de treino, escolher aleatoriamente uma amostra a ser explicada a partir da base de teste, gerar o conjunto de ruído utilizando a técnica de amostragem dentre as apresentadas na seção 4.4 que se mostrar mais estável; fazer a aproximação local utilizando a PDT ajustada via JSO e os demais métodos concorrentes; gerar as explicações locais, e avaliar a fidelidade e estabilidade de todos os explicadores avaliados; e ao final fazer a avaliação qualitativa das explicações fornecidas.

5.1.5 Explicadores concorrentes

Foram selecionados dois métodos de explicação local para comparação quantitativa, a saber: DT e LIME. Conforme mencionado na seção 2.5, o LIME é uma abordagem agnóstica em relação ao modelo de ML caixa preta, que utiliza um modelo local interpretável para explicar cada previsão individual. Semelhante ao PDTX, gera-se uma vizinhança em torno da amostra de interesse e por meio de um modelo linear é feita a aproximação das predições realizadas pelo caixa-preta. A DT pode-se ser utilizada aplicando este mesmo princípio para gerar explicações locais. Sendo assim, propõe-se a utilização deste dois métodos para comparação quantitativa quanto à aproximação realizada localmente por estes métodos, em termos de estabilidade e fidelidade. Além disso, foi proposto a comparação qualitativa das explicações com o SHAP, também aplicado à explicação local.

5.2 Resultados – Avaliação quantitativa

Nesta seção são apresentados os resultados obtidos para os estudos propostos. Na subseção 5.2.1 encontram-se os valores médios obtidos para estabilidade local de cada uma das técnicas de amostragem estudadas, medidas por meio da Distância de Tau. A partir do obtido nessa subseção, define-se a amostragem a ser utilizada pelo PDTX nas seções seguintes.

Na subseção 5.3 são apresentados os valores alcançados em termos de fidelidade local dos explicadores estudados para os contextos de classificação e regressão, obtidos via medida de acurácia e MSE, respectivamente. Também são apresentados os resultados quanto estabilidade dessas mesmas abordagens, na subseção 5.4. Por fim, os resultados obtidos quanto à simplicidade de cada método são descritos na subseção 5.5. Em cada subseção são discutidos os pontos relevantes observados, e ao final são apresentadas as conclusões gerais dos resultados obtidos. Ao final do capítulo, apresenta-se alguns casos de uso de regressão e classificação para comparação visual qualitativa.

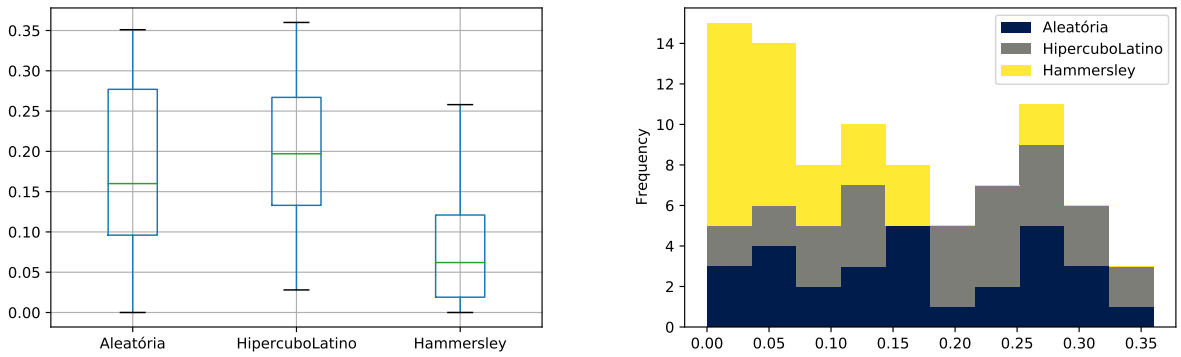
5.2.1 Amostragem da vizinhança local

No estudo da vizinhança local, referente ao método de amostragem a ser considerado no processo de evolução do PDTX, foram consideradas as amostragem aleatória, a partir de uma distribuição gaussiana, por hipercubo latino e via sequência de Hammersley. A Tabela 6 apresenta os resultados das Distâncias de Tau médias obtidas para o *benchmark* adotado.

Dentro das configurações pré-estabelecidas, o método de amostragem local que se mostrou mais estável foi a gerada via sequência de Hammersley. Quando comparadas com as amostragens aleatória a partir de uma distribuição gaussiana, e com a amostragem por Hipercubo Latino, observa-se que o resultado dessa amostragem foi pelo menos duas vezes melhor do que o observado para os demais tipos de amostragem. Dessas três, a amostragem via Hipercubo Latino foi a que teve o pior desempenho para esta métrica. Por outro lado, foi avaliado também se há diferenças nos valores obtidos para fidelidade. Nesse caso, foi observado que a amostragem aleatória foi a que obteve melhores resultados. Todavia, conforme será tratado adiante, dos valores de fidelidade obtidos para quaisquer uma das três técnicas superam significativamente os obtidos por dois dos explicadores tradicionais da literatura aqui comparados, conforme pode ser observado na Tabela 7. A Figura 31 apresenta visualmente a diferença na média e na distribuição dos valores médios retornados pela Distância de Tau para as bases de dados de regressão adotadas. Como pode-se notar, a distribuição Hammesley produz resultados bem menores que as demais.

Figura 31 – Comportamento das diferentes técnicas de amostragem local

(a) Box-plot (b) Histograma



Fonte: Os autores.

Foi aplicado o teste estatístico *Wilcoxon Signed Rank*, ao nível de confiança de 95%, para mensurar se as diferenças observadas são estatisticamente significantes. Obteve-se o valor de $p = 0{,}619$ na comparação par a par entre Amostragem aleatória $\times$ Amostragem por Hipercubo Latino, $p = 9{,}304 \times 10^{-6}$ na comparação entre Amostragem aleatória $\times$

Tabela 6 – Estabilidade das técnicas de amostragens: Aleatória × Hipercubo Latino × Hammersley via Distância de Tau (média ± desvio padrão)

Base de dados	**ML**	**Aleatória**	**Hipercubo Latino**	**Hammersley**
227_cpu_small	SVM	0,155 ± 0,041	0,098 ± 0,060	0,024 ± 0,059
229_pwLinear	SVM	0,043 ± 0,054	0,028 ± 0,048	0,000 ± 0,000
230_machine_cpu	SVM	0,133 ± 0,118	0,031 ± 0,069	0,036 ± 0,059
522_pm10	SVM	0,059 ± 0,086	0,140 ± 0,137	0,000 ± 0,000
593_fri_c1_1000_10	SVM	0,144 ± 0,123	0,106 ± 0,091	0,000 ± 0,000
621_fri_c0_100_10	SVM	0,047 ± 0,025	0,050 ± 0,048	0,000 ± 0,000
623_fri_c4_1000_10	SVM	0,096 ± 0,069	0,193 ± 0,120	0,124 ± 0,113
627_fri_c2_500_10	SVM	0,160 ± 0,158	0,099 ± 0,078	0,104 ± 0,119
659_sleuth_ex1714	SVM	0,108 ± 0,149	0,184 ± 0,133	0,121 ± 0,124
706_sleuth_case1202	SVM	0,258 ± 0,249	0,133 ± 0,133	0,062 ± 0,107
Resultado médio SVM		0,120 ± 0,107	0,106 ± 0,092	0,047 ± 0,058
227_cpu_small	RF	0,221 ± 0,083	0,223 ± 0,079	0,144 ± 0,186
229_pwLinear	RF	0,230 ± 0,142	0,219 ± 0,129	0,104 ± 0,055
230_machine_cpu	RF	0,289 ± 0,169	0,221 ± 0,148	0,258 ± 0,168
522_pm10	RF	0,000 ± 0,000	0,267 ± 0,166	0,067 ± 0,111
593_fri_c1_1000_10	RF	0,284 ± 0,091	0,332 ± 0,148	0,256 ± 0,078
623_fri_c4_1000_10	RF	0,006 ± 0,005	0,252 ± 0,091	0,168 ± 0,078
627_fri_c2_500_10	RF	0,351 ± 0,111	0,360 ± 0,105	0,067 ± 0,000
621_fri_c0_100_10	RF	0,102 ± 0,116	0,269 ± 0,110	0,156 ± 0,000
659_sleuth_ex1714	RF	0,286 ± 0,177	0,197 ± 0,114	0,138 ± 0,103
706_sleuth_case1202	RF	0,001 ± 0,001	0,289 ± 0,190	0,040 ± 0,087
Resultado médio RF		0,177 ± 0,090	0,263 ± 0,128	0,140 ± 0,087
227_cpu_small	MLP	0,170 ± 0,083	0,141 ± 0,070	0,065 ± 0,125
229_pwLinear	MLP	0,131 ± 0,099	0,142 ± 0,101	0,010 ± 0,022
230_machine_cpu	MLP	0,036 ± 0,064	0,047 ± 0,025	0,085 ± 0,086
522_pm10	MLP	0,321 ± 0,136	0,187 ± 0,183	0,010 ± 0,036
623_fri_c4_1000_10	MLP	0,300 ± 0,180	0,294 ± 0,229	0,044 ± 0,115
627_fri_c2_500_10	MLP	0,277 ± 0,195	0,247 ± 0,149	0,050 ± 0,062
621_fri_c0_100_10	MLP	0,280 ± 0,145	0,221 ± 0,148	0,019 ± 0,034
659_sleuth_ex1714	MLP	0,184 ± 0,150	0,279 ± 0,178	0,000 ± 0,000
706_sleuth_case1202	MLP	0,169 ± 0,172	0,320 ± 0,207	0,019 ± 0,049
Resultado médio MLP		0,208 ± 0,136	0,209 ± 0,143	0,034 ± 0,059
Média geral		0,167 ± 0,110	0,191 ± 0,120	0,072 ± 0,068

Amostragem Hammersley e $p = 6{,}867 \times 10^{-4}$ entre Amostragem por Hipercubo Latino × Hammersley. Ou seja, as diferenças observadas em termos de média entre o comparativo entre as vizinhanças aleatória e aquelas geradas via sequência de Hammersley, bem como via Hipercubo Latino e Hammersley são significativas dentro do nível de confiança adotado.

Testes empíricos mostraram que é possível obter alta fidelidade e estabilidade

utilizando uma amostragem aleatória, mas algumas ressalvas devem ser consideradas. Caso se utilize uma amostra pequena (< 500 pontos – valor definido empiricamente com base em observações experimentais), pode-se chegar a uma situação de *underfitting* das PDTs geradas, pois é possível que elas não consigam compreender todas as particularidades da sub-região amostrada. Assim, a cada novo conjunto de ruído gerado aleatoriamente, os coeficientes utilizados para geração das explicações serão diferentes para outras execuções, considerando-se os mesmos hiper-parâmetros. Uma possível solução para evitar isso neste contexto é a amostragem considerando uma grande quantidade de pontos, de acordo com a quantidade de atributos utilizados nesse processo. Considerando os exaustivos testes realizados, não foi possível definir uma regra que determine a quantidade de gerações necessárias para evoluir a PDT por meio do jSO, que possa ser generalizada para qualquer base de dados, para essa distribuição.

Por outro lado, de forma semelhante a algumas extensões do LIME (Ribeiro et al., 2016), como Zhou et al. (2021) e Botari et al. (2020), a estabilidade também pode ser melhorada considerando um grande conjunto de pontos na vizinhança pré-estabelecida, o que também reduz o sub-ajuste. Nesse caso, é necessário ajustar o jSO para que se execute uma quantidade de gerações tal que se alcance uma situação de estabilidade dos coeficientes obtidos pela árvore. Entretanto, além de um alto custo computacional, a quantidade de gerações necessárias deve ser grande, e dependerá não só do tamanho da vizinhança local, e da altura da PDT, como também das características de cada base de dados em concomitância com o caixa-preta a ser explicado. Sendo assim, embora seja possível a utilização desta técnica de amostragem, e ela esteja disponível no *framework* gerado por este trabalho, considera-se que ela não deve ser a primeira opção de amostragem a ser considerada.

Tabela 7 – Fidelidade das técnicas de amostragens: Aleatória × Hipercubo Latino × Hammersley (média ± desvio padrão)

Base de dados	**ML**	**Aleatória**	**Hipercubo Latino**	**Hammersley**
227_cpu_small	SVM	$5{,}965 \times 10^{-21} \pm 1{,}174 \times 10^{-21}$	$5{,}785 \times 10^{-21} \pm 1{,}314 \times 10^{-21}$	$7{,}305 \times 10^{-9} \pm 3{,}582 \times 10^{-9}$
229_pwlinear	SVM	$1{,}008 \times 10^{-11} \pm 3{,}198 \times 10^{-12}$	$1{,}122 \times 10^{-11} \pm 3{,}148 \times 10^{-12}$	$4{,}737 \times 10^{-8} \pm 2{,}469 \times 10^{-8}$
230_machine_cpu	SVM	$1{,}012 \times 10^{-6} \pm 2{,}199 \times 10^{-7}$	$1{,}852 \times 10^{-10} \pm 3{,}656 \times 10^{-10}$	$3{,}006 \times 10^{-10} \pm 2{,}155 \times 10^{-10}$
522_pm10	SVM	$3{,}018 \times 10^{-8} \pm 1{,}125 \times 10^{-8}$	$2{,}814 \times 10^{-8} \pm 1{,}363 \times 10^{-8}$	$1{,}343 \times 10^{-10} \pm 7{,}731 \times 10^{-11}$
593_fri_c1_1000_10	SVM	$2{,}462 \times 10^{-13} \pm 1{,}507 \times 10^{-13}$	$2{,}321 \times 10^{-13} \pm 1{,}241 \times 10^{-13}$	$1{,}133 \times 10^{-9} \pm 3{,}595 \times 10^{-10}$
621_fri_c0_100_10	SVM	$3{,}275 \times 10^{-13} \pm 1{,}818 \times 10^{-13}$	$1{,}263 \times 10^{-7} \pm 1{,}379 \times 10^{-7}$	$4{,}469 \times 10^{-10} \pm 1{,}408 \times 10^{-10}$
623_fri_c4_1000_10	SVM	$5{,}024 \times 10^{-13} \pm 1{,}802 \times 10^{-13}$	$4{,}544 \times 10^{-13} \pm 1{,}675 \times 10^{-13}$	$2.423 \times 10^{-7} \pm 2.648 \times 10^{-7}$
627_fri_c2_500_10	SVM	$9{,}295 \times 10^{-14} \pm 5{,}863 \times 10^{-14}$	$7{,}856 \times 10^{-14} \pm 3{,}632 \times 10^{-14}$	$9{,}108 \times 10^{-8} \pm 1.305 \times 10^{-8}$
659_sleuth_ex1714	SVM	$1{,}425 \times 10^{-7} \pm 5{,}370 \times 10^{-8}$	$1{,}821 \times 10^{-7} \pm 8{,}097 \times 10^{-8}$	$1{,}853 \times 10^{-7} \pm 7{,}430 \times 10^{-8}$
706_sleuth_case1202	SVM	$2{,}580 \times 10^{-6} \pm 8{,}884 \times 10^{-7}$	$2{,}729 \times 10^{-6} \pm 9{,}814 \times 10^{-7}$	$2{,}567 \times 10^{-6} \pm 9{,}974 \times 10^{-7}$
Resultado médio SVM		$3{,}765 \times 10^{-7} \pm 1{,}173 \times 10^{-7}$	$3{,}066 \times 10^{-7} \pm 1{,}214 \times 10^{-7}$	$2{,}452 \times 10^{-5} \pm 2{,}790 \times 10^{-5}$
227_cpu_small	RF	$1{,}722 \times 10^{-3} \pm 1{,}404 \times 10^{-3}$	$1{,}639 \times 10^{-3} \pm 1{,}954 \times 10^{-3}$	$7{,}586 \times 10^{-30} \pm 1{,}137 \times 10^{-29}$
229_pwlinear	RF	$3{,}004 \times 10^{-1} \pm 1{,}190 \times 10^{-1}$	$3{,}291 \times 10^{-1} \pm 1{,}596 \times 10^{-1}$	$5{,}81 \times 10^{-2} \pm 1{,}112 \times 10^{-1}$
522_pm10	RF	$2{,}292 \times 10^{-4} \pm 2{,}103 \times 10^{-4}$	$2{,}137 \times 10^{-4} \pm 2{,}463 \times 10^{-4}$	$2{,}721 \times 10^{-4} \pm 3{,}882 \times 10^{-4}$
593_fri_c1_1000_10	RF	$1{,}423 \times 10^{-2} \pm 1{,}066 \times 10^{-2}$	$9{,}228 \times 10^{-3} \pm 6{,}090 \times 10^{-3}$	$4{,}086 \times 10^{-4} \pm 2{,}906 \times 10^{-4}$
621_fri_c0_100_10	RF	$7{,}144 \times 10^{-4} \pm 6{,}491 \times 10^{-4}$	$1{,}103 \times 10^{-3} \pm 1{,}821 \times 10^{-3}$	$2{,}693 \times 10^{-2} \pm 1{,}257 \times 10^{-3}$
623_fri_c4_1000_10	RF	$5{,}924 \times 10^{-3} \pm 4{,}581 \times 10^{-3}$	$6{,}334 \times 10^{-3} \pm 5{,}554 \times 10^{-3}$	$7{,}085 \times 10^{-4} \pm 1{,}401 \times 10^{-3}$
627_fri_c2_500_10	RF	$1{,}480 \times 10^{-3} \pm 2{,}168 \times 10^{-3}$	$4{,}318 \times 10^{-4} \pm 4{,}521 \times 10^{-4}$	$7{,}685 \times 10^{-4} \pm 7{,}488 \times 10^{-5}$
659_sleuth_ex1714	RF	$1{,}991 \times 10^{+3} \pm 2{,}120 \times 10^{+3}$	$2{,}301 \times 10^{+3} \pm 2{,}561 \times 10^{+3}$	$2{,}900 \times 10^{+3} \pm 5{,}918 \times 10^{+3}$
706_sleuth_case1202	RF	$6{,}159 \times 10^{-4} \pm 1{,}165 \times 10^{-3}$	$4{,}102 \times 10^{-3} \pm 1{,}630 \times 10^{-2}$	$1.772 \times 10^{0} \pm 4.196 \times 10^{0}$
Resultado médio RF		$2{,}489 \times 10^{2} \pm 2{,}650 \times 10^{2}$	$2{,}877 \times 10^{2} \pm 3{,}201 \times 10^{2}$	$5{,}840 \times 10^{2} \pm 1{,}264 \times 10^{3}$
227_cpu_small	MLP	$8{,}003 \times 10^{-4} \pm 2{,}146 \times 10^{-3}$	$5{,}088 \times 10^{-7} \pm 1{,}341 \times 10^{-6}$	$3{,}449 \times 10^{-4} \pm 9{,}865 \times 10^{-4}$
229_pwlinear	MLP	$6{,}674 \times 10^{-4} \pm 8{,}345 \times 10^{-4}$	$2{,}539 \times 10^{-1} \pm 2{,}112 \times 10^{-1}$	$1{,}074 \times 10^{-5} \pm 1{,}243 \times 10^{-5}$
522_pm10	MLP	$1{,}382 \times 10^{-4} \pm 3{,}163 \times 10^{-4}$	$5{,}265 \times 10^{-5} \pm 1{,}486 \times 10^{-4}$	$1{,}135 \times 10^{-9} \pm 3{,}635 \times 10^{-9}$
593_fri_c1_1000_10	MLP	$4{,}720 \times 10^{-3} \pm 3{,}465 \times 10^{-3}$	$1{,}189 \times 10^{-2} \pm 8{,}901 \times 10^{-3}$	$8{,}262 \times 10^{-3} \pm 1{,}009 \times 10^{-2}$
621_fri_c0_100_10	MLP	$1{,}431 \times 10^{-4} \pm 1{,}290 \times 10^{-4}$	$6{,}818 \times 10^{-3} \pm 7{,}398 \times 10^{-3}$	$1{,}960 \times 10^{-6} \pm 2{,}119 \times 10^{-6}$
623_fri_c4_1000_10	MLP	$2{,}510 \times 10^{-3} \pm 1{,}913 \times 10^{-3}$	$6{,}869 \times 10^{-3} \pm 1{,}304 \times 10^{-2}$	$1{,}455 \times 10^{-6} \pm 3{,}366 \times 10^{-6}$
627_fri_c2_500_10	MLP	$1{,}358 \times 10^{-3} \pm 2{,}748 \times 10^{-3}$	$5{,}504 \times 10^{-4} \pm 6{,}662 \times 10^{-4}$	$7{,}230 \times 10^{-6} \pm 1{,}327 \times 10^{-5}$
659_sleuth_ex1714	MLP	$5{,}706 \times 10^{-7} \pm 1{,}688 \times 10^{-6}$	$2{,}741 \times 10^{-7} \pm 6{,}207 \times 10^{-7}$	$6{,}706 \times 10^{-26} \pm 1{,}131 \times 10^{-26}$
706_sleuth_case1202	MLP	$9{,}017 \times 10^{-5} \pm 2{,}879 \times 10^{-4}$	$4{,}583 \times 10^{-4} \pm 1{,}607 \times 10^{-3}$	$6{,}848 \times 10^{-11} \pm 3{,}536 \times 10^{-10}$
Resultado médio MLP		$1{,}159 \times 10^{-3} \pm 1{,}316 \times 10^{-3}$	$3{,}117 \times 10^{-2} \pm 2{,}700 \times 10^{-2}$	$9{,}587 \times 10^{-4} \pm 1{,}234 \times 10^{-3}$
MSE Médio		$7{,}112 \times 10^{1} \pm 7{,}572 \times 10^{1}$	$8{,}220 \times 10^{1} \pm 9{,}148 \times 10^{1}$	$1{,}669 \times 10^{2} \pm 3{,}612 \times 10^{2}$

5.3 Fidelidade

Na sequência, apresenta-se a fidelidade obtida no comparativo entre os explicadores avaliados para os contextos de regressão e classificação.

5.3.1 Regressão

Considerando o procedimento experimental proposto, foi feita a comparação entre os valores de erro obtidos, via MSE para os três explicadores comparados, para cada uma das 10 bases escolhidas, tendo em conta os 3 métodos caixa-preta avaliados. A Tabela 8 sumariza os resultados encontrados a partir do proposto. Observa-se que os resultados encontrados pelo PDTX são consideravelmente melhores do que o do LIME e da DT dentro do contexto de avaliação proposto.

Tabela 8 – Fidelidade dos explicadores (via MSE) - Regressão (média ± desvio padrão)

Base de dados	**ML**	**LIME**	**DT**	**PDTX**
227_cpu_small	SVM	$2{,}009 \times 10^{3} \pm 2{,}613 \times 10^{3}$	$1{,}753 \times 10^{3} \pm 1{,}327 \times 10^{3}$	$7{,}305 \times 10^{-9} \pm 3{,}582 \times 10^{-9}$
229_pwLinear	SVM	$3{,}680 \times 10^{1} \pm 2{,}694 \times 10^{1}$	$1{,}179 \times 10^{1} \pm 4{,}377 \times 10^{0}$	$4{,}737 \times 10^{-8} \pm 2{,}469 \times 10^{-8}$
230_machine_cpu	SVM	$1{,}136 \times 10^{0} \pm 1{,}613 \times 10^{0}$	$3{,}972 \times 10^{5} \pm 1{,}223 \times 10^{5}$	$3{,}006 \times 10^{-10} \pm 3{,}350 \times 10^{-10}$
522_pm10	SVM	$7{,}102 \times 10^{1} \pm 6{,}668 \times 10^{1}$	$1{,}066 \times 10^{0} \pm 0{,}389 \times 10^{0}$	$1{,}343 \times 10^{-10} \pm 7{,}731 \times 10^{-11}$
593_fri_c1_1000_10	SVM	$1{,}380 \times 10^{1} \pm 9{,}000 \times 10^{0}$	$8{,}750 \times 10^{-1} \pm 2{,}360 \times 10^{-1}$	$1{,}133 \times 10^{-9} \pm 3{,}595 \times 10^{-10}$
621_fri_c0	SVM	$2{,}158 \times 10^{0} \pm 1{,}722 \times 10^{0}$	$7{,}860 \times 10^{-1} \pm 4{,}470 \times 10^{-1}$	$4{,}469 \times 10^{-10} \pm 1{,}408 \times 10^{-10}$
623_fri_c4_1000_10	SVM	$7{,}856 \times 10^{1} \pm 4{,}628 \times 10^{1}$	$8{,}980 \times 10^{-1} \pm 4{,}810 \times 10^{-1}$	$2{,}423 \times 10^{-7} \pm 2{,}648 \times 10^{-7}$
627_fri_c2_500_10	SVM	$7{,}213 \times 10^{0} \pm 5{,}298 \times 10^{0}$	$8{,}160 \times 10^{1} \pm 4{,}560 \times 10^{1}$	$9{,}108 \times 10^{-8} \pm 1{,}305 \times 10^{-8}$
620_fri_c1_1000_25	SVM	$1{,}208 \times 10^{0} \pm 9{,}440 \times 10^{1}$	$8{,}540 \times 10^{1} \pm 3{,}310 \times 10^{1}$	$1{,}937 \times 10^{-7} \pm 7{,}023 \times 10^{-7}$
706_sleuth_case1202	SVM	$4{,}000 \times 10^{-3} \pm 0{,}002 \times 10^{-3}$	$1{,}190 \times 10^{4} \pm 1{,}111 \times 10^{4}$	$2{,}567 \times 10^{-6} \pm 9{,}974 \times 10^{-7}$
Resultado médio SVM		$2{,}221 \times 10^{2} \pm 6{,}671 \times 10^{3}$	$4{,}110 \times 10^{4} \pm 1{,}348 \times 10^{4}$	$3{,}151 \times 10^{-7} \pm 2{,}007 \times 10^{-7}$
227_cpu_small	RF	$1{,}393 \times 10^{3} \pm 1{,}751 \times 10^{3}$	$7{,}204 \times 10^{2} \pm 2{,}082 \times 10^{2}$	$5{,}81 \times 10^{-2} \pm 1{,}112 \times 10^{-1}$
229_pwLinear	RF	$2{,}681 \times 10^{1} \pm 1{,}581 \times 10^{1}$	$2{,}125 \times 10^{0} \pm 7{,}575 \times 10^{-1}$	$7{,}586 \times 10^{-30} \pm 1{,}137 \times 10^{-29}$
230_machine_cpu	RF	$2{,}730 \times 10^{5} \pm 8{,}978 \times 10^{4}$	$4{,}501 \times 10^{3} \pm 4{,}505 \times 10^{3}$	$4{,}148 \times 10^{0} \pm 1{,}225 \times 10^{1}$
522_pm10	RF	$2{,}310 \times 10^{-1} \pm 1{,}080 \times 10^{-1}$	$5{,}47 \times 10^{-1} \pm 1{,}518 \times 10^{-1}$	$2{,}721 \times 10^{-4} \pm 3{,}882 \times 10^{-4}$
593_fri_c1_1000_10	RF	$8{,}560 \times 10^{-1} \pm 3{,}140 \times 10^{-1}$	$1{,}55 \times 10^{-1} \pm 5{,}066 \times 10^{-2}$	$4{,}086 \times 10^{-4} \pm 2{,}906 \times 10^{-4}$
621_fri_c0	RF	$9{,}560 \times 10^{-1} \pm 7{,}850 \times 10^{-1}$	$5{,}270 \times 10^{-1} \pm 5{,}255 \times 10^{-1}$	$2{,}693 \times 10^{-2} \pm 1{,}257 \times 10^{-3}$
623_fri_c4_1000_10	RF	$1{,}085 \times 10^{0} \pm 5{,}010 \times 10^{1}$	$1{,}76 \times 10^{-1} \pm 9{,}398 \times 10^{-2}$	$7{,}085 \times 10^{-4} \pm 1{,}401 \times 10^{-3}$
627_fri_c2_500_10	RF	$7{,}260 \times 10^{1} \pm 2{,}840 \times 10^{0}$	$2{,}110 \times 10^{-1} \pm 1{,}334 \times 10^{-1}$	$7{,}685 \times 10^{-4} \pm 7{,}488 \times 10^{-5}$
620_fri_c1_1000_25	RF	$1{,}820 \times 10^{-1} \pm 4{,}195 \times 10^{-2}$	$1{,}820 \times 10^{-1} \pm 4{,}195 \times 10^{-2}$	$3{,}046 \times 10^{-5} \pm 0{,}001 \times 10^{-1}$
706_sleuth_case1202	RF	$6{,}511 \times 10^{3} \pm 2{,}999 \times 10^{3}$	$2{,}400 \times 10^{3} \pm 2{,}719 \times 10^{3}$	$1{,}772 \times 10^{0} \pm 4{,}196 \times 10^{0}$
Resultado médio RF		$2{,}810 \times 10^{4} \pm 9{,}464 \times 10^{3}$	$7{,}625 \times 10^{2} \pm 7{,}434 \times 10^{2}$	$6{,}007 \times 10^{-1} \pm 1{,}656 \times 10^{0}$
227_cpu_small	MLP	$5{,}815 \times 10^{2} \pm 1{,}578 \times 10^{3}$	$1{,}293 \times 10^{3} \pm 1{,}069 \times 10^{3}$	$3{,}449 \times 10^{-4} \pm 9{,}865 \times 10^{-4}$
229_pwLinear	MLP	$5{,}328 \times 10^{1} \pm 3{,}659 \times 10^{1}$	$1{,}336 \times 10^{1} \pm 5{,}994 \times 10^{0}$	$1{,}074 \times 10^{-5} \pm 1{,}243 \times 10^{-5}$
230_machine_cpu	MLP	$2{,}910 \times 10^{1} \pm 3{,}879 \times 10^{1}$	$1{,}114 \times 10^{4} \pm 1{,}854 \times 10^{4}$	$1{,}772 \times 10^{0} \pm 4{,}196 \times 10^{0}$
522_pm10	MLP	$1{,}680 \times 10^{-1} \pm 2{,}030 \times 10^{1}$	$7{,}940 \times 10^{-1} \pm 3{,}406 \times 10^{-1}$	$1{,}135 \times 10^{-9} \pm 3{,}635 \times 10^{-9}$
593_fri_c1_1000_10	MLP	$8{,}750 \times 10^{-1} \pm 2{,}356 \times 10^{-1}$	$1{,}267 \times 10^{0} \pm 1{,}362 \times 10^{0}$	$8{,}262 \times 10^{-3} \pm 1{,}009 \times 10^{-2}$
623_fri_c4_1000_10	MLP	$1{,}659 \times 10^{0} \pm 1{,}266 \times 10^{0}$	$1{,}039 \times 10^{0} \pm 9{,}721 \times 10^{-1}$	$1{,}455 \times 10^{-6} \pm 3{,}366 \times 10^{-6}$
627_fri_c2_500_10	MLP	$1{,}673 \times 10^{0} \pm 0{,}987 \times 10^{0}$	$9{,}300 \times 10^{-1} \pm 8{,}318 \times 10^{-1}$	$7{,}230 \times 10^{-6} \pm 1{,}327 \times 10^{-5}$
620_fri_c1_1000_25	MLP	$1{,}851 \times 10^{0} \pm 1{,}125 \times 10^{0}$	$1{,}909 \times 10^{0} \pm 1{,}251 \times 10^{0}$	$1{,}937 \times 10^{-7} \pm 7{,}024 \times 10^{-7}$
621_fri_c0_100_10	MLP	$2{,}835 \times 10^{0} \pm 2{,}019 \times 10^{0}$	$1{,}134 \times 10^{0} \pm 6{,}225 \times 10^{-1}$	$1{,}960 \times 10^{-6} \pm 2{,}119 \times 10^{-6}$
706_sleuth_case1202	MLP	$1{,}554 \times 10^{0} \pm 0{,}978 \times 10^{0}$	$7{,}613 \times 10^{3} \pm 5{,}639 \times 10^{3}$	$6{,}848 \times 10^{-11} \pm 3{,}536 \times 10^{-10}$
Resultado médio MLP		$6{,}745 \times 10^{1} \pm 6{,}321 \times 10^{2}$	$2{,}006 \times 10^{3} \pm 2{,}525 \times 10^{3}$	$1{,}781 \times 10^{-1} \pm 4{,}207 \times 10^{-1}$
MSE Médio		$9{,}463 \times 10^{3} \pm 5{,}589 \times 10^{3}$	$1{,}462 \times 10^{4} \pm 5{,}583 \times 10^{3}$	$2{,}757 \times 10^{-1} \pm 6{,}922 \times 10^{-1}$

Observa-se que o PDTX produziu menor MSE do que o LIME e a DT em todos os contextos estudados. Dessa forma, a fidelidade do PDTX supera os demais métodos não somente na média, mas também para todos os métodos ML individualmente.

5.3.2 Classificação

Em relação aos experimentos para problemas de classificação, a Tabela 9 apresenta os resultados obtidos. Nesse contexto, onde a fidelidade é mensurada por meio da acurácia entre o explicador e o método caixa-preta, observa-se que o PDTX apresentou melhor qualidade de aproximação quando comparado ao DT e LIME para o conjunto de dados estudado, bem como para todos os métodos de ML definidos.

No artigo gerado para a versão de classificação essa métrica também foi mensurada para outro conjunto de *bechmark*, veja Santos et al. (2021). Conforme apresentado naquele artigo, poucas avaliações da função objetivo na aplicação do jSO são necessárias para evoluir a PDT e alcançar valores relevantes para o PDTX, que são significativamente melhores que os obtidos pelo LIME e pela DT. Todavia, para se obter uma situação de estabilidade, mais avaliações são necessárias, para que assim o processo do otimização convirja para o ótimo global procurado, ou para a proximidade deste. O ótimo global é obtido quando o erro de classificação ou predição é nulo. Em ambos os *benchmarks* avaliados, os resultados obtidos superam significativamente àqueles obtidos pelo LIME e pela DT.

Os resultados aqui encontrados encontram-se em consonância com os apresentados em Santos et al. (2021) e em Ferreira et al. (2020). Essa diferença é esperada visto que a PDT possui melhor capacidade de aproximação local que uma DT tradicional, bem como no que se refere a modelos lineares simples, como aqueles utilizados pelo LIME. Além disso, o fato de o LIME não delimitar bem a região da vizinhança local, não adotando estratégias que garantam que a sub-região adotada será suficientemente pequena para um bom ajuste linear, reforçam o resultado encontrado. A Tabela 10 apresenta os resultados encontrados para a aplicação do teste de Wilcoxon, todas as comparações par a par apresentam diferenças significantes em termo de média, ao nível de confiança de 95%.

5.3.3 Fidelidade do PDTX com aplicação de seleção de características

O resultado da comparação entre as técnicas de seleção de características para problema de classificação são apresentados na sequência.

Conforme pode ser observado, as cinco técnicas retornaram resultados médios muito próximos um dos outros. Embora não possa ser comparada diretamente aos resultados originais de fidelidade produzidos pelo PDTX, apresentados na Tabela 4.3, visto que os

Tabela 9 – Fidelidade dos explicadores - Classificação (média ± desvio padrão)

Base de dados	ML	LIME	DT	PDTX
Adult	SVM	0,581 ± 0,172	0,632 ± 0,080	0,893 ± 0,082
Analcatdata_bankruptcy	SVM	0,473 ± 0,213	0,746 ± 0,125	0,999 ± 0,004
Breast_w	SVM	0,631 ± 0,129	0,767 ± 0,125	0,957 ± 0,053
Cleveland_nominal	SVM	0,593 ± 0,005	0,521 ± 0,212	0,895 ± 0,079
Heart_statlog	SVM	0,000 ± 0,000	0,707 ± 0,116	0,897 ± 0,034
Hepatitis	SVM	0,977 ± 0,064	0,633 ± 0,149	1,000 ± 0,000
Irish	SVM	0,408 ± 0,281	0,521 ± 0,209	0,919 ± 0,058
Tic_tac_toe	SVM	0,757 ± 0,198	0,463 ± 0,125	0,927 ± 0,075
Vowel	SVM	0,080 ± 0,159	0,294 ± 0,105	0,903 ± 0,094
Threeof9	SVM	0,408 ± 0,168	0,690 ± 0,109	0,908 ± 0,042
Resultado médio SVM		0,564 ± 0,171	0,597 ± 0,118	0,930 ± 0,033
Adult	RF	0,714 ± 0,144	0,778 ± 0,061	0,835 ± 0,060
Analcatdata_bankruptcy	RF	0,556 ± 0,270	0,766 ± 0,103	0,942 ± 0,070
Breast_w	RF	0,539 ± 0,409	0,811 ± 0,096	0,955 ± 0,040
Cleveland_nominal	RF	0,096 ± 0,079	0,779 ± 0,146	0,782 ± 0,159
Heart_statlog	RF	0,378 ± 0,331	0,756 ± 0,087	0,918 ± 0,056
Hepatitis	RF	0,817 ± 0,243	0,722 ± 0,118	0,937 ± 0,047
Irish	RF	0,404 ± 0,199	0,945 ± 0,058	0,943 ± 0,046
Tic_tac_toe	RF	0,564 ± 0,191	0,858 ± 0,065	0,838 ± 0,057
Vowel	RF	0,097 ± 0,095	0,506 ± 0,102	0,632 ± 0,051
Threeof9	RF	0,489 ± 0,278	0,999 ± 0,002	0,830 ± 0,047
Resultado médio RF		0,466 ± 0,182	0,792 ± 0,130	0,861 ± 0,037
Adult	MLP	0,591 ± 0,232	0,680 ± 0,080	0,903 ± 0,065
Analcatdata_bankruptcy	MLP	0,506 ± 0,262	0,708 ± 0,119	0,967 ± 0,028
Breast_w	MLP	0,388 ± 0,407	0,751 ± 0,128	0,990 ± 0,013
Cleveland_nominal	MLP	0,194 ± 0,171	0,582 ± 0,175	0,871 ± 0,107
Heart_statlog	MLP	0,495 ± 0,338	0,669 ± 0,108	0,979 ± 0,025
Hepatitis	MLP	0,864 ± 0,206	0,662 ± 0,123	0,998 ± 0,002
Irish	MLP	0,447 ± 0,231	0,868 ± 0,040	0,941 ± 0,047
Tic_tac_toe	MLP	0,605 ± 0,180	0,716 ± 0,065	0,933 ± 0,029
Vowel	MLP	0,104 ± 0,117	0,318 ± 0,109	0,537 ± 0,114
Threeof9	MLP	0,379 ± 0,245	0,800 ± 0,074	0,926 ± 0,048
Resultado médio MLP		0,457 ± 0,172	0,675 ± 0,131	0,905 ± 0,031
Média		0,496 ± 0,171	0,688 ± 0,107	0,899 ± 0,054

parâmetros de ajustes da PDT foram distintos, nota-se que os resultados apresentados por todos os métodos superam àqueles obtidos pela DT e pelo LIME. Sendo assim, para bases com muitos atributos, recomenda-se que uma dessas técnicas seja utilizada em conjunto com o PDTX. A Tabela 12 mostra que não existe diferenças significativas dentre os valores de fidelidade obtidos para os cinco métodos estudados.

Tabela 10 – Resultados para o teste de Wilcoxon Signed-Rank quanto à fidelidade local para problemas de classificação

Explicadores	p-value
PDTX vs. DT	7.325×10^{-6}
PDTX vs. LIME	$1{,}824 \times 10^{-6}$
DT vs. LIME	$1{,}358 \times 10^{-4}$

Tabela 11 – Resultados quanto à fidelidade com a utilização de métodos de seleção de características

Base de dados	**ML**	**Chi**	**Extra**	**RFE**	**LR**	**Boruta**
Adult	SVM	$0{,}801 \pm 0{,}054$	$0{,}799 \pm 0{,}059$	$0{,}788 \pm 0{,}051$	$0{,}782 \pm 0{,}044$	$0{,}804 \pm 0{,}047$
Analcatdata_bankruptcy	SVM	$0{,}949 \pm 0{,}045$	$0{,}943 \pm 0{,}045$	$0{,}941 \pm 0{,}045$	$0{,}935 \pm 0{,}049$	$0{,}921 \pm 0{,}055$
Breast_w	SVM	$0{,}900 \pm 0{,}046$	$0{,}898 \pm 0{,}033$	$0{,}896 \pm 0{,}040$	$0{,}896 \pm 0{,}038$	$0{,}906 \pm 0{,}034$
Cleveland_nominal	SVM	$0{,}765 \pm 0{,}051$	$0{,}763 \pm 0{,}045$	$0{,}759 \pm 0{,}049$	$0{,}763 \pm 0{,}058$	$0{,}754 \pm 0{,}064$
Heart_statlog	SVM	$0{,}821 \pm 0{,}052$	$0{,}825 \pm 0{,}044$	$0{,}812 \pm 0{,}056$	$0{,}800 \pm 0{,}057$	$0{,}815 \pm 0{,}039$
Hepatitis	SVM	$0{,}975 \pm 0{,}014$	$0{,}975 \pm 0{,}017$	$0{,}980 \pm 0{,}012$	$0{,}974 \pm 0{,}017$	$0{,}969 \pm 0{,}016$
Irish	SVM	$0{,}824 \pm 0{,}030$	$0{,}839 \pm 0{,}033$	$0{,}821 \pm 0{,}039$	$0{,}812 \pm 0{,}032$	$0{,}813 \pm 0{,}039$
Tic_tac_toe	SVM	$0{,}761 \pm 0{,}048$	$0{,}763 \pm 0{,}040$	$0{,}755 \pm 0{,}038$	$0{,}758 \pm 0{,}044$	$0{,}749 \pm 0{,}041$
Vowel	SVM	$0{,}593 \pm 0{,}157$	$0{,}579 \pm 0{,}172$	$0{,}568 \pm 0{,}171$	$0{,}568 \pm 0{,}172$	$0{,}563 \pm 0{,}174$
Threeof9	SVM	$0{,}775 \pm 0{,}043$	$0{,}780 \pm 0{,}041$	$0{,}780 \pm 0{,}038$	$0{,}777 \pm 0{,}040$	$0{,}769 \pm 0{,}048$
Resultado médio SVM		$0{,}816 \pm 0{,}077$	$0{,}816 \pm 0{,}080$	$0{,}810 \pm 0{,}080$	$0{,}807 \pm 0{,}078$	$0{,}806 \pm 0{,}079$
Adult	RF	$0{,}877 \pm 0{,}056$	$0{,}864 \pm 0{,}035$	$0{,}884 \pm 0{,}048$	$0{,}954 \pm 0{,}028$	$0{,}950 \pm 0{,}038$
Analcatdata_bankruptcy	RF	$0{,}862 \pm 0{,}042$	$0{,}859 \pm 0{,}044$	$0{,}861 \pm 0{,}043$	$0{,}879 \pm 0{,}050$	$0{,}859 \pm 0{,}045$
Breast_w	RF	$0{,}875 \pm 0{,}031$	$0{,}875 \pm 0{,}032$	$0{,}879 \pm 0{,}033$	$0{,}869 \pm 0{,}036$	$0{,}881 \pm 0{,}030$
Cleveland_nominal	RF	$0{,}749 \pm 0{,}066$	$0{,}737 \pm 0{,}072$	$0{,}737 \pm 0{,}071$	$0{,}742 \pm 0{,}069$	$0{,}741 \pm 0{,}063$
Heart_statlog	RF	$0{,}822 \pm 0{,}031$	$0{,}818 \pm 0{,}029$	$0{,}825 \pm 0{,}029$	$0{,}818 \pm 0{,}031$	$0{,}819 \pm 0{,}036$
Hepatitis	RF	$0{,}893 \pm 0{,}029$	$0{,}889 \pm 0{,}029$	$0{,}901 \pm 0{,}024$	$0{,}897 \pm 0{,}029$	$0{,}892 \pm 0{,}025$
Irish	RF	$0{,}832 \pm 0{,}036$	$0{,}823 \pm 0{,}028$	$0{,}839 \pm 0{,}033$	$0{,}877 \pm 0{,}034$	$0{,}888 \pm 0{,}033$
Tic_tac_toe	RF	$0{,}819 \pm 0{,}036$	$0{,}827 \pm 0{,}034$	$0{,}818 \pm 0{,}038$	$0{,}823 \pm 0{,}034$	$0{,}821 \pm 0{,}034$
Vowel	RF	$0{,}533 \pm 0{,}038$	$0{,}506 \pm 0{,}064$	$0{,}516 \pm 0{,}068$	$0{,}523 \pm 0{,}072$	$0{,}516 \pm 0{,}072$
Threeof9	RF	$0{,}810 \pm 0{,}027$	$0{,}816 \pm 0{,}033$	$0{,}813 \pm 0{,}025$	$0{,}813 \pm 0{,}022$	$0{,}807 \pm 0{,}028$
Resultado médio RF		$0{,}807 \pm 0{,}081$	$0{,}801 \pm 0{,}081$	$0{,}807 \pm 0{,}085$	$0{,}820 \pm 0{,}093$	$0{,}817 \pm 0{,}092$
Adult	MLP	$0{,}865 \pm 0{,}051$	$0{,}869 \pm 0{,}042$	$0{,}866 \pm 0{,}041$	$0{,}912 \pm 0{,}035$	$0{,}889 \pm 0{,}057$
Analcatdata_bankruptcy	MLP	$0{,}892 \pm 0{,}035$	$0{,}887 \pm 0{,}040$	$0{,}893 \pm 0{,}035$	$0{,}885 \pm 0{,}043$	$0{,}901 \pm 0{,}038$
Breast_w	MLP	$0{,}914 \pm 0{,}033$	$0{,}910 \pm 0{,}034$	$0{,}908 \pm 0{,}035$	$0{,}907 \pm 0{,}037$	$0{,}917 \pm 0{,}025$
Cleveland_nominal	MLP	$0{,}805 \pm 0{,}092$	$0{,}799 \pm 0{,}096$	$0{,}791 \pm 0{,}100$	$0{,}790 \pm 0{,}088$	$0{,}792 \pm 0{,}092$
Heart_statlog	MLP	$0{,}926 \pm 0{,}024$	$0{,}915 \pm 0{,}025$	$0{,}922 \pm 0{,}029$	$0{,}927 \pm 0{,}027$	$0{,}909 \pm 0{,}030$
Hepatitis	MLP	$0{,}937 \pm 0{,}021$	$0{,}933 \pm 0{,}019$	$0{,}943 \pm 0{,}017$	$0{,}933 \pm 0{,}017$	$0{,}927 \pm 0{,}017$
Irish	MLP	$0{,}824 \pm 0{,}039$	$0{,}839 \pm 0{,}038$	$0{,}825 \pm 0{,}035$	$0{,}819 \pm 0{,}055$	$0{,}844 \pm 0{,}045$
Tic_tac_toe	MLP	$0{,}832 \pm 0{,}033$	$0{,}833 \pm 0{,}039$	$0{,}834 \pm 0{,}033$	$0{,}832 \pm 0{,}030$	$0{,}819 \pm 0{,}047$
Vowel	MLP	$0{,}490 \pm 0{,}062$	$0{,}496 \pm 0{,}061$	$0{,}481 \pm 0{,}056$	$0{,}496 \pm 0{,}065$	$0{,}477 \pm 0{,}076$
Threeof9	MLP	$0{,}874 \pm 0{,}039$	$0{,}875 \pm 0{,}040$	$0{,}876 \pm 0{,}039$	$0{,}875 \pm 0{,}037$	$0{,}879 \pm 0{,}036$
Resultado médio MLP		$0{,}833 \pm 0{,}073$	$0{,}832 \pm 0{,}075$	$0{,}830 \pm 0{,}077$	$0{,}829 \pm 0{,}086$	$0{,}829 \pm 0{,}085$
Média		$0{,}820 \pm 0{,}110$	$0{,}818 \pm 0{,}112$	$0{,}817 \pm 0{,}115$	$0{,}821 \pm 0{,}115$	$0{,}820 \pm 0{,}117$

5.4 Estabilidade

De forma semelhante, a estabilidade foi calculada para os contextos de classificação e regressão por meio da Distância de Tau. Nas subseções seguintes encontram-se os resultados médios obtidos nessa avaliação.

Tabela 12 – Resultados para o teste de Wilcoxon Signed-Rank quanto à fidelidade local com a aplicação de seleção de características

Métodos de seleção de características	p-value
Chi vs. Extra	0.205
Chi vs. RFE	0.117
Chi vs. Lr embutido	0.191
Chi vs. Boruta	0.181
Extra vs. RFE	0.589
Extra vs. Lr embutido	0.741
Extra vs. Boruta	0.410
RFE vs. Lr embutido	0.991
RFE vs. Boruta	0.869
Lr embutido vs. Boruta	0.394

5.4.1 Regressão

Conforme pode ser observado, o PDTX retornou o menor valor médio de Distância de Tau para as bases testadas, quando comparado os demais explicadores, para todos os métodos ML avaliados, o que indica que a metodologia proposta é mais estável. Ou seja, houve menos variações da hierarquia de atributos retornadas para um mesmo contexto experimental utilizado o PDTX, que os outros métodos.

5.4.2 Classificação

No contexto de classificação também foi observado que

A Tabela 15 apresenta os resultados quanto à estabilidade dos explicadores estudados. Nota-se que, ao nível de confiança de 95% não foi possível refutar a hipótese nula de que as populações da DT e do PDTX provém de diferentes distribuições. Por outro lado, as diferenças observadas nas comparações par a par entre LIME e PDTX e LIME e DT apresentam $p < 0{,}05$, que mostram que as diferenças observadas são significativas.

5.5 Simplicidade

No que se refere à simplicidade, a Tabela 16 sumariza os resultados encontrados para as bases de dado de classificação.

Os resultados apresentados nas Tabelas 16 e 18 foram normalizados por meio da divisão individual pelo maior comprimento de regra obtido para cada execução de um mesmo método, numa mesma base. De acordo com o critério adotado pela literatura quanto ao comprimento de regras, a quantidade de regras obtidas utilizando as partições oblíquas da literatura diretamente mostra-se mais simples, segundo essa definição. Porém,

Tabela 13 – Estabilidade dos explicadores via Distância de Tau - Regressão

Base de dados	**ML**	**LIME**	**DT**	**PDTX**
227_cpu_small	SVM	0,382 ± 0,099	0,181 ± 0,156	0,024 ± 0,059
229_pwLinear	SVM	0,162 ± 0,061	0,067 ± 0,091	0,000 ± 0,000
230_machine_cpu	SVM	0,329 ± 0,179	0,231 ± 0,249	0,036 ± 0,059
522_pm10	SVM	0,456 ± 0,164	0,117 ± 0,137	0,000 ± 0,000
593_fri_c1_1000_10	SVM	0,430 ± 0,105	0,076 ± 0,052	0,000 ± 0,000
623_fri_c4_1000_10	SVM	0,382 ± 0,113	0,093 ± 0,062	0,124 ± 0,113
627_fri_c2_500_10	SVM	0,380 ± 0,121	0,102 ± 0,061	0,104 ± 0,119
621_fri_c0	SVM	0,300 ± 0,105	0,227 ± 0,115	0,000 ± 0,000
659_sleuth_ex1714	SVM	0,351 ± 0,118	0,225 ± 0,158	0,121 ± 0,124
706_sleuth_case1202	SVM	0,340 ± 0,153	0,067 ± 0,133	0,062 ± 0,107
Resultado médio SVM		0,351 ± 0,055	0,139 ± 0,062	0,047 ± 0,045
227_cpu_small	RF	0,359 ± 0,075	0,147 ± 0,126	0,144 ± 0,186
229_pwLinear	RF	0,281 ± 0,096	0,110 ± 0,112	0,104 ± 0,055
230_machine_cpu	RF	0,327 ± 0,178	0,240 ± 0,272	0,258 ± 0,168
522_pm10	RF	0,329 ± 0,156	0,124 ± 0,148	0,067 ± 0,111
593_fri_c1_1000_10	RF	0,294 ± 0,127	0,039 ± 0,026	0,256 ± 0,078
621_fri_c0_100_10	RF	0,311 ± 0,090	0,167 ± 0,126	0,156 ± 0,000
623_fri_c4_1000_10	RF	0,365 ± 0,102	0,071 ± 0,056	0,168 ± 0,078
627_fri_c2_500_10	RF	0,349 ± 0,103	0,101 ± 0,075	0,067 ± 0,000
620_fri_c1_1000_25	RF	0,423 ± 0,069	0,273 ± 0,062	0,246 ± 0,071
659_sleuth_ex1714	RF	0,349 ± 0,175	0,222 ± 0,149	0,138 ± 0,103
706_sleuth_case1202	RF	0,353 ± 0,142	0,098 ± 0,131	0,040 ± 0,087
Resultado médio RF		0,339 ± 0,053	0,149 ± 0,058	0,160 ± 0,052
227_cpu_small	MLP	0,407 ± 0,088	0,133 ± 0,136	0,065 ± 0,125
229_pwLinear	MLP	0,180 ± 0,061	0,08 ± 0,079	0,010 ± 0,022
230_machine_cpu	MLP	0,371 ± 0,177	0,311 ± 0,282	0,085 ± 0,086
522_pm10	MLP	0,349 ± 0,170	0,073 ± 0,128	0,010 ± 0,036
593_fri_c1_1000_10	MLP	0,356 ± 0,091	0,084 ± 0,05	0,376 ± 0,232
623_fri_c4_1000_10	MLP	0,342 ± 0,135	0,071 ± 0,057	0,044 ± 0,115
627_fri_c2_500_10	MLP	0,353 ± 0,134	0,079 ± 0,061	0,050 ± 0,062
620_fri_c1_1000_25	MLP	0,411 ± 0,076	0,293 ± 0,044	0,235 ± 0,078
621_fri_c0_100_10	MLP	0,296 ± 0,088	0,154 ± 0,109	0,019 ± 0,034
659_sleuth_ex1714	MLP	0,419 ± 0,170	0,200 ± 0,181	0,000 ± 0,000
706_sleuth_case1202	MLP	0,278 ± 0,157	0,124 ± 0,220	0,019 ± 0,049
Resultado médio MLP		0,341 ± 0,041	0,142 ± 0,054	0,099 ± 0,049
Média		0,344 ± 0,122	0,143 ± 0,120	0,095 ± 0,074

este formato, embora interpretável, por mostrar uma relação matemática que associe cada entrada a cada saída, parece ser menos explicável que os demais avaliados, por isso, ele não foi considerado nessa avaliação. Por outro lado, foi considerado todas as regras geradas no caminho da predição de x, após eliminação das redundâncias. Avaliando dessa maneira,

Tabela 14 – Estabilidade dos explicadores via Distância de Tau - Classificação

Base de dados	**ML**	**LIME**	**DT**	**PDTX**
Adult	SVM	0,581 ± 0,172	0,054 ± 0,063	0,219 ± 0,095
Analcatdata_bankruptcy	SVM	0,271 ± 0,160	0,071 ± 0,147	0,195 ± 0,204
Breast_w	SVM	0,218 ± 0,092	0,244 ± 0,120	0,131 ± 0,148
Cleveland_nominal	SVM	0,268 ± 0,158	0,197 ± 0,147	0,088 ± 0,107
Heart_statlog	SVM	0,378 ± 0,115	0,289 ± 0,127	0,110 ± 0,049
Hepatitis	SVM	0,360 ± 0,079	0,286 ± 0,166	0,188 ± 0,102
Irish	SVM	0,258 ± 0,161	0,000 ± 0,000	0,133 ± 0,131
Tic_tac_toe	SVM	0,284 ± 0,094	0,259 ± 0,210	0,220 ± 0,146
ThreeOf9	SVM	0,402 ± 0,139	0,156 ± 0,138	0,086 ± 0,087
Vowel	SVM	0,310 ± 0,095	0,172 ± 0,098	0,358 ± 0,106
Resultado médio SVM		0,333 ± 0,078	0,173 ± 0,082	0,173 ± 0,063
Adult	RF	0,714 ± 0,144	0,051 ± 0,076	0,196 ± 0,104
Analcatdata_bankruptcy	RF	0,219 ± 0,119	0,124 ± 0,202	0,174 ± 0,169
Breast_w	RF	0,271 ± 0,118	0,293 ± 0,178	0,171 ± 0,226
Cleveland_nominal	RF	0,308 ± 0,123	0,210 ± 0,186	0,114 ± 0,116
Heart_statlog	RF	0,201 ± 0,070	0,279 ± 0,117	0,114 ± 0,094
Hepatitis	RF	0,284 ± 0,087	0,346 ± 0,138	0,175 ± 0,113
Irish	RF	0,315 ± 0,171	0,000 ± 0,000	0,133 ± 0,131
Tic_tac_toe	RF	0,427 ± 0,159	0,272 ± 0,113	0,124 ± 0,096
ThreeOf9	RF	0,276 ± 0,107	0,145 ± 0,073	0,107 ± 0,093
Vowel	RF	0,303 ± 0,080	0,209 ± 0,230	0,350 ± 0,139
Resultado médio RF		0,332 ± 0,094	0,193 ± 0,083	0,166 ± 0,061
Adult	MLP	0,591 ± 0,232	0,066 ± 0,075	0,202 ± 0,078
Analcatdata_bankruptcy	MLP	0,237 ± 0,144	0,178 ± 0,229	0,143 ± 0,104
Breast_w	MLP	0,240 ± 0,092	0,209 ± 0,167	0,187 ± 0,211
Cleveland_nominal	MLP	0,104 ± 0,081	0,098 ± 0,126	0,108 ± 0,130
Heart_statlog	MLP	0,336 ± 0,105	0,253 ± 0,071	0,167 ± 0,132
Hepatitis	MLP	0,337 ± 0,082	0,356 ± 0,136	0,201 ± 0,128
Irish	MLP	0,240 ± 0,175	0,000 ± 0,000	0,111 ± 0,126
Tic_tac_toe	MLP	0,366 ± 0,116	0,285 ± 0,210	0,114 ± 0,094
ThreeOf9	MLP	0,213 ± 0,105	0,163 ± 0,256	0,078 ± 0,085
Vowel	MLP	0,339 ± 0,091	0,151 ± 0,099	0,342 ± 0,107
Resultado médio MLP		0,268 ± 0,098	0,188 ± 0,077	0,161 ± 0,059
Média		0,322 ± 0,122	0,181 ± 0,130	0,168 ± 0,122

o melhor resultado encontrado foi com a DT, seguido do PDTX, e por último o LIME, que desses três teve o pior resultado para esta métrica, para os problemas de classificação, e da DT, LIME e PDTX para regressão. Como os resultados do LIME e do PDTX são próximos em ambos os casos, a avaliação estatística torna-se interessante.

Conforme Tabela 17, nota-se que a hipótese nula do teste de Wilcocon não foi

Tabela 15 – Resultados para o teste de Wilcoxon Signed-Rank quanto à estabilidade local para problemas de classificação

Explicadores	p-value
PDTX vs. DT	$7{,}112 \times 10^{-1}$
PDTX vs. LIME	6.039×10^{-6}
DT vs. LIME	3.561×10^{-5}

refutada nas comparações entre LIME e PDTX, ao nível de confiança de 95%. Assim, pode-se presumir que a simplicidade dos dois explicadores é comparável entre si, e ambas são menos simples que a DT.

Os resultados encontrados pela análise estatística para regressão, mostrados na Tabela 19, é análogo ao encontrado para classificação, apresentados em 17, que mostra que a DT se apresenta mais simples que os demais explicadores estudados, e que o LIME e PDTX possuem simplicidade média semelhante.

5.6 Resultados – Avaliação Qualitativa

Para esse estudo foram escolhidos quatro casos de usos para averiguação dos resultados visuais apresentados por cada explicador, a saber: Diagnóstico de Lupus, Diagnóstico de câncer de mama, Previsão de falência e Predição de diabetes.

5.6.1 Caso de uso problema de classificação: Diagnóstico de Lupus

Nessa seção, serão apresentados exemplos quanto ao funcionamento do PDTX, em que as saídas são comparadas visualmente àquelas geradas por outros métodos de explicação possíveis. Foi escolhida aleatoriamente uma amostra da base de dados Lupus, pertencente ao PMLB, para primeira demonstração do PDTX. Essa base possui três atributos: TIME, DURATION e LOG. O primeiro atributo refere-se ao tempo desde o lúpus até o diagnóstico (biópsia), o segundo tem relação com a duração desde o diagnóstico (biópsia) até a morte, e o último calcula o logaritmo de (1 + DURAÇÃO). Para o exemplo apresentado essas características foram renomeadas para x_1, x_2 e x_3, respectivamente. Trata-se de um problema binário com duas classes possíveis, Classe 0 correspondente a "Vivo" e Classe 1 referente a "Morto".

Foi treinado um modelo *Random Forest* com 1000 árvores, utilizando-se para tal a biblioteca *Scikit Learn* (Buitinck et al., 2013), onde a base de treino foi dividida na proporção de 80% dos dados para treino, e o demais para teste. Escolheu-se uma amostra aleatória para explicar a predição do RF para esta amostra. A partir deste modelo

Tabela 16 – Simplicidade dos explicadores via Comprimento de regras - Classificação

Base de dados	**ML**	**LIME**	**DT**	**PDTX**
Analcatdata_bankruptcy	SVM	$0{,}997 \pm 0{,}018$	$0{,}336 \pm 0{,}102$	$0{,}686 \pm 0{,}138$
Breast_w	SVM	$0{,}867 \pm 0{,}101$	$0{,}473 \pm 0{,}105$	$0{,}960 \pm 0{,}073$
Cleveland_nominal	SVM	$0{,}836 \pm 0{,}156$	$0{,}709 \pm 0{,}211$	$0{,}922 \pm 0{,}122$
Heart_statlog	SVM	$0{,}989 \pm 0{,}036$	$0{,}324 \pm 0{,}073$	$0{,}833 \pm 0{,}144$
Hepatitis	SVM	$0{,}948 \pm 0{,}070$	$0{,}240 \pm 0{,}055$	$0{,}897 \pm 0{,}104$
Irish	SVM	$0{,}902 \pm 0{,}099$	$0{,}435 \pm 0{,}058$	$0{,}939 \pm 0{,}080$
Tic_tac_toe	SVM	$0{,}978 \pm 0{,}046$	$0{,}586 \pm 0{,}164$	$0{,}844 \pm 0{,}117$
Vowel	SVM	$0{,}933 \pm 0{,}083$	$0{,}383 \pm 0{,}136$	$0{,}924 \pm 0{,}115$
Threeof9	SVM	$0{,}910 \pm 0{,}093$	$0{,}440 \pm 0{,}123$	$0{,}942 \pm 0{,}083$
Resultado médio SVM		$0{,}934 \pm 0{,}044$	$0{,}425 \pm 0{,}104$	$0{,}879 \pm 0{,}062$
Analcatdata_bankruptcy	RF	$0{,}980 \pm 0{,}046$	$0{,}325 \pm 0{,}089$	$0{,}845 \pm 0{,}166$
Breast_w	RF	$0{,}919 \pm 0{,}108$	$0{,}472 \pm 0{,}099$	$0{,}909 \pm 0{,}091$
Cleveland_nominal	RF	$0{,}883 \pm 0{,}126$	$0{,}738 \pm 0{,}180$	$0{,}906 \pm 0{,}117$
Heart_statlog	RF	$0{,}988 \pm 0{,}036$	$0{,}317 \pm 0{,}063$	$0{,}825 \pm 0{,}138$
Hepatitis	RF	$0{,}883 \pm 0{,}101$	$0{,}215 \pm 0{,}048$	$0{,}940 \pm 0{,}077$
Irish	RF	$0{,}895 \pm 0{,}128$	$0{,}384 \pm 0{,}096$	$0{,}939 \pm 0{,}080$
Tic_tac_toe	RF	$0{,}971 \pm 0{,}056$	$0{,}547 \pm 0{,}169$	$0{,}869 \pm 0{,}111$
Vowel	RF	$0{,}873 \pm 0{,}111$	$0{,}373 \pm 0{,}125$	$0{,}983 \pm 0{,}043$
Threeof9	RF	$1{,}000 \pm 0{,}000$	$0{,}491 \pm 0{,}069$	$0{,}455 \pm 0{,}000$
Resultado médio RF		$0{,}935 \pm 0{,}039$	$0{,}445 \pm 0{,}097$	$0{,}853 \pm 0{,}037$
Analcatdata_bankruptcy	MLP	$0{,}983 \pm 0{,}038$	$0{,}356 \pm 0{,}087$	$0{,}826 \pm 0{,}159$
Breast_w	MLP	$0{,}955 \pm 0{,}091$	$0{,}553 \pm 0{,}103$	$0{,}878 \pm 0{,}082$
Cleveland_nominal	MLP	$0{,}874 \pm 0{,}150$	$0{,}732 \pm 0{,}201$	$0{,}876 \pm 0{,}129$
Heart_statlog	MLP	$0{,}983 \pm 0{,}043$	$0{,}309 \pm 0{,}057$	$0{,}872 \pm 0{,}131$
Hepatitis	MLP	$0{,}962 \pm 0{,}065$	$0{,}230 \pm 0{,}061$	$0{,}872 \pm 0{,}099$
Irish	MLP	$0{,}943 \pm 0{,}106$	$0{,}414 \pm 0{,}088$	$0{,}917 \pm 0{,}083$
Tic_tac_toe	MLP	$0{,}966 \pm 0{,}057$	$0{,}558 \pm 0{,}132$	$0{,}892 \pm 0{,}124$
Vowel	MLP	$0{,}827 \pm 0{,}114$	$0{,}364 \pm 0{,}117$	$0{,}977 \pm 0{,}079$
Threeof9	MLP	$1{,}000 \pm 0{,}000$	$0{,}509 \pm 0{,}093$	$0{,}455 \pm 0{,}000$
Resultado médio MLP		$0{,}945 \pm 0{,}038$	$0{,}403 \pm 0{,}129$	$0{,}835 \pm 0{,}033$
Média		$0{,}935 \pm 0{,}102$	$0{,}438 \pm 0{,}184$	$0{,}859 \pm 0{,}167$

Tabela 17 – Resultados para o teste de Wilcoxon Signed-Rank quanto à simplicidade local para problemas de classificação

Explicadores	p-value
PDTX vs. DT	0,001
PDTX vs. LIME	0,06
DT vs. LIME	0,001

pré-treinado, foram gerados 150 pontos ao redor de x, seguindo as distribuição Hammesley a partir da biblioteca *Sckopt*, do pacote *Scikit-optimize* (Buitinck et al., 2013). A Figura 5.6.1 apresenta os resultados obtidos.

Conforme pode ser observado, a árvore gerada neste exemplo obteve fidelidade igual a 1.0 ou 100% de acerto. O PDTX gerou como saída a saída "Classe 1", para a amostra em questão, seguindo o caminho direito da árvore gerada. A árvore de tamanho 3 inicialmente gerada, foi podada o que resultou em uma árvore com somente um hiperplano, apresentado na Figura 32. Este hiperplano mapeia claramente uma regra matemática local que pode ser utilizada para predizer outras amostras nessa mesma sub-região. A partir do caminho gerado, gera-se todas as regras associadas à sub-região mapeada, que nesse contexto foi $x_1 \leq 0{,}344$, $x_2 > -25{,}825$ e $x_3 > -3{,}558$. Essas regras em conjunto delimitam a região a qual x se encontra. Tais valores estão coerentes com os atribuídos a x. A árvore do canto inferior esquerdo apresenta ao usuário possibilidades de se alcançar a "Classe 0". Por exemplo, se x_1 for alterado para um valor maior que 0,344, a "Classe 0". Com essa informação, o especialista pode entender se o critério utilizado pelo caixa preta para avaliar se a pessoa com Lupus sobreviveu ou não, e se esse diagnóstico encontra-se correto, de acordo com os conhecimentos clínicos.

Verificando o LIME, Figura 33, observa-se que os mesmos graus de hierarquia de características foram obtidos. Os limite de decisão, embora diferentes, especialmente pelo tamanho do raio da vizinhança local distintas entre o LIME e o PDTX, encontram-se em concordância.

Em relação ao SHAP, também ficou evidenciado a alta importância de x_1 em relações aos demais recursos localmente. O SHAP retornou que x_2 e x_3 possuem importância local nula. Enquanto o LIME e o PDTX retornaram valores próximos de zero, mas não negativo.

Em relação à DT utilizada, que utilizou somente x_1 para fazer a partição do espaço, e obteve fidelidade de 0.933 nessa aproximação, o valor de *threshold* associado à x_1 adotado nesse processo encontra-se coerente com o obtido pelo PDTX. Além disso, a hierarquia de importância dos atributos também encontra-se coerente.

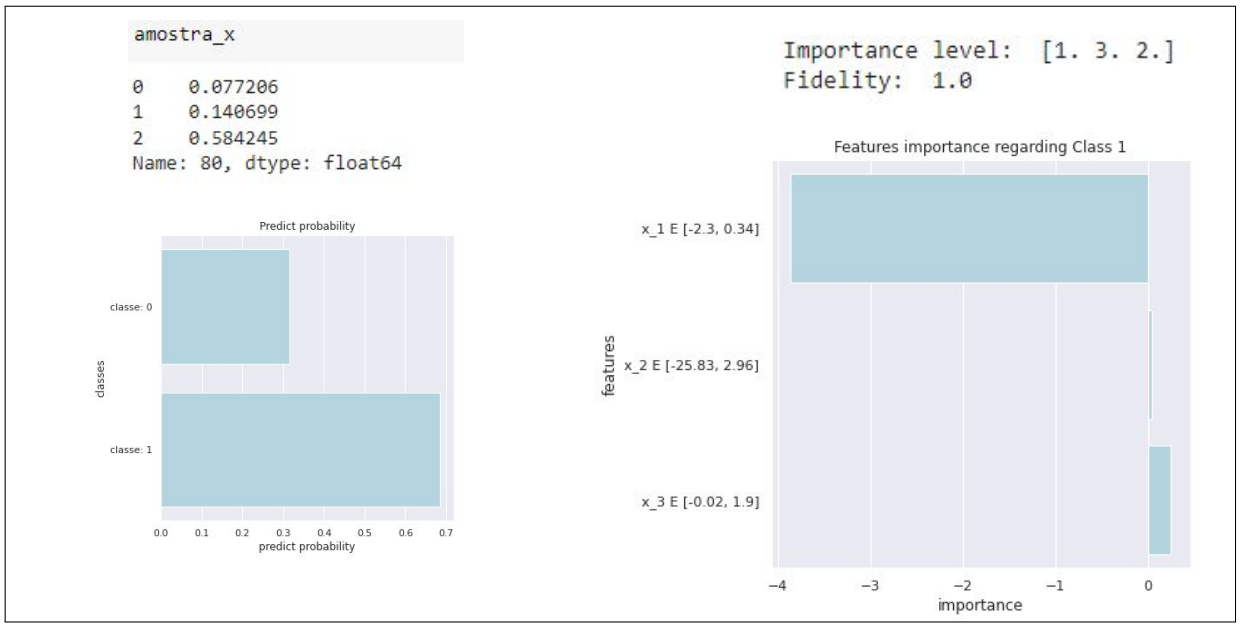
Na sequência, apresenta-se as explicações geradas para outro problema muito

Tabela 18 – Simplicidade dos explicadores via Comprimento de regras - Regressão

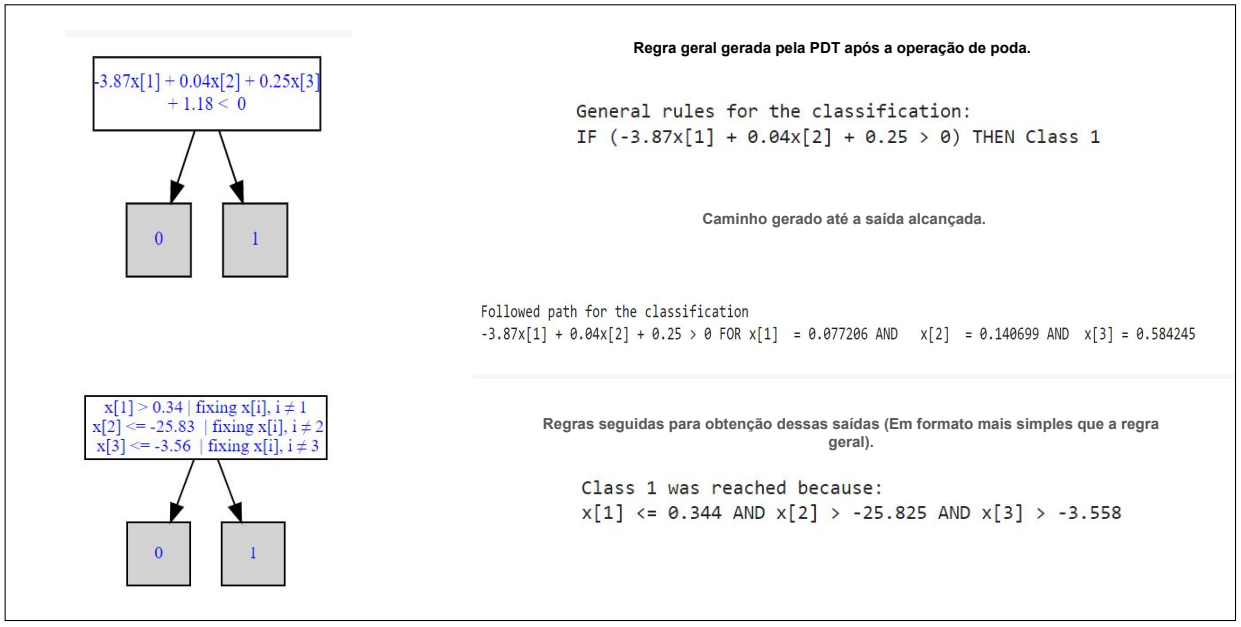
Base de dados	**ML**	**LIME**	**DT**	**PDTX**
227_cpu_small	SVM	$0{,}955 \pm 0{,}080$	$0{,}331 \pm 0{,}095$	$0{,}928 \pm 0{,}094$
229_pwlinear	SVM	$0{,}977 \pm 0{,}047$	$0{,}333 \pm 0{,}079$	$0{,}905 \pm 0{,}098$
230_machine_cpu	SVM	$0{,}599 \pm 0{,}137$	$0{,}401 \pm 0{,}095$	$1{,}000 \pm 0{,}000$
522_pm10	SVM	$0{,}713 \pm 0{,}124$	$0{,}483 \pm 0{,}146$	$0{,}997 \pm 0{,}014$
574_house_16h	SVM	$1{,}000 \pm 0{,}000$	$0{,}268 \pm 0{,}064$	$0{,}629 \pm 0{,}104$
593_fri_c1_1000_10	SVM	$0{,}944 \pm 0{,}085$	$0{,}429 \pm 0{,}101$	$0{,}938 \pm 0{,}076$
620_fri_c1_1000_25	SVM	$1{,}000 \pm 0{,}000$	$0{,}182 \pm 0{,}044$	$0{,}412 \pm 0{,}044$
621_fri_c0_100_10	SVM	$0{,}951 \pm 0{,}068$	$0{,}437 \pm 0{,}159$	$0{,}933 \pm 0{,}082$
627_fri_c2_500_10	SVM	$0{,}949 \pm 0{,}068$	$0{,}445 \pm 0{,}111$	$0{,}928 \pm 0{,}092$
659_sleuth_ex1714	SVM	$0{,}722 \pm 0{,}163$	$0{,}488 \pm 0{,}132$	$0{,}995 \pm 0{,}019$
706_sleuth_case1202	SVM	$0{,}609 \pm 0{,}148$	$0{,}440 \pm 0{,}123$	$1{,}000 \pm 0{,}000$
Resultado médio SVM		$0{,}856 \pm 0{,}142$	$0{,}385 \pm 0{,}078$	$0{,}879 \pm 0{,}130$
227_cpu_small	RF	$0{,}906 \pm 0{,}113$	$0{,}378 \pm 0{,}059$	$0{,}935 \pm 0{,}087$
229_pwlinear	RF	$0{,}980 \pm 0{,}042$	$0{,}556 \pm 0{,}118$	$0{,}910 \pm 0{,}111$
230_machine_cpu	RF	$0{,}628 \pm 0{,}131$	$0{,}588 \pm 0{,}131$	$1{,}000 \pm 0{,}000$
522_pm10	RF	$0{,}700 \pm 0{,}106$	$0{,}503 \pm 0{,}116$	$1{,}000 \pm 0{,}000$
574_house_16h	RF	$1{,}000 \pm 0{,}000$	$0{,}293 \pm 0{,}080$	$0{,}656 \pm 0{,}100$
593_fri_c1_1000_10	RF	$0{,}919 \pm 0{,}107$	$0{,}515 \pm 0{,}118$	$0{,}923 \pm 0{,}100$
620_fri_c1_1000_25	RF	$1{,}000 \pm 0{,}000$	$0{,}206 \pm 0{,}058$	$0{,}407 \pm 0{,}045$
621_fri_c0_100_10	RF	$0{,}960 \pm 0{,}066$	$0{,}529 \pm 0{,}133$	$0{,}915 \pm 0{,}080$
627_fri_c2_500_10	RF	$0{,}956 \pm 0{,}063$	$0{,}515 \pm 0{,}105$	$0{,}949 \pm 0{,}081$
659_sleuth_ex1714	RF	$0{,}719 \pm 0{,}111$	$0{,}447 \pm 0{,}114$	$1{,}000 \pm 0{,}000$
706_sleuth_case1202	RF	$0{,}606 \pm 0{,}119$	$0{,}558 \pm 0{,}138$	$1{,}000 \pm 0{,}000$
Resultado médio RF		$0{,}852 \pm 0{,}137$	$0{,}463 \pm 0{,}096$	$0{,}881 \pm 0{,}127$
227_cpu_small	MLP	$0{,}927 \pm 0{,}113$	$0{,}344 \pm 0{,}093$	$0{,}939 \pm 0{,}080$
229_pwlinear	MLP	$0{,}981 \pm 0{,}045$	$0{,}362 \pm 0{,}112$	$0{,}930 \pm 0{,}076$
230_machine_cpu	MLP	$0{,}607 \pm 0{,}125$	$0{,}408 \pm 0{,}125$	$1{,}000 \pm 0{,}000$
522_pm10	MLP	$0{,}688 \pm 0{,}102$	$0{,}438 \pm 0{,}121$	$1{,}000 \pm 0{,}000$
574_house_16h	MLP	$1{,}000 \pm 0{,}000$	$0{,}285 \pm 0{,}082$	$0{,}647 \pm 0{,}110$
593_fri_c1_1000_10	MLP	$0{,}945 \pm 0{,}092$	$0{,}456 \pm 0{,}134$	$0{,}947 \pm 0{,}067$
620_fri_c1_1000_25	MLP	$1{,}000 \pm 0{,}000$	$0{,}184 \pm 0{,}053$	$0{,}414 \pm 0{,}052$
621_fri_c0_100_10	MLP	$0{,}944 \pm 0{,}095$	$0{,}471 \pm 0{,}083$	$0{,}944 \pm 0{,}072$
627_fri_c2_500_10	MLP	$0{,}931 \pm 0{,}083$	$0{,}438 \pm 0{,}119$	$0{,}932 \pm 0{,}088$
659_sleuth_ex1714	MLP	$0{,}676 \pm 0{,}139$	$0{,}408 \pm 0{,}111$	$1{,}000 \pm 0{,}000$
706_sleuth_case1202	MLP	$0{,}597 \pm 0{,}120$	$0{,}432 \pm 0{,}109$	$1{,}000 \pm 0{,}000$
Resultado médio MLP		$0{,}845 \pm 0{,}148$	$0{,}384 \pm 0{,}066$	$0{,}887 \pm 0{,}130$
Média		$0{,}851 \pm 0{,}180$	$0{,}411 \pm 0{,}150$	$0{,}882 \pm 0{,}190$

Figura 32 – Explicações geradas pelo PDTX

(a) PDTX - Importância dos Atributos - à direita



(b) PDTX - Regras de predição

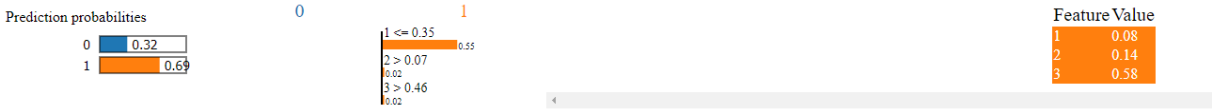


Fonte: Os autores.

Tabela 19 – Resultados para o teste de Wilcoxon Signed-Rank quanto à simplicidade local para problemas de regressão

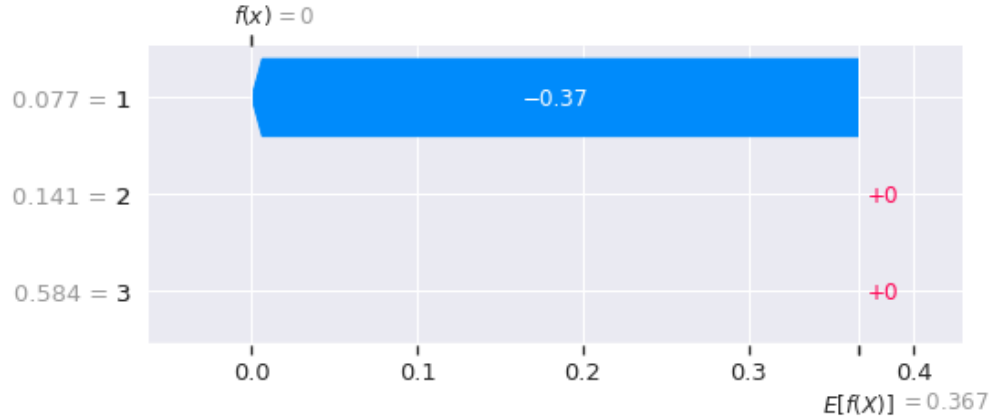
Explicadores	p-value
PDTX vs. DT	0,001
PDTX vs. LIME	0.679
DT vs. LIME	0,001

Figura 33 – Lime - Explicação



Fonte: Os autores

Figura 34 – SHAP - Explicação



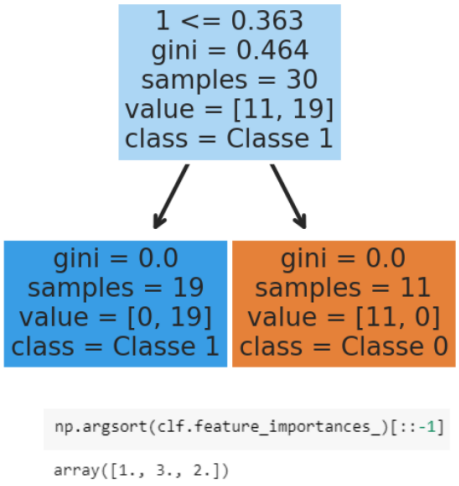
Fonte: Os autores

comum na área da saúde, o câncer de mama.

5.6.2 Diagnóstico de câncer de mama

Neste novo exemplo foi utilizada a base "Breast" também pertencente ao PMLB. Assim como o exemplo anterior, a aplicação de métodos de explicação de decisão para aplicações na área da saúde é essencial para implantação deste tipo de sistema, bem como para manutenção deste, já que propiciam aos especialistas averiguarem se os motivos pelos quais as saídas foram alcançadas encontram-se em consonância com o esperado. Essa base possui 9 atributos, correspondentes a espessura do aglomerado, uniformidade do tamanho da célula, uniformidade do formato da célula, adesão marginal, tamanho da célula epitelial única, núcleos desencapados, cromatina suave, nucléolos normais e mitoses,

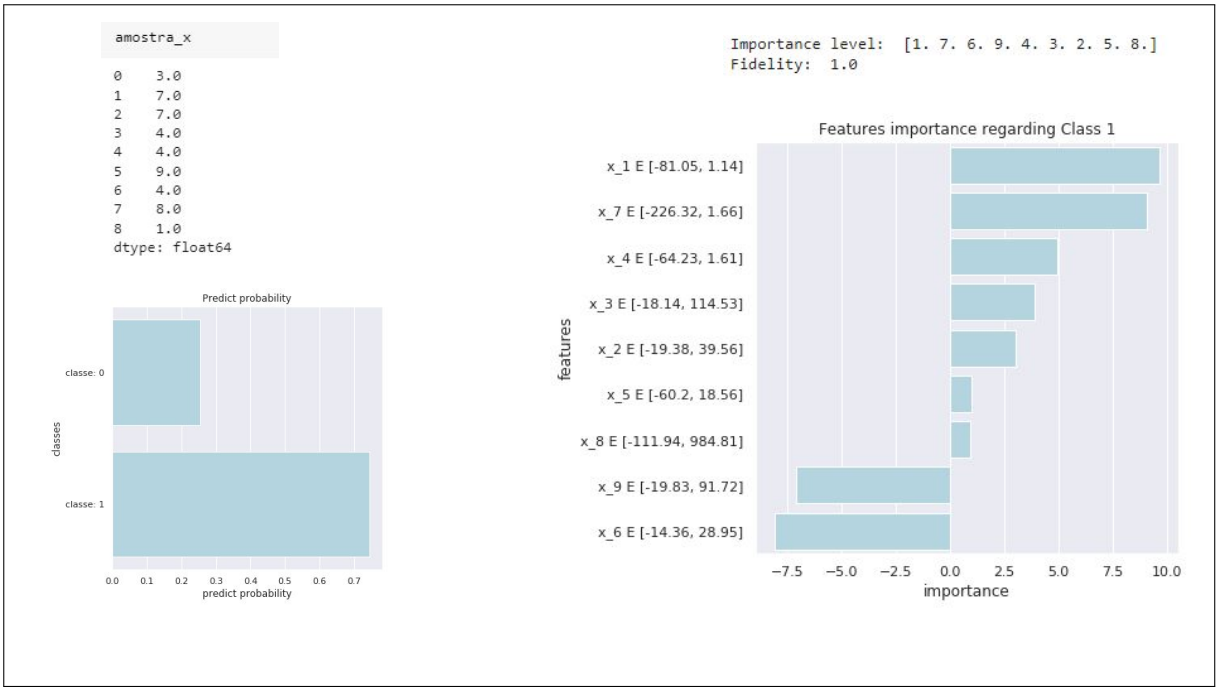
Figura 35 – DT - Explicação



Fonte: Os autores

os quais tiveram os nomes reais substituídos por x_i com $i \in \{1,2,...9\}$. Para este exemplo o PDTX também retornou 100% de fidelidade.

Figura 36 – PDTX - Explicação base breast



Fonte: Os autores

De acordo com o apresentado na Figura 36, o atributo x_1 é o que tem maior importância para a classificação gerada, sendo que essa relevância é positiva, ou seja, se o valor de x_1 for aumentado, espera-se que a probabilidade de predição para a referida classe também aumente. Os resultados apresentados pela DT e SHAP estão em concordância de

que este é o atributo mais importante, como pode ser visto nas Figuras 38 e 39. Em relação ao segundo atributo mais importante, o PDTX e DT encontram-se em concordância de que trata-se do recurso x_7.

Figura 37 – DT - Explicação base breast

Fonte: Os autores

Figura 38 – DT - Fidelidade e importância dos atributos

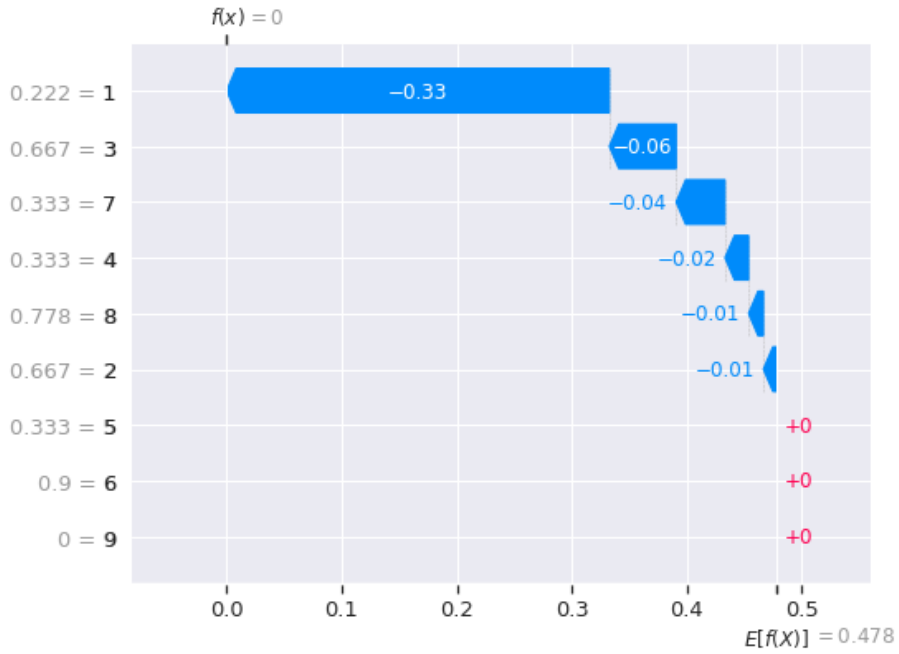
```
0.9888888888888889

array([1., 7., 3., 2., 4., 8., 9., 6., 5.])
```

Fonte: Os autores

Para o SHAP este é o terceiro atributo mais importante, já que para ele o

segundo mais importante é o x_3, conforme Figura 39. Ou seja, são notadas concordâncias e divergências entre as ordens de classificação proferidas entre os explicadores. Algumas diferenças são esperadas, devido ao processo de interno de computação ser distinto entre os métodos apresentados.

Figura 39 – SHAP - Explicação base breast



Fonte: Os autores

O LIME também utiliza regularização para zerar o peso de alguns atributos. Para este exemplo em questão, as saídas geadas pelo LIME foram as que tiveram menor concordância com os demais, como pode ser observado na Figura 40.

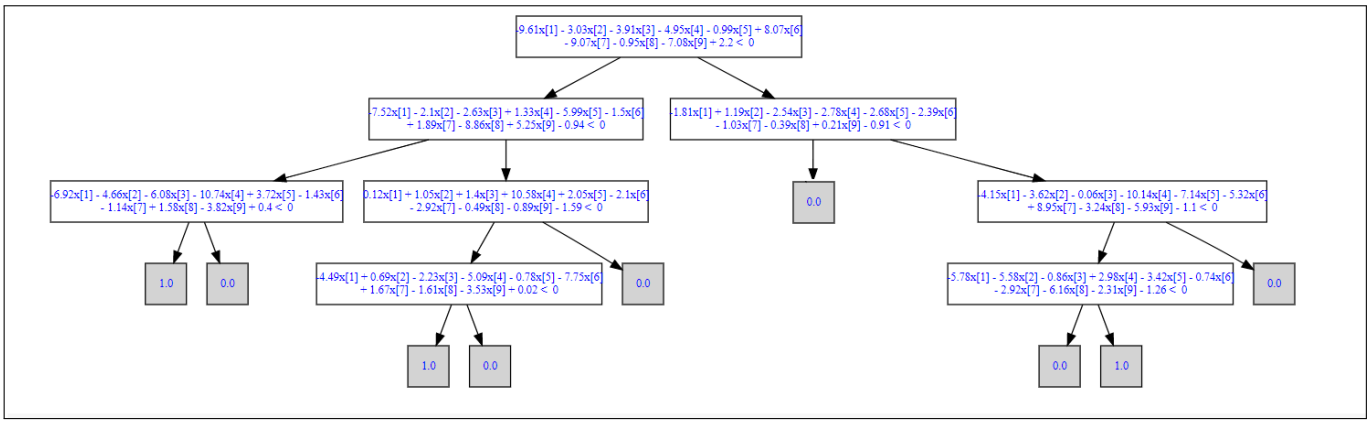
Figura 40 – LIME - Explicação base breast

Fonte: Os autores

Isso pode ser explicado observando a PDT final gerada pelo PDTX, apresentada na Figura 42. Como pode-se notar, observando os nós folhas dessa árvore, a sub-região de interesse possui diversos caminhos que podem retornar a *Classe 0*, e de forma semelhante, a *Classe 1*, o que impossibilitou que a árvore fosse simplificada para uma equivalente de tamanho menor. É um indício que a sub-região trabalhada possui muitas variações que

podem levar a diferentes classificações, a depender de como os recursos são minimamente modificados. Uma vez que o LIME utiliza um ajuste completamente linear, uma região com muitas variações poderá ficar sub-ajustada.

Figura 41 – PDTX - Árvore final gerada - Caso de uso com base Breast



Fonte: Os autores

Essa árvore pode ser decomposta em várias sub-regras, quando cada um dos atributos são explorados individualmente, enquanto os demais são mantidos fixos. A Figura 44 apresenta o conjunto de todas as sub-regras geradas considerando x e os hiperplanos da árvore gerada, incluindo regras redundantes. O intuito é a avaliação individual para cada atributo, cujo filtro está disponível na aplicação gerada. Observe que se for avaliado o atributo x_1, considerando valores reais factíveis, somente a regra $x_1 > -5{,}037$ é suficiente para determinar a classe a ser predita, se os demais atributos forem mantidos fixos.

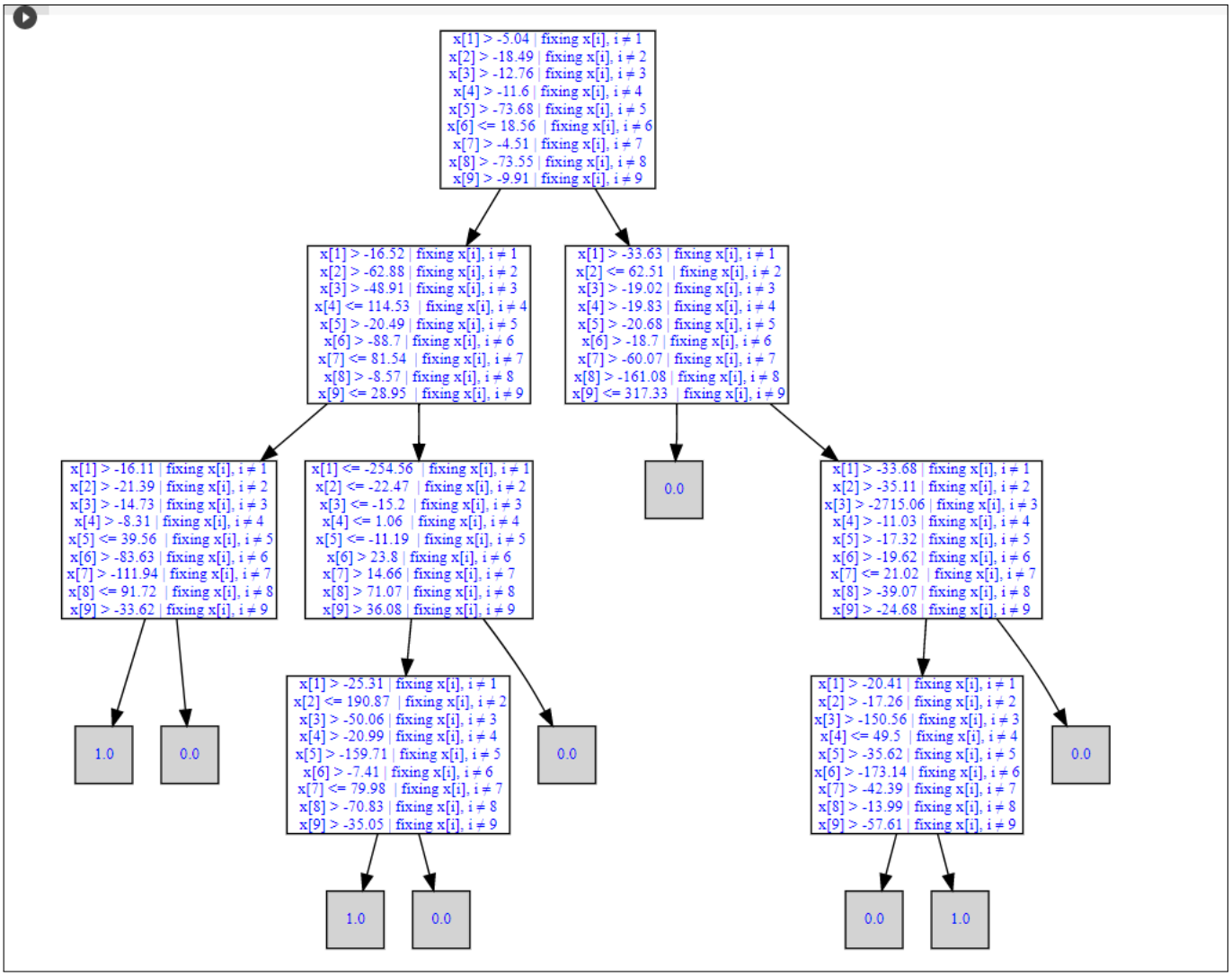
Uma vez que o atributo x_1 é tido como o mais importante, pode-se visualizar as demais precisões possíveis fazendo pequenas mudanças somente nesse atributo, conforme mostrado na Figura 43 O mesmo processo pode ser feito para cada um dos atributos.

A Figura 44 apresenta o conjunto de regras gerados para a instância estudada. Nesse caso, foi obtido o caminho de predição entre o nó raiz e a folha, e foi feito a eliminação das redundâncias.

5.6.3 Área financeira - Previsão de falência

Outra área de muita relevância em se entender as previsões de métodos de decisão automática, é a área financeira. De acordo com Aly et al. (2022), um dos tópicos mais fundamentais na tomada de decisões financeiras e de investimento é a previsão antecipada de eventos de inadimplência de crédito, que podem levar à falência e, portanto, graves consequências tanto na micro quanto na macroeconomia, e por causa disso, justifica-se a utilização de técnicas eficientes de previsão de falência. Tendo isso em vista, foi selecionada a base de dados “analcatdata_bankruptcy” do PMLB, onde os atributos também foram

Figura 42 – PDTX - Regras individuais por hiperplano - Caso de uso com base Breast



Fonte: Os autores

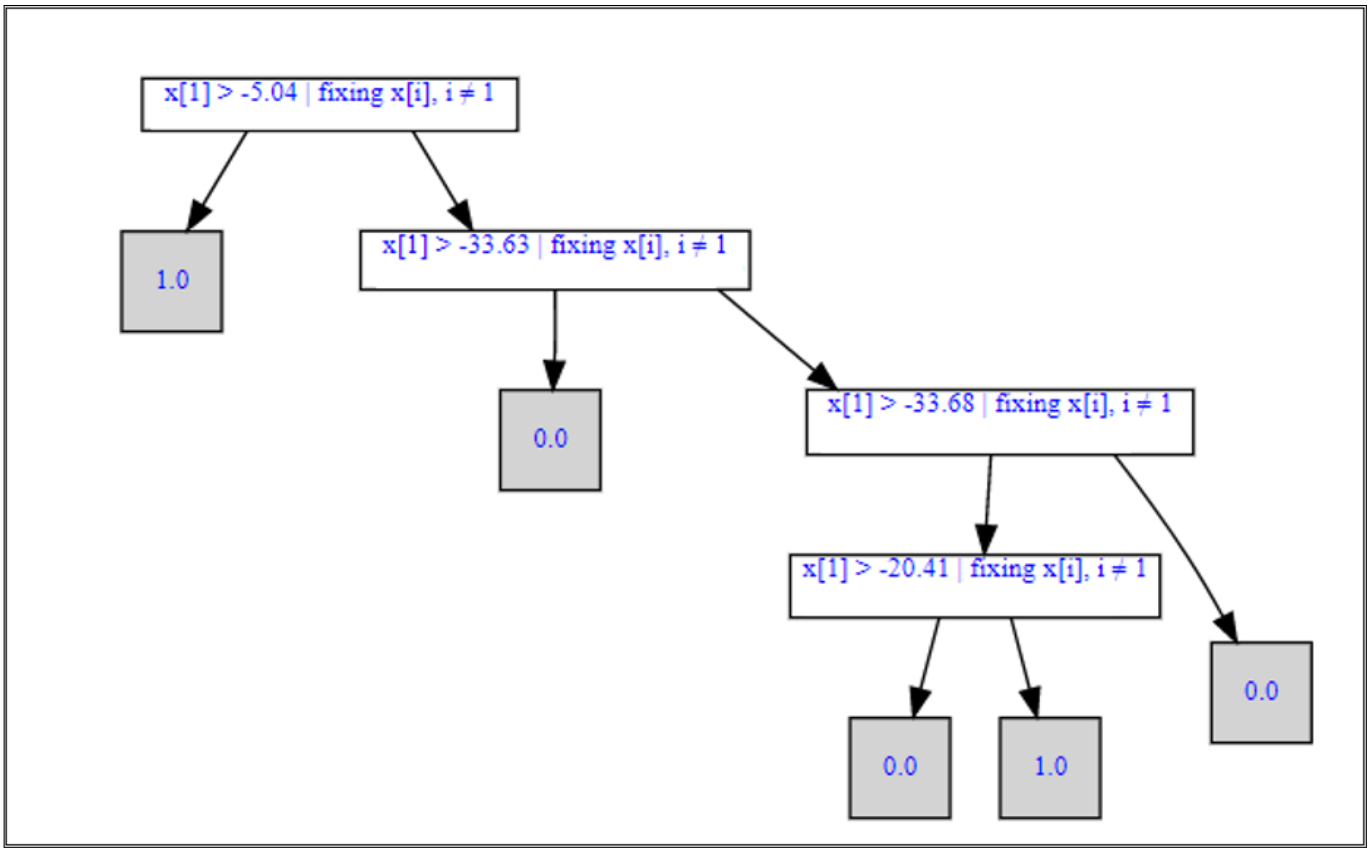
identificados por x_i, que corresponde a empresa, WC/TA, RE/TA, EBIT/TA, S/TA, BVE/BVL, respectivamente. Os resultados obtidos pelo PDTX para a aplicação do MLP sobre um conjunto com 120 amostras foram comparados com os mesmos explicadores das subseções anteriores. As Figuras 45, 46 e 47 apresentam os resultados obtidos pelo PDTX. Foi sorteada a amostra x aleatoriamente.

Conforme pode-se observar, a predição de x é gerada a partir do ativação das seguintes regras:

$$1.56x[1] + 2.44x[2] - 5.07x[3] - 6.21x[4] - 0.18x[5] - 5.21x[6] + 6.68 <= 0$$

AND

Figura 43 – PDTX - Regras individuais para o atributo x_1



Fonte: Os autores

Figura 44 – PDTX - Lista de regras- Caso de uso com base Breast

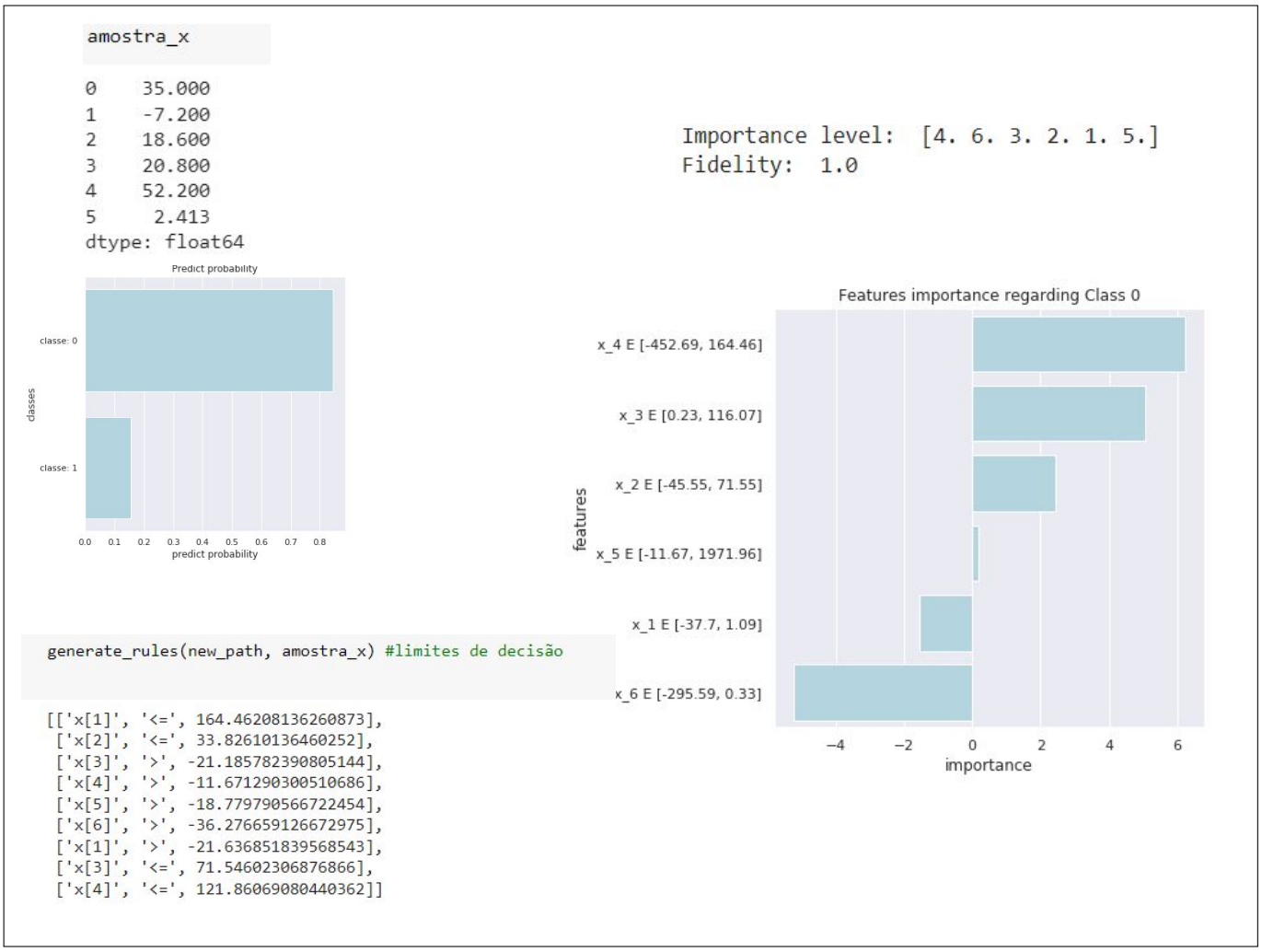
```
generate_rules(new_path, amostra_x) #limites de decisão

[['x[1]', '>', -5.036783951750975],
 ['x[2]', '>', -18.48976666634103],
 ['x[3]', '>', -12.762209994307495],
 ['x[4]', '>', -8.311484195997743],
 ['x[5]', '>', -20.491428996967148],
 ['x[6]', '<=', 18.563591933251022],
 ['x[7]', '>', -4.51437932417216],
 ['x[8]', '>', -8.574188982511927],
 ['x[9]', '>', -9.911745025511589],
 ['x[4]', '<=', 114.52590292826491],
 ['x[6]', '>', -88.69943694135212],
 ['x[7]', '<=', 81.5391253911486],
 ['x[9]', '<=', 28.94809617043434],
 ['x[5]', '<=', 39.5618756179166],
 ['x[6]', '>', -83.63075101596903],
 ['x[8]', '<=', 91.71602807037006]]
```

Fonte: Os autores

$$8.02x[1] - 11.07x[2] - 8.58x[3] - 4.49x[4] + 6.4x[5] + 5.65x[6] - 0.82 > 0$$

que gera a saída Classe 0 que indica que a instituição irá declarar falência – correspondente ao terceiro nó, da esquerda para a direita. Ao determinar que o PDTX obtenha as

Figura 45 – PDTX – Importância dos atributos, limites de decisão e regras de predição - previsão de falência

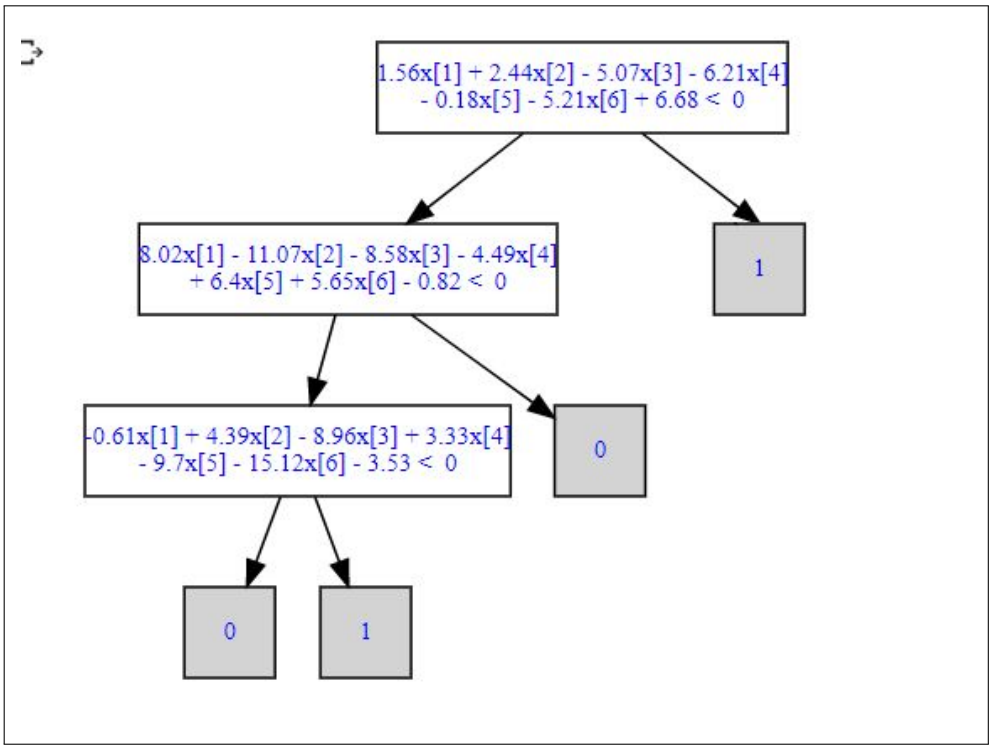


Fonte: Os autores

explicações para essa predição, pode-se que observar que há um conjunto de características que em conjunto contribuem para essa predição, como pode-se inferir pela Figura 45, por causa dos limites de decisão associados a cada x_i. Já na Figura 47 pode-se ver ainda, outras condições que também podem levar a essa mesma predição. Para essa predição, o PDTX teve fidelidade de 100%, o que indica que ele conseguiu reproduzir o comportamento de todas as amostras na vizinhança de x.

Há concordância entre todos os explicadores que as variáveis mais importante são x_4, x_6 e x_3, considerando o retornado pelo PDTX, SHAP e DT, e x_4 e x_3, no que diz respeito ao LIME .

Figura 46 – PDTX – Árvore final gerada - previsão de falência



Fonte: Os autores

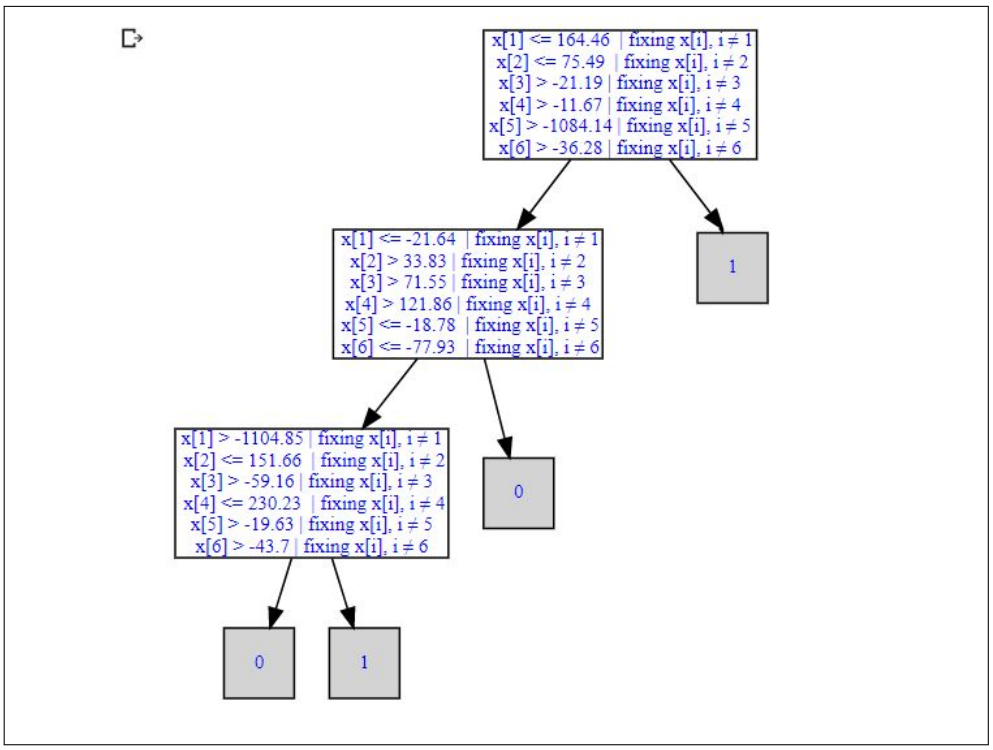
5.6.4 Caso de uso problema de regressão: Predição de Diabetes

Há problemas de regressão que também são de interesse que tenham as previsões explicadas. Neste estudo de caso foi utilizado a base de dados Diabetes disponível na biblioteca Sklearn (Buitinck et al., 2013), que contém 10 preditores, e uma saída que é a taxa de glicose no sangue, indicadora dessa doença. A base de dados foi dividida na prorporção de 80% dos dados para treino e 20% para teste, onde a primeira foi utilizada para treinar um MLP de Regressão com 100 neurônios na camada oculta. A partir da base de teste foi sorteado um valor de x aleatoriamente, e foram gerados 500 pontos na vizinhança de x. Foram usados o PDTX, uma Árvore de Decisão de Regressão, uma Regressão Linear, o SHAP e o LIME para gerarem informações a respeito da predição de x. A Figura 50 mostra a árvore gerada pela PDTX neste processo, com 1500 avaliações da função objetivo, onde obteve-se o $MSE = 0.1081$. A ordem de importância dos atributos, bem como o MSE alcançado pelo PDTX são mostrados na Figura 51.

Os limites de decisão associados a x bem como a ordenação da importância dos atributos são mostrados na Figura 52

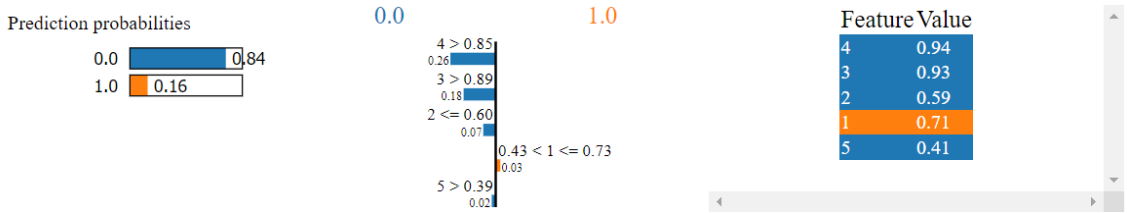
De acordo com a ordem de importância atribuída pela RL contida no nó folha do caminho da predição, a ordem de importância segue a seguinte hierarquia:

Figura 47 – PDTX – Árvore exibindo todas as regras de predição possíveis – Previsão de falência



Fonte: Os autores

Figura 48 – LIME - Explicação - previsão de falência



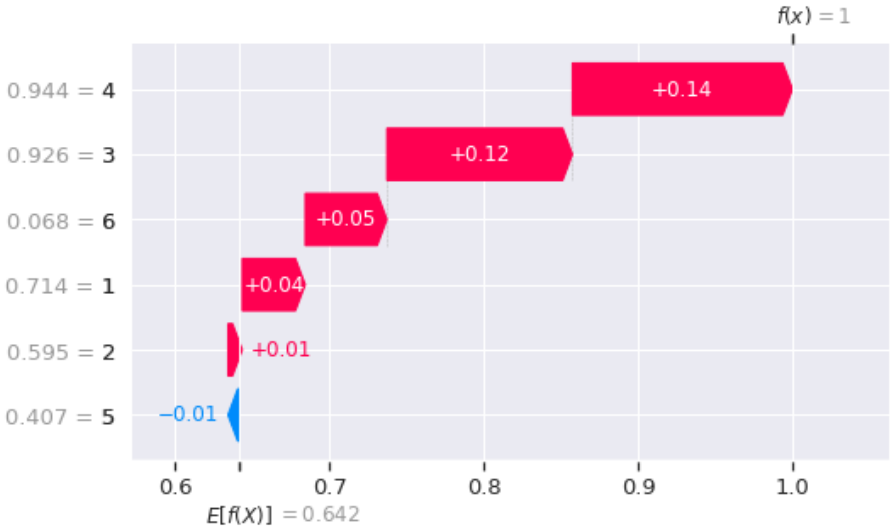
Fonte: Os autores

$$x_2 > x_8 > x_3 > x_7 > x_9 > x_0 > x_4 > x_5 > x_6 > x_1$$

Nota-se que a ordem de importância gerada pelo PDTX encontra-se em perfeita consonância com a ordenação dos 6 primeiro itens da regressão linear, que obteve o $MSE = 1.1334$, conforme apresentado na Figura 53. A ordenação atribuída a este método foi gerada via ordenação pelo valor do coeficiente em módulo.

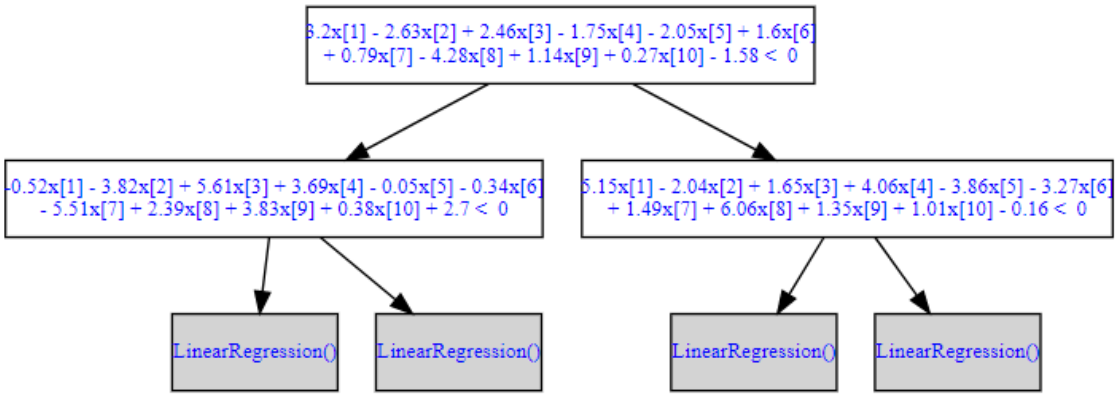
E também uma forte concordância com os 6 primeiros itens retornados como mais importantes pela Árvore de Decisão de Regressão, divergindo em relação a uma troca entre a ordenação dos atributos 3 e 8. Porém, ressalta-se que a qualidade de aproximação do PDTX superaram ambos os métodos em questão. Figura 54. O MSE dessa correspondeu

Figura 49 – SHAP - Explicação - previsão de falência



Fonte: Os autores

Figura 50 – PDTX - Árvore gerada - Diabetes



Fonte: Os autores

Figura 51 – PDTX - MSE e predição - Base Diabetes

```
[2 8 3 7 9 0 1 4 6 5]
0.10811707481866108
```

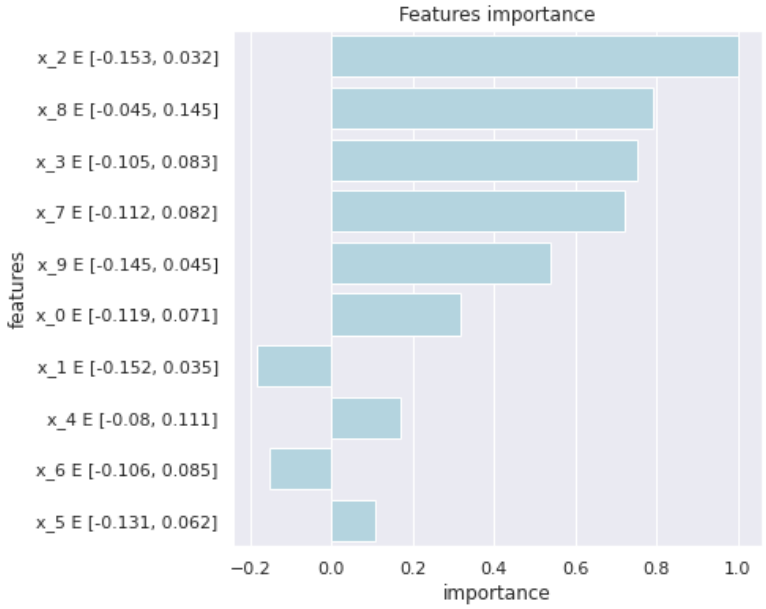
Fonte: Os autores

a 176.48.

Por outro lado, para o SHAP os preditores com maiores importância são os x_8, x_2 e x_9, nessa ordem, conforme Figura 55.

O LIME, por sua vez, considerou como mais importante os recursos x_1, x_9 e x_0, conforme pode ser observado na Figura 56. Assim, nota-se que para essa predição não

Figura 52 – PDTX - Explicação - base Diabetes




```
print('Predict value ',predict_(population, amostra_x))
```

```
Predict value  [128.88875932]
```

```
amostra_x
```

	0	1	2	3	4	5	6	7	8	9
73	0.216667	1.0	0.305785	0.399061	0.27451	0.292829	0.155844	0.352609	0.447363	0.378788

Fonte: Os autores

Figura 53 – Regressão linear - Ordem de importância - Base Diabetes

```
1.1334097188709953
array([2, 8, 3, 7, 9, 0, 4, 5, 6, 1])
```

Fonte: Os autores

houve uma concordância global entre todos os explicadores estudados.

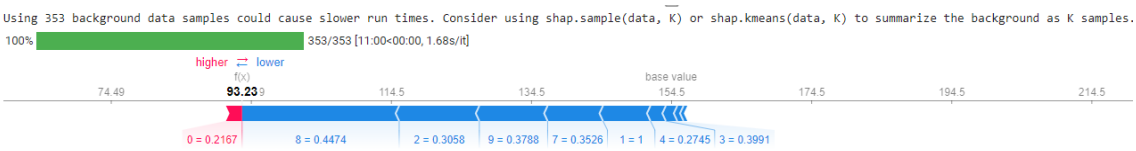
Pequenas diferenças entre os métodos são esperadas, em virtude dos diferentes mecanismos de identificação de importância entre os atributos. Uma vez que o PDTX primeiro isola uma sub-região dentro da vizinhança da instância trabalhada, e por fim, executa uma Regressão Linear sobre eles, é esperado que a qualidade dessa aproximação seja melhor do que a execução isolada da RL, ou de uma DT, que faz uma operação de média sobre as amostras filtrados para as folhas, além de realizar divisões paralelas ou eixo de atributos. O mesmo raciocínio aplica-se ao LIME, que também faz uso de modelos

Figura 54 – DT - Explicação - Base Diabetes

```
176.4823799550856
array([2, 3, 8, 7, 9, 0, 4, 6, 5, 1])
```

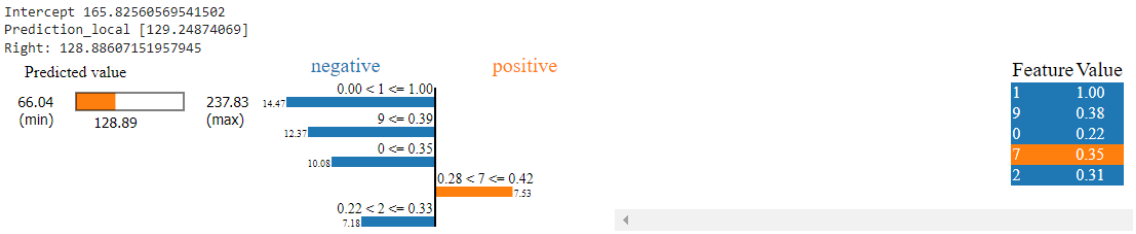
Fonte: Os autores

Figura 55 – SHAP - Explicação - Base Diabetes



Fonte: Os autores

Figura 56 – LIME - Explicação - Base Diabetes



Fonte: Os autores

globalmente lineares. A concordância entre diferentes métodos reforça a importância dos atributos que se destacam.

Quando as explicações fornecidas entre explicadores são significativamente diferentes, a fidelidade local pode ser a métrica escolhida para o usuário definir qual método de explicação adotar, no contexto em que os métodos que fazem a aproximação local para gerar as explicações são utilizados. Isso porque essa métrica consegue quantificar a qualidade da aproximação realizada, e a aproximação de boa qualidade é essencial para confiabilidade da explicação gerada.

Capítulo 6

Conclusão

A demanda por interpretabilidade em IA permanece um tema de alta relevância dentro da atualidade. A proposta de explicadores, especialmente os *post-hoc* e agnósticos quanto ao modelo ajudam a cumprir este requisito, tanto para métodos opacos já existentes, como outros que venham ser propostos. Tais soluções ajudam a aumentar a transparência quanto ao processo de tomada de decisão realizado, identificar vieses nos dados, melhorar a manutenibilidade do sistema, e a cumprir as legislações propostas para proteger os cidadãos e garantir a transparência da tomada de decisões em contextos que os impactem.

Nesse contexto, foi proposta a avaliação quanto a aplicação de árvores oblíquas para gerações de explicações locais para modelos ML opacos, sendo que essa investigação resultou em um explicador chamado PDTX. O PDTX em sua versão atual é capaz de fornecer explicações mais robustas e mais fidedignas quando comparadas a alguns métodos da literatura, como LIME e DT. Por utilizar como base PDTs para geração das explicações, este método consegue fazer uma boa aproximação, mesmo em contextos que explicadores lineares falham – quando a vizinhança local mapeia uma função complexa. Além disso, outras formas de visualizações são propostas para melhor compreensão dos usuários quanto à importância dos atributos, os limites de decisões, o caminho percorrido para chegar à resposta, e as *features* que se mostraram importantes na determinação das divisões oblíquas.

Os métodos de explicação que se baseiam em modelos lineares podem apresentar baixa fidelidade ao aproximar regiões com muitas não-linearidades por serem simples, semelhante à situação de *underfitting* (Somogyi, 2021). Isso significa que a região selecionada ao redor da amostra deve ser tão pequena que consiga ser aproximada por meio de um modelo com essas características. Caso contrário, as explicações geradas não corresponderão à realidade . Alguns trabalhos propõem diferentes abordagens que definam tanto uma sub-região como uma vizinhança específica que permita a utilização deste tipo de método (Botari et al., 2020). Uma vez que uma PDT é um método linear por partes, o próprio método permite encontrar essas sub-regiões sem dificuldades. Ainda sim, ela possui maior

capacidade de aproximação local quando se comparado a modelos mais simples. Dessa forma, o estabelecimento de uma vizinhança esta de tamanho representativo possibilita a geração de explicações fidedignas por meio deste método. Assim, enquanto a aplicação de métodos lineares pode gerar baixa fidelidade e instabilidade, ou demande cálculos complexos para estabelecimento da vizinhança e/ou sub-região, a PDT consegue gerar resultados com menor complexidade. Além disso, o PDTX fornece mais de um tipo de explicação para ML de classificação e regressão com dados estruturados. São geradas explicações referente à hierarquia de importância das características, apresenta-se os limites de decisões de cada recurso, e as regras utilizadas pela PDT para chegar à saída, etc. Dessa forma, a aplicação do PDTX para esses métodos gera resultados mais completos e fornece mais explicações para o tomador de decisão. A avaliação quantitativa do método proposto mostrou que o PDTX gera resultados promissores quanto à fidelidade e estabilidade quando se comparado ao LIME e a DT, sendo portanto uma boa alternativa como método de explicação local.

Neste trabalho também foi proposto sugestões de interpretação de uma PDT que, embora seja um método transparente e interpretável, a forma de interpretá-la foi muito pouco explorada na literatura. Diversos dos trabalhos que propõe métodos de evolução de uma PDT não fazem menção de como utilizá-la para entender as saídas geradas (Wickramarachchi et al., 2016; Murthy et al., 1994). Foram encontrados poucos trabalhos que fazem menção de como extrair informações por meio dela (Carreira-Perpinán e Hada, 2021).

Tendo em vista que um dos aspectos importantes a ser considerado ao implementar um explicador local como este é a estabilidade, e que a aleatoriedade do processo de geração de pontos sintéticos pode introduzir instabilidade ao sistema, neste trabalho também foi feito o estudo de diferentes técnicas de amostragem para geração da vizinhança local em torno da amostra a ser explicada, avaliando especialmente o impacto na estabilidade do método, sendo que a abordagem que gera melhor estabilidade e as condições para tal são aplicadas na versão final do PDTX como padrão. Contudo, o *framework* resultante deste trabalho permite ao usuário o uso de diferentes definições de vizinhança. Os resultados mostraram que o processo de geração amostral baseado na sequência de Hammersley gera resultados mais estáveis quanto à hierarquia de atributos para diferentes execuções do PDTX mantendo-se o mesmo procedimento experimental, quando comparados àqueles gerados pela amostragem aleatória a partir de uma distribuição gaussiana, ou a partir da amostragem por hipercubo latino.

Outrossim, cinco métodos de seleção de características existentes na literatura foram avaliados no tocante a fidelidade do PDTX. Ou seja, foi verificado se algum deles produzia melhor desempenho concernente a essa métrica, considerando as mesmas especificidades de entrada. Os métodos de seleção considerados foram: *Chi-squared*, o RFE, a seleção utilizando regularização (LR), a seleção por Extremely Randomized

Trees Classifier e o Boruta. Os resultados gerados não apresentam diferença estatística significativa entre eles, ao nível de confiança de 95%. Porém, apresentou resultados para fidelidade do PDTX melhores que os concorrentes avaliados, utilizando cerca de 1/10 da quantidade de gerações do experimento de classificação de comparação entre os explicadores. Embora esses resultados – com a seleção de atributos– não são diretamente comparáveis – sem essa seleção, visto que os procedimentos experimentais adotados foram distintos, há evidências de que a seleção de característica aplicada no início do processo evolucionário se traduz em ganho computacional para o PDTX, visto que nem todos os recursos são localmente relevantes.

Por fim, foram apresentados os resultados para aplicação do PDTX, em conjunto com o LIME, SHAP e outros métodos que podem ser usados para o propósito de explicação local, em especial para áreas consideradas críticas. Foi demonstrado as diferentes maneiras pelas quais o PDTX gera as explicações, bem como a coerência das explicações quando são comparadas a de outros métodos. Tantos nos resultados experimentais, como casos de usos apresentados, foram utilizadas bases de dados reais.

6.1 Limitações

Embora o PDTX consiga superar explicadores que se baseiam em modelos lineares em problemas que tenham muitas não-linearidades, inclusive na região ao redor da instância de interesse, a performance da PDTX é, atualmente, dependente dos hiper-parâmetros pré-definidos. Nesse sentido, a altura da árvore que seja suficiente para produzir explicações que ao mesmo tempo que tenham alta fidelidade sejam também estáveis deve ser definida conforme a especificidade do problema. Dessa forma, acredita-se que uma boa árvore deve ter a menor complexidade possível que consiga captar todas as particularidades nas redondezas de x. Os hiper-parâmetros utilizados pelo PDTX para avaliação dos explicadores, foram obtidos empiricamente, sendo necessário mais estudos quanto a otimalidade desses.

Além disso, a implementação utilizada neste trabalho utiliza uma estratégia heurística para obtenção otimizada de uma PDT. Apesar de o PDTX obter aproximações muito melhores que o de explicadores propostos na literatura, existe um esforço computacional associado à utilização de métodos não determinísticos, especialmente neste contexto que várias soluções candidatas são avaliadas. Para reduzir este problema, foi proposto neste trabalho considerar a utilização do processo de seleção de características no início do processo evolutivo dentro da vizinhança da instância de interesse, em especial para bases grandes. Com isso, somente características consideradas importantes são mantidas ao longo do processo evolucionário. Todavia, acredita-se que abordagens determinísticas, que consigam gerar resultados tão bons ou melhores que da implementação proposta possam reduzir ainda mais o custo computacional associado à aplicação da versão atual da PDTX.

De forma semelhante, a implementação de paralelismo mantendo-se a opção heurística pode minimizar este problema, bem como a reimplementação da técnica utilizando linguagens compiladas e popularmente mais rápidas, como o C++, ao invés do Python.

6.2 Trabalhos Futuros

Tendo em vista que o PDTX consegue fornecer um conjunto de informações com alta fidelidade e estabilidade em comparação com as metodologias estudadas, e que apresenta informações relevantes para a tomada de decisão por especialistas, como trabalho futuro dessa abordagem, cita-se a elaboração de uma interface de explicação do PDTX, que tenha integração com o usuário e seja visualmente agradável, de forma a ajudar o usuário na tomada de decisão, ao fornecer um conjunto de informações compreensíveis para tal.

Considera-se relevante também a avaliação da expansão desta metodologia para atender dados não estruturados, como imagens, textos, sons, vídeos, etc. A demanda por utilização de métodos ML que trabalham com esses tipos de informações permanece alta e muitos dos explicadores existentes não atendem essa necessidade. Assim, a investigação da aplicação de árvores oblíquas para atender tais categorias deve ser considerada.

Finalmente, o estudo de outros métodos para evolução da PDT também se faz importante, tendo em vista que, embora o jSO seja uma técnica muito poderosa e com alto poder de otimização, é importante investigar alternativas que mantenham o poder de otimização dos pesos da PDT, ao mesmo tempo que produzam resultados mais imediatos, principalmente opções determinísticas. A implementação de paralelismo em relação à técnica atual também pode ser explorada.

Referências

A. Adadi e M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

J. Adeyemo e F. Otieno. Multi-objective differential evolution algorithm for solving engineering problems. *Journal of Applied Sciences*, 9(20):3652–3661, 2009.

T. Ahmad e J. Gesley. Regulation of artificial intelligence: International and regional approaches. Disponível em: https://www.loc.gov/law/help/artificial-intelligence/international.php, 2019. Acesso em: 4 de agosto de 2020.

G. S. I. Aldeia e F. O. de França. Measuring feature importance of symbolic regression models using partial effects. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 750–758, 2021.

G. S. I. Aldeia e F. O. de França. Interpretability in symbolic regression: a benchmark of explanatory methods using the feynman data set. *Genetic Programming and Evolvable Machines*, pages 1–41, 2022.

M. Ali. Differential evolution with preferential crossover. *European Journal of Operational Research*, 181(3):1137–1147, 2007.

M. Ali e M. Pant. Improving the performance of differential evolution algorithm using cauchy mutation. *Soft Computing*, 15(5):991–1007, 2011.

M. Ali, M. Pant, e A. Abraham. Unconventional initialization methods for differential evolution. *Applied Mathematics and Computation*, 219(9):4474–4494, 2013.

G. Alves. Regressão linear, 2019. Disponível em: https://medium.com/@gisely.alves/regress%C3%A3o-linear-7d9d3b2ec815. Acesso em: 12 junho 2022.

S. Aly, M. Alfonse, e A.-B. M. Salem. Bankruptcy prediction using artificial intelligence techniques: A survey. In *Digital Transformation Technology*, pages 335–360. Springer, 2022.

J. Amann, A. Blasimme, E. Vayena, D. Frey, e V. I. Madai. Explainability for artificial intelligence in healthcare: a multidisciplinary perspective. *BMC Medical Informatics and Decision Making*, 20(1):1–9, 2020.

A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

A. Asuncion e D. Newman. UCI machine learning repository, 2007. Disponível em: https://archive.ics.uci.edu/ml/index.php. Acesso em: 12 de janeiro de 2022.

R. C. Barros, R. Cerri, P. A. Jaskowiak, e A. C. De Carvalho. A bottom-up oblique decision tree induction algorithm. In *2011 11th International Conference on Intelligent Systems Design and Applications*, pages 450–456. IEEE, 2011.

R. C. Barros, P. A. Jaskowiak, R. Cerri, e A. C. de Carvalho. A framework for bottom-up induction of oblique decision trees. *Neurocomputing*, 135:3–12, 2014.

N. Beat. Ml | extra tree classifier for feature selection, 2020. Disponível em: https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/. Acesso em: 15 de maio de 2022.

Z. Beheshti e S. M. H. Shamsuddin. A review of population-based meta-heuristic algorithms. *Int. J. Adv. Soft Comput. Appl*, 5(1):1–35, 2013.

K. P. Bennett e O. L. Mangasarian. Multicategory discrimination via linear programming. *Optimization methods and Software*, 3(1-3):27–39, 1994.

K. P. Bennett, D. Wu, e L. Auslender. On support vector decision trees for database marketing. *Department of Mathematical Sciences Math Report*, (98-100), 1998.

P. Biecek. Dalex: explainers for complex predictive models in R. *The Journal of Machine Learning Research*, 19(1):3245–3249, 2018.

M. Biswas, M. S. Kaiser, M. Mahmud, S. Al Mamun, M. Hossain, M. A. Rahman, et al. An xai based autism detection: The context behind the detection. In *International Conference on Brain Informatics*, pages 448–459. Springer, 2021.

M. C. Bot e W. B. Langdon. Application of genetic programming to induction of linear classification trees. In *European Conference on Genetic Programming*, pages 247–258. Springer, 2000.

T. Botari, F. Hvilshøj, R. Izbicki, e A. C. de Carvalho. MeLIME: meaningful local explanation for machine learning models. *arXiv preprint arXiv:2009.05818*, 2020.

L. Breiman, J. Friedman, R. Olshen, e C. Stone. Classification and regression trees–crc press. *Boca Raton, Florida*, 1984.

J. Brest, S. Greiner, B. Boskovic, M. Mernik, e V. Zumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE transactions on evolutionary computation*, 10(6):646–657, 2006.

J. Brest, M. S. Maučec, e B. Bošković. iL-SHADE: Improved l-shade algorithm for single objective real-parameter optimization. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 1188–1195. IEEE, 2016.

J. Brest, M. S. Maučec, e B. Bošković. Single objective real-parameter optimization: Algorithm jso. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1311–1318, 2017. doi: 10.1109/CEC.2017.7969456.

L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, e G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

P. Bujok e J. Tvrdík. New variants of adaptive differential evolution algorithm with competing strategies. *Acta Electrotech. Inform*, 15(2):49–56, 2015.

J. Cai, J. Luo, S. Wang, e S. Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70–79, 2018.

M. A. Carreira-Perpinán e S. S. Hada. Counterfactual explanations for oblique decision trees: Exact, efficient algorithms. In *Proc. of the 35th AAAI Conference on Artificial Intelligence (AAAI'21)*, 2021.

M.-L. Cauwet, C. Couprie, J. Dehos, P. Luc, J. Rapin, M. Riviere, F. Teytaud, O. Teytaud, e N. Usunier. Fully parallel hyperparameter search: Reshaped space-filling. In *International Conference on Machine Learning*, pages 1338–1348. PMLR, 2020.

X.-w. Chen e J. C. Jeong. Enhanced recursive feature elimination. In *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, pages 429–435. IEEE, 2007.

L. Chiticariu, Y. Li, e F. Reiss. Transparent machine learning for information extraction: state-of-the-art and the future. *EMNLP (tutorial)*, 2015.

V. A. Cicirello. Kendall tau sequence distance: Extending kendall tau from ranks to sequences. *arXiv preprint arXiv:1905.02752*, 2019.

M.-A. Clinciu e H. Hastie. A survey of explainable ai terminology. In *Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019)*, pages 8–13, 2019.

R. Confalonieri, L. Coba, B. Wagner, e T. R. Besold. A historical perspective of explainable artificial intelligence. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(1):e1391, 2021.

M. W. Craven. Extracting comprehensible models from trained neural networks. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1996.

A. Cutler, D. R. Cutler, e J. R. Stevens. Random forests. In *Ensemble machine learning*, pages 157–175. Springer, 2012.

C. E. da Silva Santos, R. C. Sampaio, L. dos Santos Coelho, G. A. Bestard, e C. H. Llanos. Multi-objective adaptive differential evolution for svm/svr hyperparameters selection. *Pattern Recognition*, 110:107649, 2021.

V. V. de Melo e A. C. B. Delbem. Investigating smart sampling as a population initialization method for differential evolution in continuous problems. *Information Sciences*, 193: 36–53, 2012.

W. Deng, S. Shang, X. Cai, H. Zhao, Y. Song, e J. Xu. An improved differential evolution algorithm and its application in optimization problem. *Soft Computing*, 25(7):5277–5298, 2021.

O. Deperlioglu, U. Kose, D. Gupta, A. Khanna, F. Giampaolo, e G. Fortino. Explainable framework for glaucoma diagnosis by image processing and convolutional neural network synergy: Analysis with doctor evaluation. *Future Generation Computer Systems*, 129: 152–169, 2022.

Z. Dexuan e G. Liqun. An efficient improved differential evolution algorithm. In *Proceedings of the 31st Chinese Control Conference*, pages 2385–2390. IEEE, 2012.

D. Doran, S. Schulz, e T. R. Besold. What does explainable AI really mean? A new conceptualization of perspectives. *CoRR*, abs/1710.00794, 2017. Disponível em: http://arxiv.org/abs/1710.00794. Acesso em: 04 de março de 2022.

F. K. Došilović, M. Brčić, e N. Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 0210–0215. IEEE, 2018.

T. Eltaeib e A. Mahmood. Differential evolution: A survey and analysis. *Applied Sciences*, 8(10):1945, 2018.

H.-Y. Fan, J. Liu, e J. Lampinen. Some improvement to the mutation donor of differential evolution. *Engineering Computations*, 2010.

Q. Fan e Y. Zhang. Self-adaptive differential evolution algorithm with crossover strategies adaptation and its application in parameter estimation. *Chemometrics and Intelligent Laboratory Systems*, 151:164–171, 2016.

L. Farber. 5 distribuição normal de probabilidade, 2015.

L. A. Ferreira, F. G. Guimarães, e R. Silva. Applying genetic programming to improve interpretability in machine learning models. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

I. Fister, A. Tepeh, e I. Fister Jr. Epistatic arithmetic crossover based on cartesian graph product in ensemble differential evolution. *Applied Mathematics and Computation*, 283: 181–194, 2016.

E. J. d. R. Freitas, S. S. Santos, e T. M. Rezende. *Redes Neurais Artificiais: Uma visão histórica*. 2022.

R. Gämperle, S. D. Müller, e P. Koumoutsakos. A parameter study for differential evolution. *Advances in intelligent systems, fuzzy systems, evolutionary computation*, 10(10):293–298, 2002.

M. W. Gardner e S. Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15): 2627–2636, 1998.

M. Georgioudakis e V. Plevris. On the performance of differential evolution variants in constrained structural optimization. *Procedia Manufacturing*, 44:371–378, 2020.

M. Gleicher. A framework for considering comprehensibility in modeling. *Big data*, 4(2): 75–88, 2016.

A. Goldstein, A. Kapelner, J. Bleich, e E. Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

I. J. Goodfellow, J. Shlens, e C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

B. Goodman e S. Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3):50–57, 2017.

A. Gosiewska e P. Biecek. Ibreakdown: Uncertainty of model explanations for non-additive predictive models. *arXiv preprint arXiv:1903.11420*, 2019.

B. M. Greenwell. pdp: An r package for constructing partial dependence plots. *R J.*, 9(1): 421, 2017.

R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, e D. Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5): 1–42, 2018.

D. Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2:2, 2017.

J. M. Hammersley. Monte carlo methods for solving multivariable problems. *Annals of the New York Academy of Sciences*, 86(3):844–874, 1960.

A. Holzinger, G. Langs, H. Denk, K. Zatloukal, e H. Müller. Causability and explainability of artificial intelligence in medicine. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(4):e1312, 2019.

S. C. Hora e J. C. Helton. A distribution-free test for the relationship between model input and output when using latin hypercube sampling. *Reliability Engineering & System Safety*, 79(3):333–339, 2003.

M. Ibrahim, M. Louie, C. Modarres, e J. Paisley. Global explanations of neural networks: Mapping the landscape of predictions. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 279–287, 2019.

S. M. Islam. Swagatam das, saurav ghosh, subhrajit roy, and ponnuthurai nagaratnam suganthan." an adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization.". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42:482–500, 2012.

Y. Jia, J. Bailey, K. Ramamohanarao, C. Leckie, e M. E. Houle. Improving the quality of explanations with local embedding perturbations. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 875–884, 2019.

E. Jiang. Uniformlime: A uniformly perturbed local interpretable model-agnostic explanations approach for aerodynamics. In *Journal of Physics: Conference Series*, volume 2171, page 012025. IOP Publishing, 2022.

G. Joshi, R. Walambe, e K. Kotecha. A review on explainability in multimodal deep neural nets. *IEEE Access*, 2021.

J. Júnior. Projeto cria marco legal para uso de inteligência artificial no brasil, 2020. https://www.camara.leg.br/noticias/641927-projeto-cria-marco-legal-para-uso-de-inteligencia-artificial-no-brasil/. Acesso em: 4 de agosto de 2020.

J. R. Kalagnanam e U. M. Diwekar. An efficient sampling technique for off-line quality control. *Technometrics*, 39(3):308–319, 1997.

D. Karger, C. Stein, e J. Wein. Algorithms and theory of computation handbook. *chap. Scheduling Algorithms*, pages 20–20, 2010.

J. Kasza e R. Wolfe. Interpretation of commonly used statistical regression models. *Respirology*, 19(1):14–21, 2014.

R. Kavya, J. Christopher, S. Panda, e Y. B. Lazarus. Machine learning and xai approaches for allergy diagnosis. *Biomedical Signal Processing and Control*, 69:102681, 2021.

B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

J. W. Kim, B. H. Lee, M. J. Shaw, H.-L. Chang, e M. Nelson. Application of decision-tree induction techniques to personalized advertisements on internet storefronts. *International Journal of Electronic Commerce*, 5(3):45–62, 2001.

A. Kotriwala, B. Klöpper, M. Dix, G. Gopalakrishnan, D. Ziobro, e A. Potschka. Xai for operations in the process industry-applications, theses, and research directions. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*, 2021.

A. Krizhevsky, I. Sutskever, e G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

C. Kuo. Explain your model with lime, 2020. Disponível em: https://medium.com/dataman-in-ai/explain-your-model-with-lime-5a1a5867b423. Acesso em 10 de junho de 2022.

M. B. Kursa e W. R. Rudnicki. Feature selection with the boruta package. *Journal of statistical software*, 36:1–13, 2010.

M. B. Kursa, A. Jankowski, e W. R. Rudnicki. Boruta–a system for feature selection. *Fundamenta Informaticae*, 101(4):271–285, 2010.

V. La Gatta, V. Moscato, M. Postiglione, e G. Sperlì. Castle: Cluster-aided space transformation for local explanations. *Expert Systems with Applications*, 179:115045, 2021.

F. J. V. Z. Levy Boccato, Romis Ribeiro de Faissol Attux. Evolução diferencial: Introdução e conceitos básicos, 2014. Disponível em: https://www.dca.fee.unicamp.br/l̃boccato/topico_11_evolucao_diferencial.pdf. Acesso em 19 julho 2021.

H. Li, F. He, Y. Chen, e Y. Pan. Mlfs-ccde: multi-objective large-scale feature selection by cooperative coevolutionary differential evolution. *Memetic Computing*, 13(1):1–18, 2021.

J. Liang, W. Xu, C. Yue, K. Yu, H. Song, O. D. Crisalle, e B. Qu. Multimodal multiobjective optimization with differential evolution. *Swarm and evolutionary computation*, 44:1028–1059, 2019.

J. Liang, K. Qiao, C. Yue, K. Yu, B. Qu, R. Xu, Z. Li, e Y. Hu. A clustering-based differential evolution algorithm for solving multimodal multi-objective optimization problems. *Swarm and Evolutionary Computation*, 60:100788, 2021.

P. Linardatos, V. Papastefanopoulos, e S. Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2020.

J. Liu e J. Lampinen. A fuzzy adaptive differential evolution algorithm. *Soft Computing*, 9(6):448–462, 2005.

W. Liu e I. W. Tsang. Sparse perceptron decision tree for millions of dimensions. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

W.-L. Loh. On latin hypercube sampling. *The annals of statistics*, 24(5):2058–2080, 1996.

R. A. Lopes, A. Freitas, R. P. Silva, e F. G. Guimarães. Differential evolution and perceptron decision trees for classification tasks. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 550–557. Springer, 2012.

A. López-Chau, J. Cervantes, L. López-García, e F. G. Lamont. Fisher's decision tree. *Expert Systems with Applications*, 40(16):6283–6291, 2013.

A. C. Lorena e A. C. De Carvalho. Uma introdução às support vector machines. *Revista de Informática Teórica e Aplicada*, 14(2):43–67, 2007.

S. M. Lundberg e S.-I. Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

G. Marcus. Deep learning: A critical appraisal. *CoRR*, abs/1801.00631, 2018. Disponível em: http://arxiv.org/abs/1801.00631. Acesso em: 26 de janeiro de 2022.

V. Margot e G. Luta. A new method to compare the interpretability of rule-based algorithms. *AI*, 2(4):621–635, 2021.

V. Margot, J.-P. Baudry, F. Guilloux, e O. Wintenberger. Consistent regression using data-dependent coverings. *Electronic Journal of Statistics*, 15(1):1743–1782, 2021.

A. Messalas, Y. Kanellopoulos, e C. Makris. Model-agnostic interpretability with shapley values. In *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–7, 2019. doi: 10.1109/IISA.2019.8900669.

T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

A. W. Mohamed. An improved differential evolution algorithm with triangular mutation for global numerical optimization. *Computers & Industrial Engineering*, 85:359–375, 2015.

C. Molnar. *Interpretable Machine Learning*. 2019. https://christophm.github.io/interpretable-ml-book/.

J. D. Moore e W. R. Swartout. Explanation in expert systemss: A survey. Technical report, University Of Southern California Marina Del Rey Information Sciences Inst, 1988.

A. Morikawa e Y. Matsubara. Safety design concepts for statistical machine learning components toward accordance with functional safety standards. *arXiv preprint arXiv:2008.01263*, 2020.

R. Mukherjee, S. Debchoudhury, R. Kundu, S. Das, e P. N. Suganthan. Adaptive differential evolution with locality based crossover for dynamic optimization. In *2013 IEEE Congress on Evolutionary Computation*, pages 63–70. IEEE, 2013.

S. K. Murthy, S. Kasif, e S. Salzberg. A system for induction of oblique decision trees. *Journal of artificial intelligence research*, 2:1–32, 1994.

N. Nahar, F. Ara, M. Neloy, A. Istiek, A. Biswas, M. S. Hossain, e K. Andersson. Feature selection based machine learning to improve prediction of parkinson disease. In *International Conference on Brain Informatics*, pages 496–508. Springer, 2021.

M. Nassar, H. Safa, A. A. Mutawa, A. Helal, e I. Gaba. Chi squared feature selection over apache spark. In *Proceedings of the 23rd International Database Applications & Engineering Symposium*, pages 1–5, 2019.

L. L. Nathans, F. L. Oswald, e K. Nimon. Interpreting multiple linear regression: a guidebook of variable importance. *Practical assessment, research & evaluation*, 17(9): n9, 2012.

R. Noothigattu, D. Bouneffouf, N. Mattei, R. Chandra, P. Madan, K. Varshney, M. Campbell, M. Singh, e F. Rossi. Interpretable multi-objective reinforcement learning through policy orchestration. *arXiv preprint arXiv:1809.08343*, 2018.

R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, e J. H. Moore. Pmlb: a large benchmark suite for machine learning evaluation and comparison. *BioData mining*, 10 (1):1–13, 2017.

G. C. Onwubolu e D. Davendra. *Differential evolution: A handbook for global permutation-based combinatorial optimization*, volume 175. Springer Science & Business Media, 2009.

M. Pant, M. Ali, e V. P. Singh. Differential evolution with parent centric crossover. In *2008 Second UKSIM European Symposium on Computer Modeling and Simulation*, pages 141–146. IEEE, 2008.

M. Pant, M. Ali, e V. P. Singh. Differential evolution using quadratic interpolation for initializing the population. In *2009 IEEE International Advance Computing Conference*, pages 375–380. IEEE, 2009.

M. Pant, H. Zaheer, L. Garcia-Hernandez, A. Abraham, et al. Differential evolution: A review of more than two decades of research. *Engineering Applications of Artificial Intelligence*, 90:103479, 2020.

D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, e F. Turini. Meaningful explanations of black box ai decision systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9780–9784, 2019.

E. Pintelas, I. E. Livieris, e P. Pintelas. A grey-box ensemble model exploiting black-box accuracy and white-box intrinsic interpretability. *Algorithms*, 13(1):17, 2020.

R. Pishchalnikov. Application of the differential evolution for simulation of the linear optical response of photosynthetic pigments. *Journal of Computational Physics*, 372: 603–615, 2018.

G. Plumb, D. Molitor, e A. S. Talwalkar. Model agnostic supervised local explanations. *Advances in neural information processing systems*, 31, 2018.

R. S. Prado, R. C. Silva, F. G. Guimarães, e O. M. Neto. Using differential evolution for combinatorial optimization: A general approach. In *2010 IEEE International Conference on Systems, Man and Cybernetics*, pages 11–18. IEEE, 2010.

A. Preece. Asking 'why'in ai: Explainability of intelligent systems–perspectives and challenges. *Intelligent Systems in Accounting, Finance and Management*, 25(2):63–72, 2018.

K. V. Price. Differential evolution. In *Handbook of optimization*, pages 187–214. Springer, 2013.

A. K. Qin e P. N. Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. In *2005 IEEE congress on evolutionary computation*, volume 2, pages 1785–1791. IEEE, 2005.

A. K. Qin, V. L. Huang, e P. N. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE transactions on Evolutionary Computation*, 13(2):398–417, 2008.

S. Rahnamayan, H. R. Tizhoosh, e M. M. Salama. Opposition-based differential evolution. *IEEE Transactions on Evolutionary computation*, 12(1):64–79, 2008.

D. RIBEIRO e F. NEVES. Dicio–dicionário online de português, 2022.

M. T. Ribeiro, S. Singh, e C. Guestrin. "why should i trust you?"explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

R. Rivera-Lopez, J. Canul-Reich, J. A. Gámez, e J. M. Puerta. OC1-DE: a differential evolution based approach for inducing oblique decision trees. In *International Conference on Artificial Intelligence and Soft Computing*, pages 427–438. Springer, 2017.

R. Rivera-Lopez, J. Canul-Reich, E. Mezura-Montes, e M. A. Cruz-Chávez. Induction of decision trees as classification models through metaheuristics. *Swarm and Evolutionary Computation*, 69:101006, 2022.

M. Robnik-Šikonja e M. Bohanec. Perturbation-based explanations of prediction models. In *Human and machine learning*, pages 159–175. Springer, 2018.

M. Robnik-Šikonja e I. Kononenko. Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589–600, 2008.

J. Rönkkönen et al. *ContinuousMultimodal Global Optimization with Differential Evolution-Based Methods*. Lappeenranta University of Technology, 2009.

R. Roscher, B. Bohn, M. F. Duarte, e J. Garcke. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 8:42200–42216, 2020.

A. Rosenfeld e A. Richardson. Explainability in human–agent systems. *Autonomous Agents and Multi-Agent Systems*, 33(6):673–705, 2019.

W. R. Rudnicki, M. Kierczak, J. Koronacki, e J. Komorowski. A statistical method for determining importance of variables in an information system. In *International Conference on Rough Sets and Current Trends in Computing*, pages 557–566. Springer, 2006.

W. F. Sacco e N. Henderson. Differential evolution with topographical mutation applied to nuclear reactor core design. *Progress in Nuclear Energy*, 70:140–148, 2014.

W. Samek, T. Wiegand, e K.-R. Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

S. S. Santos, M. A. Alves, L. A. Ferreira, e F. G. Guimaraes. PDTX: A novel local explainer based on the perceptron decision tree. In *Anais do 15o. Congresso Brasileiro de Inteligência Computacional*, pages 1–8, Joinville, SC, 2021.

J. Schrouff, S. Baur, S. Hou, D. Mincu, E. Loreaux, R. Blanes, J. Wexler, A. Karthikesalingam, e B. Kim. Best of both worlds: local and global explanations with human-understandable concepts. *arXiv preprint arXiv:2106.08641*, 2021.

M. Setzu, R. Guidotti, A. Monreale, F. Turini, D. Pedreschi, e F. Giannotti. Glocalx-from local to global explanations of black box ai models. *Artificial Intelligence*, 294:103457, 2021.

L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28): 307–317, 1953.

M. D. Shields e J. Zhang. The generalization of latin hypercube sampling. *Reliability Engineering & System Safety*, 148:96–108, 2016.

M. D. Shields, K. Teferra, A. Hapij, e R. P. Daddazio. Refined stratified sampling for efficient monte carlo based uncertainty quantification. *Reliability Engineering & System Safety*, 142:310–325, 2015.

K. Sokol, A. Hepburn, R. Santos-Rodriguez, e P. Flach. bLIMEy: surrogate prediction explanations beyond lime. *arXiv preprint arXiv:1910.13016*, 2019.

Z. Somogyi. *The Application of Artificial Intelligence: Step-by-Step Guide from Beginner to Expert*. Springer Nature, 2021.

M. Staniak e P. Biecek. Explanations of model predictions with live and breakdown packages. *arXiv preprint arXiv:1804.01955*, 2018.

V. Stanovov, S. Akhmedova, e E. Semenkin. Differential evolution with linear bias reduction in parameter adaptation. *Algorithms*, 13(11):283, 2020.

R. Storn e K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

P. Svenmarck, L. Luotsinen, M. Nilsson, e J. Schubert. Possibilities and challenges for artificial intelligence in military applications. In *Proceedings of the NATO Big Data and Artificial Intelligence for Military Decision Making Specialists' Meeting*, pages 1–16. Neuilly-sur-Seine France, 2018.

R. Tanabe e A. Fukunaga. Success-history based parameter adaptation for differential evolution. In *2013 IEEE Congress on Evolutionary Computation*, pages 71–78, 2013a. doi: 10.1109/CEC.2013.6557555.

R. Tanabe e A. Fukunaga. Success-history based parameter adaptation for differential evolution. In *2013 IEEE congress on evolutionary computation*, pages 71–78. IEEE, 2013b.

R. Tanabe e A. S. Fukunaga. Improving the search performance of shade using linear population size reduction. In *2014 IEEE congress on evolutionary computation (CEC)*, pages 1658–1665. IEEE, 2014.

M. Tayefi, H. Esmaeili, M. S. Karimian, A. A. Zadeh, M. Ebrahimi, M. Safarian, M. Nematy, S. M. R. Parizadeh, G. A. Ferns, e M. Ghayour-Mobarhan. The application of a decision tree to establish the parameters associated with hypertension. *Computer methods and programs in biomedicine*, 139:83–91, 2017.

D. Tech. Como funciona o algoritmo Árvore de decisão, 2019. Disponível em: https://didatica.tech/como-funciona-o-algoritmo-arvore-de-decisao/. Acesso em: 16 de maio de 2022.

Y. Tian e G. Liu. Mane: Model-agnostic non-linear explanations for deep learning model. In *2020 IEEE World Congress on Services (SERVICES)*, pages 33–36, 2020. doi: 10.1109/SERVICES48979.2020.00021.

C. A. Tompkins. Using and interpreting linear regression and correlation analyses: Some cautions and considerations. 1993.

M. Van Lent, W. Fisher, e M. Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the national conference on artificial intelligence*, pages 900–907. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

G. Visani, E. Bagli, e F. Chesani. OptiLIME: Optimized lime explanations for diagnostic computer algorithms. *arXiv preprint arXiv:2006.05714*, 2020a.

G. Visani, E. Bagli, F. Chesani, A. Poluzzi, e D. Capuzzo. Statistical stability indices for lime: obtaining reliable explanations for machine learning models. *Journal of the Operational Research Society*, pages 1–11, 2020b.

B. Vukobratović e R. Struharik. Evolving full oblique decision trees. In *2015 16th IEEE International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 95–100. IEEE, 2015.

S. Wachter, B. Mittelstadt, e C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31:841, 2017.

Y. Wang, Z. Cai, e Q. Zhang. Enhancing the search ability of differential evolution through orthogonal crossover. *Information Sciences*, 185(1):153–177, 2012.

Z. J. Wang, R. Turko, O. Shaikh, H. Park, N. Das, F. Hohman, M. Kahng, e D. H. P. Chau. Cnn explainer: Learning convolutional neural networks with interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1396–1406, 2020.

W. Wei, J. Wang, e M. Tao. Constrained differential evolution with multiobjective sorting mutation operators for constrained optimization. *Applied Soft Computing*, 33:207–222, 2015.

K. Weitz, D. Schiller, R. Schlagowski, T. Huber, e E. André. "do you trust me?"increasing user-trust by integrating virtual agents in explainable ai interaction design. In *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents*, pages 7–9, 2019.

D. C. Wickramarachchi, B. L. Robertson, M. Reale, C. J. Price, e J. Brown. Hhcart: an oblique decision tree. *Computational Statistics & Data Analysis*, 96:12–23, 2016.

T.-T. Wong, W.-S. Luk, e P.-A. Heng. Sampling with hammersley and halton points. *Journal of graphics tools*, 2(2):9–24, 1997.

D. A. Wood. A transparent open-box learning network provides insight to complex systems and a performance benchmark for more-opaque machine learning algorithms. *Advances in Geo-Energy Research*, 2(2):148–162, 2018.

M.-C. Wu, S.-Y. Lin, e C.-H. Lin. An effective application of decision tree to stock trading. *Expert Systems with applications*, 31(2):270–274, 2006.

W. Wu, Y. Su, X. Chen, S. Zhao, I. King, M. R. Lyu, e Y.-W. Tai. Towards global explanations of convolutional neural networks with concept attribution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

P. Xenopoulos, G. Chan, H. Doraiswamy, L. G. Nonato, B. Barr, e C. Silva. Topological representations of local explanations. *arXiv preprint arXiv:2201.02155*, 2022.

C. A. C. Yahaya, A. Firdaus, S. Mohamad, F. Ernawan, e M. F. Ab Razak. Automated feature selection using boruta algorithm to detect mobile malware. *International Journal*, 9(5), 2020.

Z. Yang, K. Tang, e X. Yao. Scalability of generalized adaptive differential evolution for large-scale continuous optimization. *Soft Computing*, 15(11):2141–2155, 2011.

R. Ying, D. Bourgeois, J. You, M. Zitnik, e J. Leskovec. Gnn explainer: A tool for post-hoc explanation of graph neural networks. *arXiv preprint arXiv:1903.03894*, 2019.

X. Yu, M. Cai, e J. Cao. A novel mutation differential evolution for global optimization. *Journal of Intelligent & Fuzzy Systems*, 28(3):1047–1060, 2015.

M. R. Zafar e N. M. Khan. DLIME: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems. *arXiv preprint arXiv:1906.10263*, 2019.

D. Zaharie. A comparative analysis of crossover variants in differential evolution. In *Proceedings of IMCSIT*, volume 2007, pages 171–181, 2007.

B. M. Zázvorka. Application of decision trees to failure detection in hvac systems. 2020.

C. Zednik. Solving the black box problem: a normative framework for explainable artificial intelligence. *Philosophy & Technology*, 34(2):265–288, 2021.

S. Zeldam. Automated failure diagnosis in aviation maintenance using explainable artificial intelligence (xai). Master's thesis, University of Twente, 2018.

J. Zhang e A. C. Sanderson. Jade: Self-adaptive differential evolution with fast and reliable convergence performance. In *2007 IEEE congress on evolutionary computation*, pages 2251–2258. IEEE, 2007.

J. Zhang e A. C. Sanderson. Jade: adaptive differential evolution with optional external archive. *IEEE Transactions on evolutionary computation*, 13(5):945–958, 2009.

L. Zhang e P. N. Suganthan. Oblique decision tree ensemble via multisurface proximal support vector machine. *IEEE transactions on cybernetics*, 45(10):2165–2176, 2014.

X. Zhao, W. Huang, X. Huang, V. Robu, e D. Flynn. Baylime: Bayesian local interpretable model-agnostic explanations. In *Uncertainty in Artificial Intelligence*, pages 887–896. PMLR, 2021.

Z. Zhou, G. Hooker, e F. Wang. S-lime: Stabilized-lime for model explanation. *arXiv preprint arXiv:2106.07875*, 2021.