Pontifícia Universidade Católica
do Rio de Janeiro

Moises Henrique Pereira

OceanUI: interface para geração de explicações contrafactuais

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós-graduação em Informática da PUC-Rio.

Orientador : Profa. Simone Diniz Junqueira Barbosa
Coorientador: Prof. Thibaut Victor Gaston Vidal

Rio de Janeiro
abril de 2022

Pontifícia Universidade Católica
do Rio de Janeiro

Moises Henrique Pereira

OceanUI: interface para geração de explicações contrafactuais

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós–graduação em Informática da PUC-Rio. Aprovada pela Comissão Examinadora abaixo:

Profa. Simone Diniz Junqueira Barbosa
Orientador
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

Prof. Thibaut Victor Gaston Vidal
Coorientador
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

Prof. Bruno Feijo
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

Prof. Hélio Côrtes Vieira Lopes
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

Dr. Marx Leles Viana
Pesquisador Autônomo

Rio de Janeiro, 5 de abril de 2022

Moises Henrique Pereira

Bacharel em Ciência da Computação na UFV, tendo já feito outros 2 anos em Matemática Licenciatura também na UFV. Foi bolsista CNPq no projeto do Editor de Metadados Geoespaciais do Brasil Sumarizado (edpMGBs). Trabalhou como estagiário voluntário no Laboratório de Métodos Epidemiológicos e Computacionais em Saúde (LMECS-UFV) na área de simulação in silico de sistemas biológicos (AutoSimmune). Atuou em projeto de acessibilidade para surdos pela CEAD-UFV. Atualmente atua como Mestrando em Ciência da Computação com especialização em Ciência de Dados pela PUC-Rio e trabalha como estagiário no Laboratório de Engenharia de Software (LES) também pela PUC-Rio.

Ficha Catalográfica

Pereira, Moises Henrique

OceanUI: interface para geração de explicações contrafactuais / Moises Henrique Pereira; orientador: Simone Diniz Junqueira Barbosa; coorientador: Thibaut Victor Gaston Vidal. – 2022.

85 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2022.

Inclui bibliografia

1. explicação countrafactual – Teses. 2. visualização de dados – Teses. 3. tree ensembles – Teses. 4. explicação countrafactual. 5. visualização de dados. 6. tree ensembles. I. Diniz Junqueira Barbosa, Simone. II. Vidal, Thibaut. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

À minha família.

Agradecimentos

Agradeço a Deus por todo cuidado e oportunidades que tem me dado.

Agradeço aos meus pais Luiz Claudio e Ana Paula, e meus irmãos Samuel, Eliseu e Raphael por todo apoio, carinho e orações.

Agradeço aos meus amigos e colegas que estiveram comigo nesta caminhada e sempre me ajudaram, tanto os do mestrado, do LES e da UFV.

Agradeço aos meus orientadores, professora Simone e professor Thibaut, por todas as orientações, paciência e experiência que me passaram.

Agradeço à PUC-Rio pela oportunidade de cursar este mestrado. E a todos os profissionais do Departamento de Informática.

Finalmente, gostaria de agradecer à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pelo financiamento parcial deste trabalho sob o processo 133262/2020-0.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

Resumo

Pereira, Moises Henrique; Diniz Junqueira Barbosa, Simone; Vidal, Thibaut. **OceanUI: interface para geração de explicações contrafactuais**. Rio de Janeiro, 2022. 85p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Atualmente algoritmos de aprendizado de máquina (ML) estão incrivelmente presentes no nosso cotidiano, desde sistemas de recomendação de filmes e músicas até áreas de alto risco como saúde, justiça criminal, finanças e assim por diante, auxiliando na tomada de decisões. Mas a complexidade de criação desses algoritmos de ML também está aumentando, enquanto sua interpretabilidade está diminuindo. Muitos algoritmos e suas decisões não podem ser facilmente explicados por desenvolvedores ou usuários, e os algoritmos também não são autoexplicáveis. Com isso, erros e vieses podem acabar ficando "ocultos," o que pode impactar profundamente a vida das pessoas. Devido a isso, iniciativas relacionadas a transparência, explicabilidade e interpretabilidade estão se tornando cada vez mais relevantes, como podemos ver no novo regulamento sobre proteção e tratamento de dados pessoais (GDPR, do inglês General Data Protection Regulation), aprovado em 2016 para a União Europeia, e também na Lei Geral de Proteção de Dados (LGPD) aprovada em 2020 no Brasil. Além de leis e regulamentações tratando sobre o tema, diversos autores consideram necessário o uso de algoritmos inerentemente interpretáveis; outros mostram alternativas para se explicar algoritmos caixa-preta usando explicações locais, tomando a vizinhança de um determinado ponto e então analisando a fronteira de decisão dessa região; enquanto ainda outros estudam o uso de explicações contrafactuais. Seguindo essa linha dos contrafactuais, nos propomos a desenvolver uma interface com usuário para o sistema "Optimal Counterfactual Explanations in Tree Ensembles" (OCEAN), denominada OceanUI, através do qual o usuário gera explicações contrafactuais plausíveis usando Programação Inteira Mista e *Isolation Forest*. O propósito desta interface é facilitar a geração de contrafactuais e permitir ao usuário obter um contrafactual personalizado e mais aplicável individualmente, por meio da utilização de restrições e gráficos interativos.

Palavras-chave

explicação countrafactual; visualização de dados; tree ensembles.

Abstract

Pereira, Moises Henrique; Diniz Junqueira Barbosa, Simone (Advisor); Vidal, Thibaut (Co-Advisor). **OceanUI: interface for counterfactual explanations generation**. Rio de Janeiro, 2022. 85p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Machine learning algorithms (ML) are becoming incredibly present in our daily lives, from movie and song recommendation systems to high-risk areas like health care, criminal justice, finance, and so on, supporting decision making. But the complexity of those algorithms is increasing while their interpretability is decreasing. Many algorithms and their decisions cannot be easily explained by either developers or users, and the algorithms are also not self-explanatory. As a result, mistakes and biases can end up being "hidden," which can profoundly impact people's lives. So, initiatives concerning transparency, explainability, and interpretability are becoming increasingly more relevant, as we can see in the General Data Protection Regulation (GDPR), approved in 2016 for the European Union, and in the General Data Protection Law (LGPD) approved in 2020 in Brazil. In addition to laws and regulations, several authors consider necessary the use of inherently interpretable algorithms; others show alternatives to explain black-box algorithms using local explanations, taking the neighborhood of a given point and then analyzing the decision boundary in that region; while yet others study the use of counterfactual explanations. Following the path of counterfactuals, we propose to develop a user interface for the system "Optimal Counterfactual Explanations in Tree Ensembles" (OCEAN), which we call OceanUI, through which the user generates plausible counterfactual explanations using Mixed Integer Programming and *Isolation Forest*. The purpose of this user interface is to facilitate the counterfactual generation and to allow the user to obtain a personal and more individually applicable counterfactual, by means ofrestrictions and interactive graphics.

Keywords

counterfactual explanation; data visualization; tree ensemble.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

FATE – Fairness, Accountability, Transparency and Ethics (Justiça, Responsabilidade, Transparência e Ética)

IA – Inteligência Artificial

IHC – Interação Humano-Computador

LVQ – Learning Vector Quantization

ML – Machine Learning (Aprendizado de Máquina)

RF – Random Forest

1
Introdução

1.1
Motivação

Atualmente algoritmos de ML estão incrivelmente presentes no nosso cotidiano, desde sistemas de recomendação de filmes e músicas até áreas de alto risco como saúde, justiça criminal, finanças e assim por diante, auxiliando na tomada de decisões. Mas a complexidade de criação desses algoritmos de ML também está aumentando, enquanto sua interpretabilidade está diminuindo (Burkart and Huber, 2021; Bibal et al., 2021). Muitos algoritmos e suas decisões não podem ser facilmente explicados por desenvolvedores ou usuários, e os algoritmos também não são autoexplicáveis. Além disso, tanto os dados quanto os algoritmos podem estar enviesados, o que aumenta ainda mais a dificuldade de entendimento (Sun et al., 2020). Logo, se esses erros e vieses ficarem "ocultos," isso pode impactar profundamente a vida das pessoas (Carvalho et al., 2019).

Mehrabi et al. (2021) nos mostram alguns exemplos de como esses vieses podem ser prejudiciais. No primeiro caso, foi descoberto que o sistema usado em cortes norte americanas, Correctional Offender Management Profiling for Alternative Sanctions (COMPAS), estava sendo injusto com pessoas afrodescendentes, uma vez que atribuía um maior risco de reincidência (cometer novos crimes) para tais pessoas quando comparado com outras de mesmo perfil, porém de diferentes etnias. Outro exemplo é relacionado com câmeras digitais e seu sistema de reconhecimento facial; nesse caso, o sistema assumia erroneamente que pessoas asiáticas estavam com seus olhos fechados ou piscando.

Outro exemplo é o caso do "Tay" um sistema de *chatbot* da Microsoft que apresentou comportamentos racistas e sexistas com apenas dezesseis horas de interação, e por isto teve de ser removido (Fuchs, 2018). Além desses, temos também o exemplo de um sistema que auxilia hospitais e seguradoras americanos, cujo propósito era fornecer cuidados especializados a pessoas com doenças crônicas, de modo que elas fossem mais bem atendidas e com custo reduzido. Entretanto, foi observado que pessoas negras tiveram que pagar mais

para ter os mesmos níveis de necessidades atendidas que pessoas brancas.[1]

Dados esses problemas e vários outros que podemos encontrar facilmente não apenas na literatura acadêmica, mas também na jornalística, iniciativas relacionadas a transparência, explicabilidade e interpretabilidade estão se tornando cada vez mais relevantes (Mehrabi et al., 2021). Exemplos disso são o novo regulamento sobre proteção e tratamento de dados pessoais (GDPR, do inglês *General Data Protection Regulation*), aprovado em 2016 para a União Europeia (Goodman and Flaxman, 2017); e a Lei Geral de Proteção de Dados (LGPD)[2] aprovada em 2020 no Brasil.

Além de leis tratando sobre isso, diversos autores apelam para a necessidade de FATE (Fairness, Accountability, Transparency and Ethics) nos sistemas de ML por meio da consideração, do incentivo e do ensino sobre o tema no âmbito acadêmico e também no profissional, como dizem Bates et al. (2020), Bogina et al. (2021) e Kasinidou et al. (2021). Pois desta forma os direitos e valores humanos seriam considerados sobre estes sistemas.

E outros autores, como Das and Rad (2020) e Rudin (2019), consideram necessário o uso de algoritmos inerentemente interpretáveis em vez de se usar algoritmos complexos e então explicá-los. Adebayo et al. (2018), Guidotti et al. (2019) e White and Garcez (2019) mostram alternativas para se explicar algoritmos caixa-preta usando explicações locais, tomando a vizinhança de um determinado ponto e então analisando a fronteira de decisão dessa região. Já Stepin et al. (2021) e Verma et al. (2020) estudam o uso de explicações contrafactuais, suas vantagens e desvantagens.

A ideia principal de uma explicação contrafactual é responder à pergunta: "Qual é a mudança mínima que um dado de entrada precisa sofrer para se obter a saída desejada?". Por exemplo: "O que preciso alterar em meus dados para obter um empréstimo negado anteriormente por um banco?" (Artelt and Hammer, 2019).

Outro exemplo é o seguinte: alguém quer alugar um apartamento e espera ganhar mais de $R\$1000$ de aluguel, mas o valor máximo que pode ser alcançado é de $R\$900$. Então, o que pode ser mudado/reformado nesse apartamento para atingir o valor de aluguel desejado? Uma possível explicação contrafactual poderia dizer que se aumentar o tamanho do apartamento em $15m^2$, então o aluguel pode aumentar para $R\$1000$, mas isso é um problema porque um apartamento não pode ter seu tamanho aumentado; trata-se de um contrafactual *não acionável*. Outra explicação contrafactual pode sugerir

[1] Último acesso: 14/04/2022. `https://www.scientificamerican.com/article/racial-bias-found-in-a-major-health-care-risk-algorithm/`

[2] Último acesso: 14/04/2022. `http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm`

permitir animais de estimação e melhorar o isolamento acústico, o que é possível e relativamente fácil de atender (Molnar, 2020).

Seguindo essa linha, Parmentier and Vidal (2021) desenvolveram o "Optimal Counterfactual Explanations in Tree Ensembles" (OCEAN). Esse trabalho permite ao usuário gerar explicações contrafactuais usando Programação Inteira Mista e Isolation Forest para garantir a sua plausibilidade, pois oferece uma visão que evita explicações extremas (*outliers*). Além disso, contribui para a construção de modelos matemáticos eficientes usando *Tree Ensembles* e fornece um código que pode ser ajustado às necessidades específicas do usuário.

Como o OCEAN usa linha de comando para a geração das explicações, uma melhoria sobre ele é a implementação de uma interface gráfica para ajudar os usuários a produzir e "negociar" uma explicação contrafactual útil. A ideia da "negociação" significa que o usuário pode impor restrições sobre o modelo de modo a obter explicações mais personalizadas. Voltemos àquele exemplo do aluguel de um apartamento: imagine uma terceira explicação contrafactual que informa que, para se obter o aluguel no valor de R\$1000, o dono do apartamento deve aumentar em 4 o número de janelas (vamos partir da ideia de que inclusão de janelas é permitida) mas, ao ver essa alternativa, o dono do apartamento informa ao sistema que pode incluir no máximo 3 novas janelas. Então com essa nova restrição, o sistema calcula um novo contrafactual. Como esse processo pode ser repetido quantas vezes for necessário para todas as *features* disponíveis, o contrafactual resultante será mais personalizado e adequado às necessidades do usuário.

1.2 Explicação Contrafactual

A ideia principal de uma explicação contrafactual é responder à pergunta: "Qual é a mudança mínima que um dado de entrada precisa sofrer para se obter a saída desejada?". Por exemplo: "O que preciso alterar em meus dados para obter um empréstimo negado anteriormente por um banco?" (Artelt and Hammer, 2019).

Outro exemplo é o seguinte: alguém quer alugar um apartamento e espera ganhar mais de R\$1000 de aluguel, mas o valor máximo que pode ser alcançado é de R\$900. Então, o que pode ser mudado/reformado nesse apartamento para atingir o valor de aluguel desejado? Uma possível explicação contrafactual poderia dizer que se aumentar o tamanho do apartamento em $15m^2$, então o aluguel pode aumentar para R\$1000, mas isso é um problema porque um apartamento não pode ter seu tamanho aumentado; trata-se de um contrafactual *não acionável*. Outra explicação contrafactual pode sugerir

permitir animais de estimação e melhorar o isolamento acústico, o que é possível e relativamente fácil de atender (Molnar, 2020).

Com isto, podemos evidenciar algumas propriedades, por exemplo: nem toda explicação contrafactual é necessariamente representativa, ou seja, uma explicação gerada pode acabar sendo um *outlier* em relação ao conjunto de dados e nem toda explicação contrafactual é necessariamente factível, ou seja, pode sugerir mudanças impossíveis na prática (Poyiadzi et al., 2020).

Devido a estas observações, algo que deve ser levado em conta no processo de geração de explicações de contrafactuais é justamente a necessidade de plausibilidade/factibilidade e também a necessidade de diversidade (quando trata-se de conjunto de explicações contrafactuais) como dizem Artelt and Hammer (2020) e Mothilal et al. (2020).

Uma vez que a explicação contrafactual gerada é entendível e factível, por parte do usuário, o mesmo ganha um certo poder em relação aos modelos de ML, como: o usuário pode entender quais *features* afetam mais fortemente o modelo de predição, pode ganhar um embasamento extra caso perceba uma injustiça por parte do modelo de predição, além de ter um *feedback* sobre o que pode estar mudando nos seus dados de entrada para obter a predição desejada (Verma et al., 2020).

1.3 Metodologia

Inicialmente projetamos uma versão de geração instantânea de contrafactuais, na qual o usuário fornece um ponto e, opcionalmente, as restrições desejadas, e então recebe uma comparação entre o ponto fornecido e o contrafactual encontrado.

Posteriormente, desenvolvemos duas versões de geração iterativa de contrafactuais, através das quais o usuário pode ir "negociando" com o sistema durante o processo de busca do contrafactual.

A primeira versão iterativa foi pensada para usar um gráfico $2D$, o ponto fornecido pelo usuário, uma amostragem da vizinhança deste ponto e a distância entre cada um dos pontos e um contrafactual mais próximo, de modo que o usuário pudesse selecionar um ponto em cada iteração, de modo a ir se aproximando de um contrafactual adequado às suas necessidades.

A versão final substitui o gráfico $2D$ por um gráfico de coordenadas paralelas e substitui a amostragem da vizinhança do ponto do usuário por até mais três pontos: o ponto original, um contrafactual sugerido e um ponto para indicar quais foram as últimas alterações feitas em um cenário anterior, caso haja. Além disso, ambas as versões iterativas permitem a inclusão de

restrições. Com isso o presente trabalho se propõe a implementar e documentar tal interface gráfica.

Para embasar o desenvolvimento dessa interface, nós consideramos Gomez et al. (2020) e Piesanen and Kerrigan (2020) como as referências mais fortemente relacionadas com nosso projeto, pois se propõem a apresentar formas de preenchimento de campos e visualização de contrafactuais que sejam fáceis de entender e de usar.

1.4 Contribuições Esperadas

A produção deste trabalho objetiva facilitar a geração de explicações contrafactuais e com isso auxiliar na disseminação da proposta de explicabilidade para algoritmos de ML. Além disso, visa a colaborar para a redução de injustiças causadas por vieses e erros em algoritmos.

1.5 Organização deste Documento

Este documento apresenta o desenvolvimento do trabalho "OceanUI" no seguinte formato: o Capítulo 2 apresenta os trabalhos relacionados, se concentrando nos tópicos Contrafactuais únicos, Conjunto contrafactual e Interfaces para contrafactuais. No Capítulo 3 Implementação do Sistema. No Capítulo 4 descrevemos o processo de Planejamento da Avaliação, seus Resultados e Discussão. E finalmente, o Capítulo 5 retoma as principais contribuições deste trabalho e traça rumos para trabalhos futuros.

2
Trabalhos Relacionados

Este capítulo apresenta alguns dos trabalhos existentes na literatura que se relacionam com a nossa pesquisa. Dentre eles, alguns se propõem a mostrar o contrafactual como o ponto mais próximo, enquanto outros escolhem pela abordagem de múltiplos contrafactuais, também conhecido como conjunto contrafactual. Além disso, também apresentamos alguns trabalhos que desenvolvem interfaces para a geração de contrafactuais.

2.1
Contrafactuais únicos

Artelt and Hammer (2019) apresentam a geração de contrafactuais usando *Learning Vector Quantization* (LVQ). Com isso, criando um conjunto de protótipos a partir dos dados iniciais catalogados e utilizando uma função de distância, o contrafactual é calculado como sendo o protótipo mais próximo em relação a um dado ponto de entrada, e que possua classe diferente deste ponto. Em relação à função de distância, o artigo explora a euclidiana, a de Manhattan e também a baseada em matriz LVQ (matriz de distâncias personalizada para determinada classe ou protótipo).

Artelt and Hammer (2020) apresentam a geração de contrafactuais que levam em conta regiões de alta densidade. A ideia dessa abordagem é que, se o contrafactual está em uma área densa, é mais provável que ele seja mais plausível (dada a maior representatividade dessas regiões) se comparado com outro contrafactual que esteja em uma região de baixa densidade. Entretanto, vale ressaltar que, por causa disso, os contrafactuais podem apresentar uma distância maior do ponto inicial. Um outro limitante dessa abordagem pode ser visto quando os dados são muito esparsos. Em relação à análise de densidade, o artigo faz uso da função de mistura gaussiana.

Poyiadzi et al. (2020) apresentam a geração de contrafactuais usando grafos ponderados, cujos vértices representam os pontos e as arestas conectam pontos próximos (dada uma função de distância que também leva em conta a densidade da região). Vale observar que o número de arestas desse grafo está sujeito a um limiar de distância. Além disso, o usuário também pode inserir restrições para "guiar" o processo de busca do contrafactual. A ideia de se fazer

um grafo considerando todas essas restrições visa que o grafo resultante tenha informações de densidade onde o contrafactual será encontrado e também da densidade do caminho até esse contrafactual. O raciocínio subjacente é que é mais provável que o contrafactual seja factível quando, além de estar em uma região densa, ele também tenha passado por um caminho de alta densidade. Uma vez gerado esse grafo, o autor utiliza um algoritmo de caminho mínimo para encontrar o contrafactual.

Van Looveren and Klaise (2019) apresentam a geração de contrafactuais usando protótipos, que podem ser obtidos por meio de uma média considerando uma determinada vizinhança de pontos. Como o processo de geração de contrafactual ocorre por meio de pequenas perturbações no ponto de entrada, isso pode ser muito custoso, então o uso de protótipos guia e acelera esse processo, além de possibilitar a geração de contrafactuais mais interpretáveis.

2.2 Conjunto contrafactual

Dandl et al. definem a geração de contrafactuais como um problema de otimização com múltiplos objetivos. Eles utilizam o *Nondominated Sorting Genetic Algorithm* II (NSGA-II (Pratap et al., 2002)), onde os genes são definidos como um vetor com os valores das *features* e a função de avaliação é a minimização de uma função que leva em conta quatro fatores: (1) a distância entre a classe do ponto de origem e a classe do contrafactual; (2) a função de distância de *Gower* para quantificar as características do contrafactual em função de sua região; (3) o número de *features* que foram mudadas; e (4) a média ponderada da distância de *Gower* entre o contrafactual e os k vizinhos mais próximos. Com isso, o sistema proposto só pára o processo de geração de populações de potenciais contrafactuais, levando em conta as mutações sobre eles, quando um número arbitrário de repetições é alcançado ou quando não é mais observada uma melhora significativa. Quando isso ocorre, a última população é tida como o conjunto diverso de contrafactuais, dado o ponto inicial.

Fernández et al. (2020) apresentam a geração de contrafactuais usando *Random Forest*. A ideia inicialmente é reduzir todas as árvores da RF em apenas uma, mas como o processo de redução pode ser muito custoso computacionalmente, é feita apenas uma redução parcial. Após essa redução, identifica-se uma sub-região do espaço de *features* onde um contrafactual (correspondente a um ponto de entrada) mantém sua classe e, usando os pontos dessa sub-região, é possível gerar um contrafactual com *ranges* (faixas) nos valores de suas *features*. Dessa forma, o contrafactual se torna mais abrangente, pois em vez de

um simples valor, o usuário descobre limiares de valores correspondentes ao contrafactual.

Mothilal et al. (2020) apresentam a geração de conjunto contrafactual em vez de apenas um único ponto, pois segundo o autor, apresentar um conjunto com mais de uma alternativa permite que o usuário analise e escolha a melhor opção. Sendo assim, esse processo precisa gerar k contrafactuais que sejam diferentes entre si, mas que não se afastem da ideia de menor número de mudanças. Com isso, a geração desse conjunto leva em conta uma matriz de distâncias entre os possíveis contrafactuais, a distância entre eles e o ponto fornecido pelo usuário, o número de *features* que precisam ser mudadas nos exemplos, e também restrições impostas pelos usuários, por considerar que tais restrições são mais parecidas com a realidade e por isso agregam muito no processo de geração de contrafactuais mais factíveis.

Finalmente, Russell (2019) apresenta a geração de conjunto contrafactual usando Programação Inteira Mista. O objetivo é minimizar uma função de distância enquanto considera restrições sobre esse modelo. Algumas dessas restrições são responsáveis por cuidar das *features* binárias e outras são responsáveis pela diversidade do conjunto contrafactual. Para usar esse modelo de otimização, o autor explica que, em dados com *features* mistas, a noção de distância pode ficar um pouco confusa, então para resolver isso usa-se a ideia de *one-hot encoding*, em que todas as opções das *features* são tratadas como novas variáveis que podem assumir o valor 0 ou 1. Com isso, iterativamente, o modelo retorna individualmente os contrafactuais.

2.3 Interfaces para contrafactuais

Para embasar o desenvolvimento desta interface, nós consideramos alguns trabalhos como Gomez et al. (2020) e Piesanen and Kerrigan (2020). Ambos os sistemas permitem ao usuário fornecer pontos de entrada e obter explicações contrafactuais, seja uma explicação única ou um conjunto de explicações, mas utilizam diferentes soluções para a visualização destas explicações contrafactuais.

Com isto, de Gomez et al. (2020) (Figura 2.1), tomamos a ideia de barras para mostrar as *features* e seus valores, e também a ideia de permitir ou não mudanças sobre elas para representar a restrição de acionabilidade.

De Piesanen and Kerrigan (2020) (Figura 2.2 e Figura 2.3), levamos em conta a forma que usam para representar a restrição de acionabilidade e também a forma como capturam os dados do usuário, seja por um ponto aleatório do *dataset* ou pelo preenchimento dos campos correspondentes.

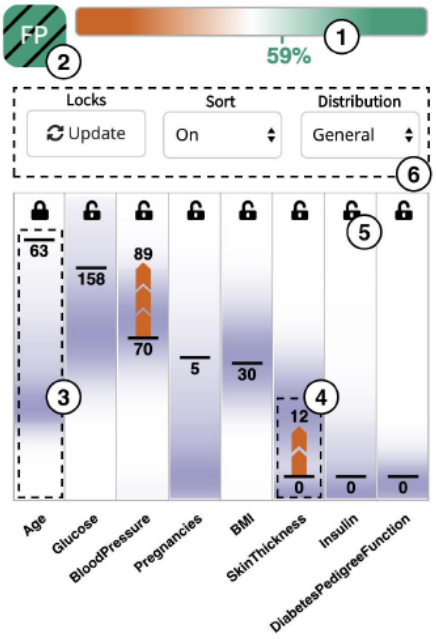

Figura 2.1: Interface do sistema ViCE (Gomez et al., 2020)

Além destas referências citadas acima, também estudamos a ferramenta What-if[1] do grupo People+AI Research da Google, que é uma ferramenta bem conhecida para o estudo de explicações contrafactuais, vide Figura 2.4. Nesta interface o usuário pode analisar o comportamento de algoritmos caixa preta e de modelos de regressão, além de visualizar e poder manipular os dados. A ferramenta pode ser usada por meio de Jupyter Notebooks.[2]

De Code (2021), consideramos a possibilidade de utilização de gráficos para mostrar o "caminhamento" da explicação contrafactual a medida que o usuário altera o ponto corrente, além da aparência do componente utilizado para informar as distâncias utilizadas na interface, que serve de inspiração para o componente de *features* binárias utilizado no OceanUI.

Kulesza et al. (2015) apresentam uma ferramenta capaz de explicar como as decisões de algoritmos foram feitas, de forma que o usuário obtenha um entendimento mais profundo, além de poder devolver informações extras para o sistema com o propósito de melhorá-lo. Neste trabalho destaca-se a ideia de interação constante do usuário, de modo a aperfeiçoar o modelo e melhorar a saída desejada.

Já com o sistema Dece (Cheng et al., 2020), o usuário pode explorar o modelo de predição e também as *features* do *dataset* utilizado para o modelo. O usuário pode incluir, editar e deletar pontos de um determinado subgrupo de estudo e, para cada ponto neste subgrupo, o sistema fornece uma explicação

[1]Último acesso: 14/04/2022. `https://pair-code.github.io/what-if-tool/`
[2]Último acesso: 14/04/2022. `https://jupyter.org/`

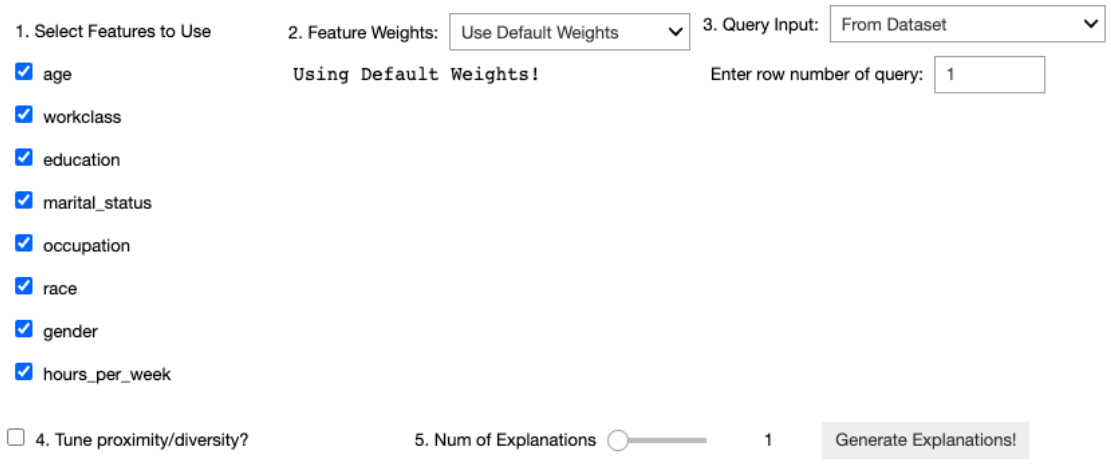

Figura 2.2: Interface do sistema VIDA Lab (Piesanen and Kerrigan, 2020)

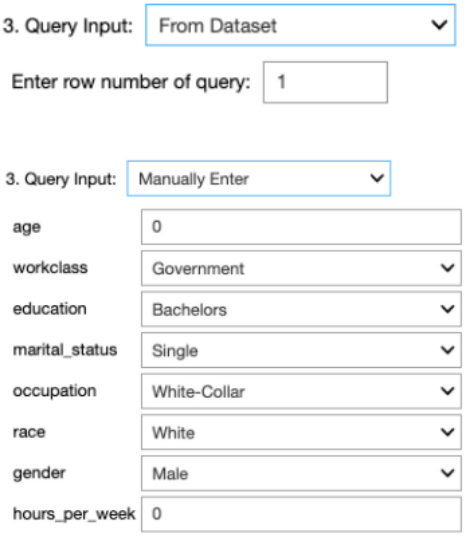

Figura 2.3: Interface de preenchimento de dados do sistema VIDA Lab (Piesanen and Kerrigan, 2020)

contrafactual.

E de Cheng et al. (2020) consideramos a utilização do gráfico de coordenadas paralelas em vez de um gráfico de coordenadas cartesianas, devido a possibilidade de apresentar mais informações e de uma forma relativamente simples.

Ao final do presente trabalho nos deparamos com a continuidade de ViCE (Gomez et al., 2020) que é o AdViCE (Gomez et al., 2021) (Figura 2.5), neste projeto observamos a abordagem utilizada para a manipulação de *features* numéricas e pudemos ver que a forma que desenvolvemos o nosso componente é promissora uma vez que é similar à feita por estes autores, além de embasar a necessidade e importância de imposição de restrições para personalização das explicações contrafactuais. E similar ao ViCE o AdViCE também permite

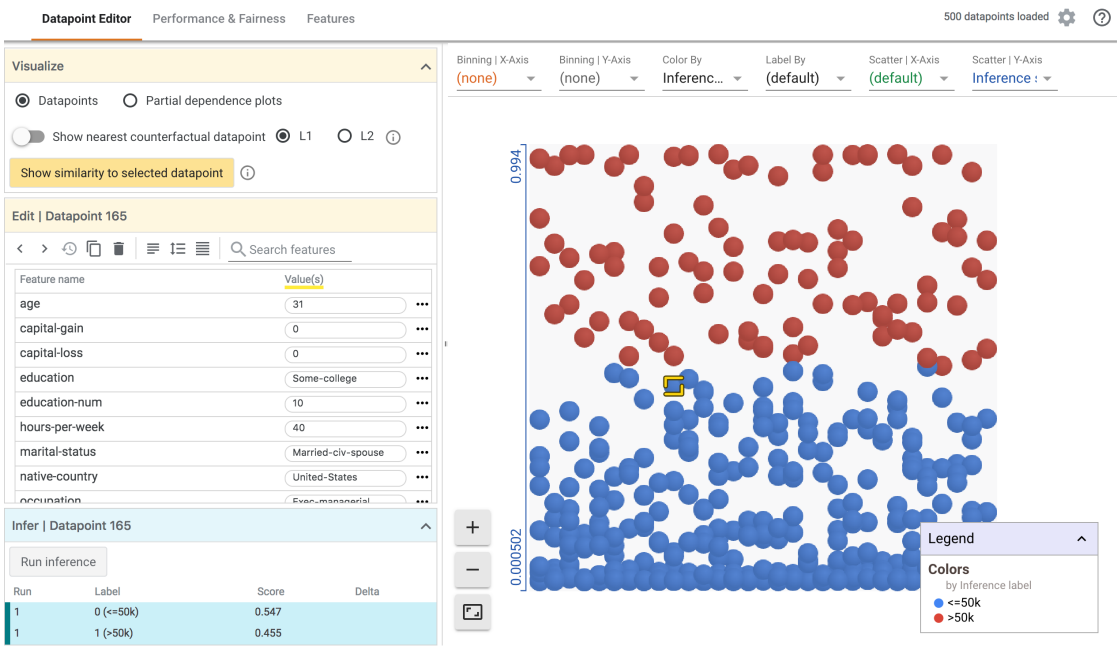

Figura 2.4: Interface do sistema What If (Code, 2021)

a geração de contrafactuais e as exibe de forma gráfica, além de permitir aos usuários impor algumas restrições e filtros sobre o modelo.

Vale destacar que após o estudo para o desenvolvimento desta seção de trabalhos relacionados, não conseguimos encontrar nenhum trabalho considerado o estado-da-arte, considerando a geração de explicações contrafactuais e exibição das mesmas.

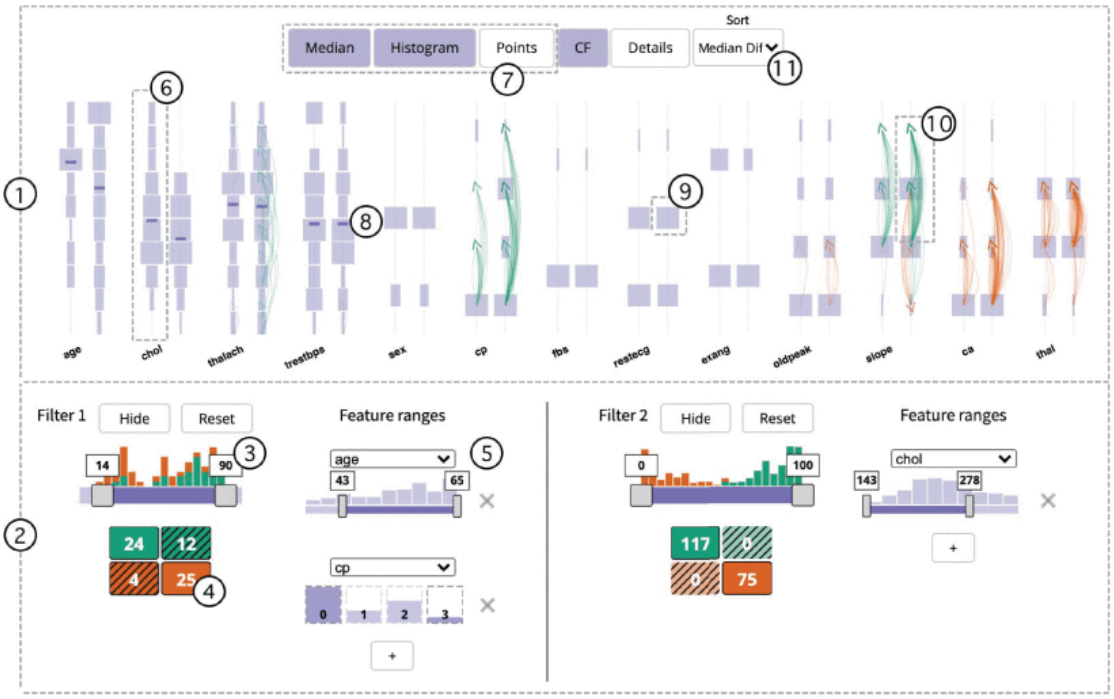

Figura 2.5: Interface do sistema AdViCE (Gomez et al., 2021)

3
Concepção e Desenvolvimento do OceanUI

Este capítulo apresenta a solução que desenvolvemos neste trabalho. Trata-se de uma interface que simplifica a geração de explicações contrafactuais. De modo que o usuário possa fornecer um ponto e em poucos passos obter uma explicação contrafactual que seja adaptada para o seu contexto, ou seja, durante o processo de geração deste contrafactual o usuário tem a possibilidade de informar restrições para ajustar a saída.

Inicialmente pensamos em uma opção de geração instantânea, onde o usuário insere um ponto e então o sistema gera em um único passo o contrafactual correspondente, fazendo um paralelo entre o dado de entrada e a explicação gerada. Mesmo sendo uma abordagem promissora que já permitiria ao usuário gerar contrafactuais e impor restrições para se obter um contrafactual personalizado, esta versão se mostrou simplista, pois o usuário não teria uma visão do processo de geração, uma vez que tudo ocorreria em um único passo.

Devido a esta limitação de não dar tanta liberdade ao usuário para manipular o ponto de entrada até um contrafactual desejado, pensamos em uma opção de geração iterativa e interativa, onde o usuário "guia" a explicação utilizando gráficos para analisar a distância do seu ponto de entrada e o contrafactual correspondente, podendo mudar interativamente e iterativamente os valores de seu ponto de modo a "caminhar" até uma explicação satisfatória.

Seguindo então está premissa de fornecer ao usuário o máximo de liberdade para a manipulação, inicialmente planejamos uma versão que utilizaria um gráfico de coordenadas cartesiano, onde o usuário poderia clicar sobre os pontos do gráfico e mover o ponto corrente até o valor desejado. Porém tal abordagem é limitada pelo número de *features* que poderiam ser exibidas simultaneamente, mesmo se considerássemos cor e tamanho dos pontos desenhados (para passar mais informações) ainda ficaríamos limitados, pois em alguns casos a explicação contrafactual pode sugerir mais mudanças do que um gráfico de coordenadas cartesiano pode exibir de uma vez.

Por isto, tomando como exemplo a representação fornecida por Cheng et al. (2020), chegamos à conclusão de que a parte gráfica deveria ser subs-

tituída por um gráfico de coordenadas paralelas. Tal mudança impactou os requisitos e o fluxo já definidos, como pode ser visto em anexo. Entretanto, tal mudança se mostra mais útil para se obter uma visão completa sobre o dado de entrada, sobre as mudanças sugeridas pelo sistema, as mudanças efetivadas pelo usuário e o quão perto se está de um contrafactual.

3.1 Implementação do Sistema

3.1.1 Geração Instantânea

Para atender os requisitos exigidos (vide anexo A), foram desenvolvidos alguns componentes, cada qual pensado para facilitar a interação do usuário. O componente de *features* binárias (Figura 3.1) apresenta de forma minimalista as duas opções disponíveis, similar à um componente encontrado em Code (2021) para seleção da distância desejada (distância usada para o cálculo de contrafactuais).

O componente de *features* categóricas (Figura 3.2 possui um *comboBox* para o usuário informar seu valor e também uma lista para informar quais valores são aceitáveis no contrafactual (e, consequentemente, quais não são), para este componente nos inspiramos tanto em Piesanen and Kerrigan (2020) quanto em Code (2021), pois apresentam *comboBoxes* para seleção de valores categóricos ou para seleção de eixos, porém acrescentamos mais informações para facilitar a usabilidade e melhorar a experiência do usuário.

E o componente de *features* numéricas (Figura 3.3) apresenta um *slider* com três ranges para o usuário informar os valores de mínimo e máximo aceitáveis e o valor correspondente do ponto atual, para este componente nos inspiramos em Gomez et al. (2020), pois apresenta a ideia de manipulação movendo um determinado ponto para um valor desejado. Além disso, todos os componentes possuem o campo *"actionable"* , para que o usuário possa informar se o valor de uma determinada *feature* pode ser alterado (*"actionable"* marcado) ou não (*"actionable"* desmarcado).

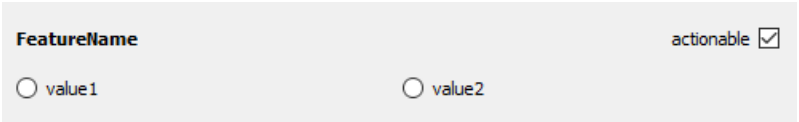

Figura 3.1: Componente usado para features binárias

Estes componentes são incluídos de forma dinâmica na interface após o usuário escolher com qual *dataset* irá trabalhar. Por isso são necessárias as

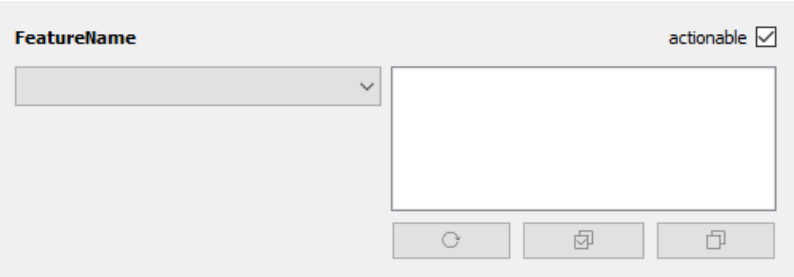

Figura 3.2: Componente usado para features categóricas

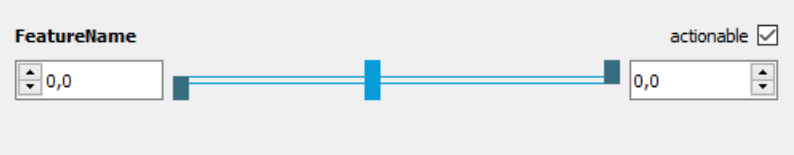

Figura 3.3: Componente usado para features numéricas

seguintes informações extras nos *datasets*:

- tipo da *feature*: para *features* binárias usamos B, para categóricas usamos C, para numéricas com valores discretos usamos D e para numéricas com valores contínuos usamos N
- acionabilidade da *feature*: para as *features* que não podem mudar seu valor usamos FIXED, para as que não possuem restrições na mudança do valor usamos FREE, para as que só podem ter seu valor aumentado usamos INC e para indicar qual representa a classe usamos PREDICT

Podemos ver na Figura 3.4 um exemplo dessas informações adicionais com um *dataset* que usamos no projeto.

```
AgeGroup,Race,Sex,PriorsCount,ChargeDegree,Class
D,B,B,D,B,B
INC,FREE,FIXED,FREE,FREE,PREDICT
```

Figura 3.4: Dataset COMPAS

Voltando ao preenchimento da interface com as *features* e suas informações, temos a Figura 3.5.

Com o cenário da Figura 3.5, o usuário pode (i) usar a opção *"Random Point"* para que o próprio sistema tome um ponto qualquer do *dataset* com classe 0 e preencha automaticamente todos os componentes; ou (ii) preencher todos os componentes manualmente. Após o preenchimento o usuário pode calcular a classe com a opção *"Calculate Class"* ou então gerar o contrafactual com o *"Generate Counterfactual"*. E após a geração do contrafactual o sistema apresenta uma comparação entre o ponto original e o contrafactual encontrado,

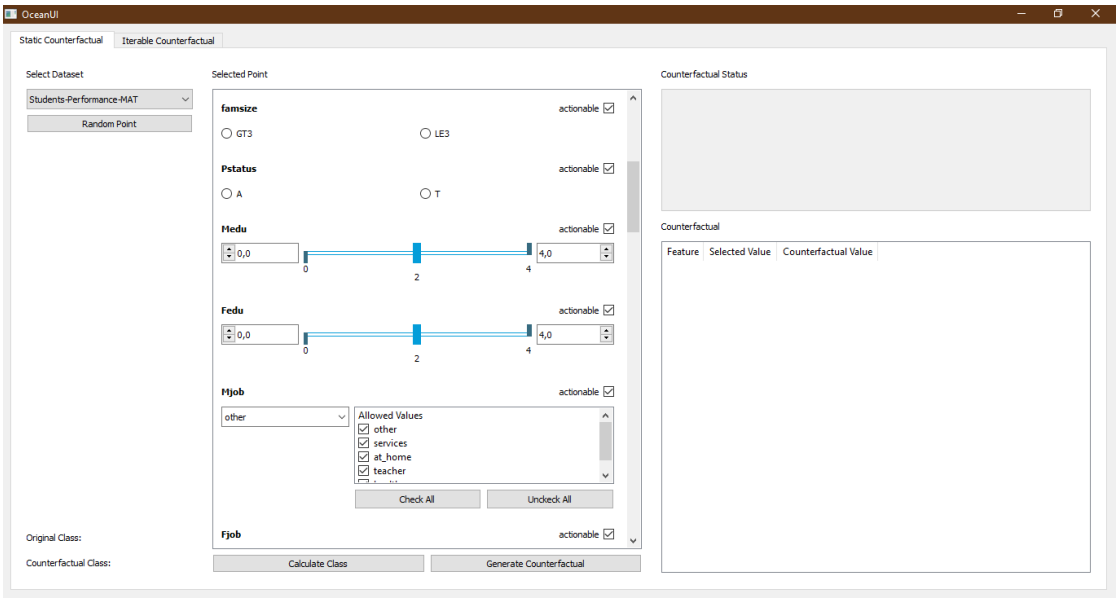

Figura 3.5: Interface de geração instantânea com um dataset selecionado

deixando em evidência os valores que precisam ser mudados, como mostra a Figura 3.6

3.1.2 Geração Iterativa Versão 1

Inicialmente temos a seguinte visão ao abrir a primeira versão da interface de geração iterativa, Figura 3.7.

Igual à versão instantânea, o usuário pode selecionar um *dataset* e, com o isso, o sistema preenche uma lista com os componentes de *features* e suas informações. Utilizando esses componentes, o usuário pode informar seu ponto de entrada ou apenas pegar um ponto aleatório do *dataset* cuja classe seja 0, Figura 3.8.

Após informar o ponto de entrada, o usuário pode utilizar a opção *"Next"* para instanciar um novo cenário, Figura 3.9. Nesse novo cenário, o usuário tem acesso à lista de componentes de *features* e pode impor restrições por meio deles. Entretanto, o usuário não pode utilizá-los para alterar o ponto corrente.

Ainda nesse cenário, o usuário pode selecionar um par de *features*, que serão consideradas os eixos X e Y de um plano cartesiano. Feito isto, o sistema plota um gráfico exibindo o valor do ponto corrente para as *features* selecionadas juntamente com uma amostragem da vizinhança deste ponto. O gráfico também faz a distinção dos pontos por classe, ou seja, os pontos com classe 0 são representados em azul, os com classe 1 são representados em laranja e o ponto fornecido pelo usuário é representado em verde. Por fim, o gráfico também fornece a informação de distância até um contrafactual mais próximo

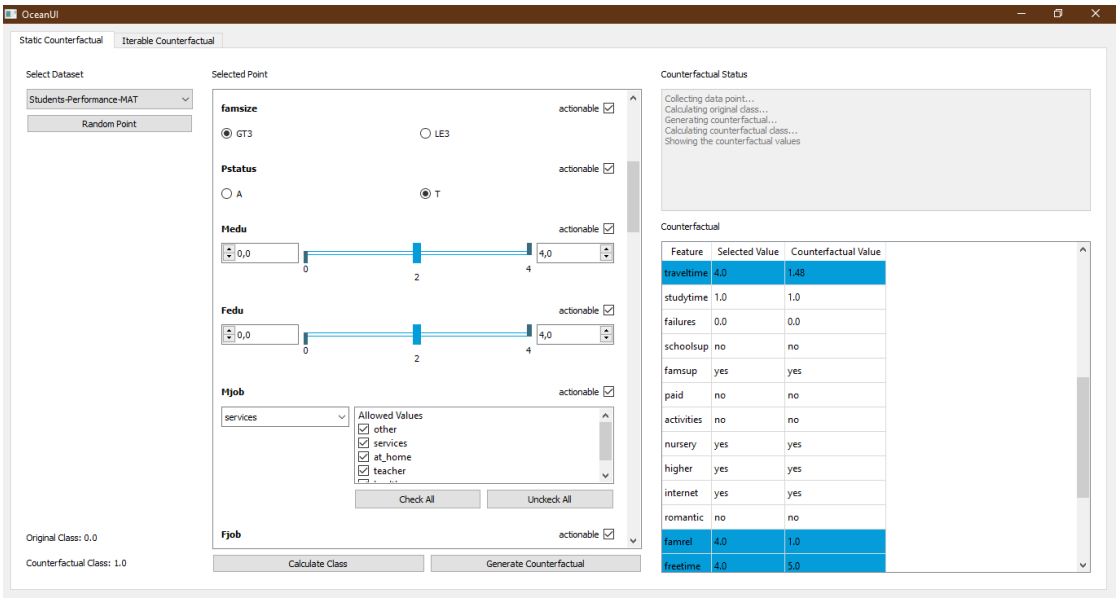

Figura 3.6: Interface de geração instantânea com resultado

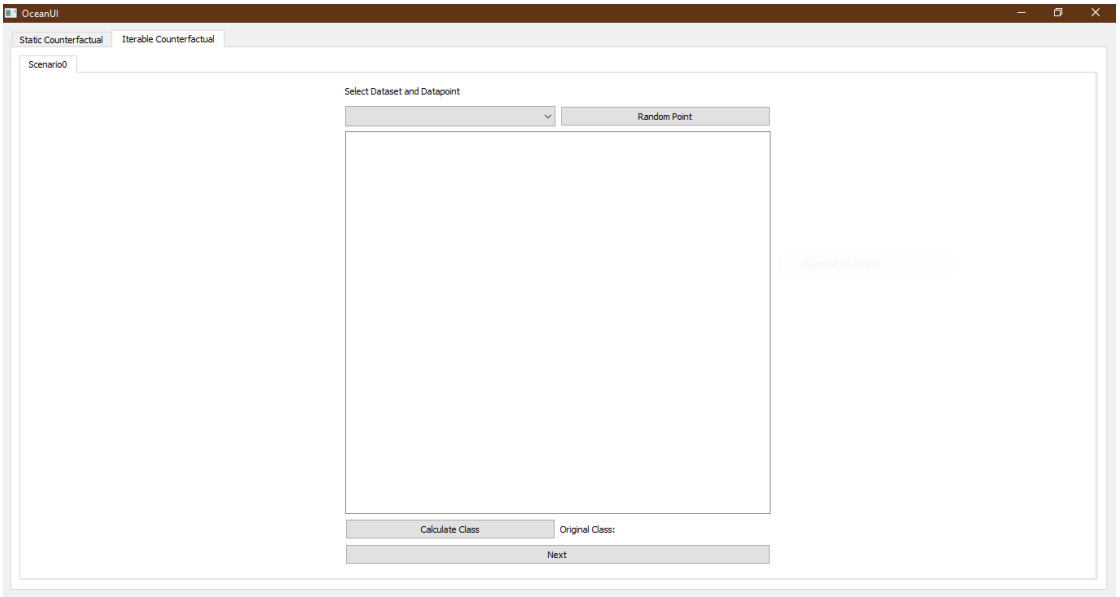

Figura 3.7: Interface de geração iterativa primeira versão em estado inicial

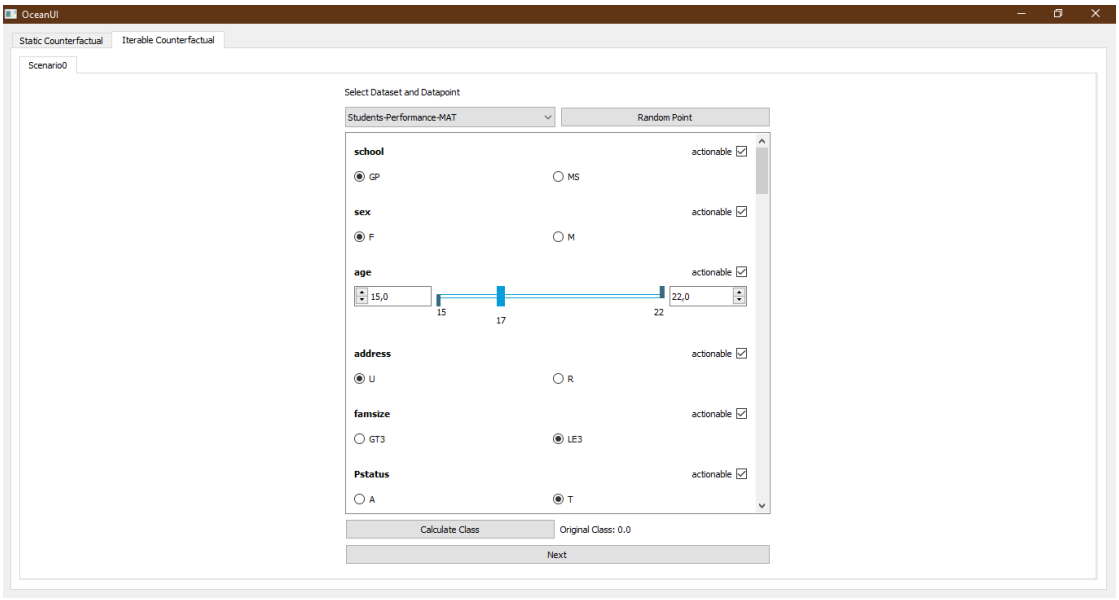

Figura 3.8: Interface de geração iterativa primeira versão ponto selecionado

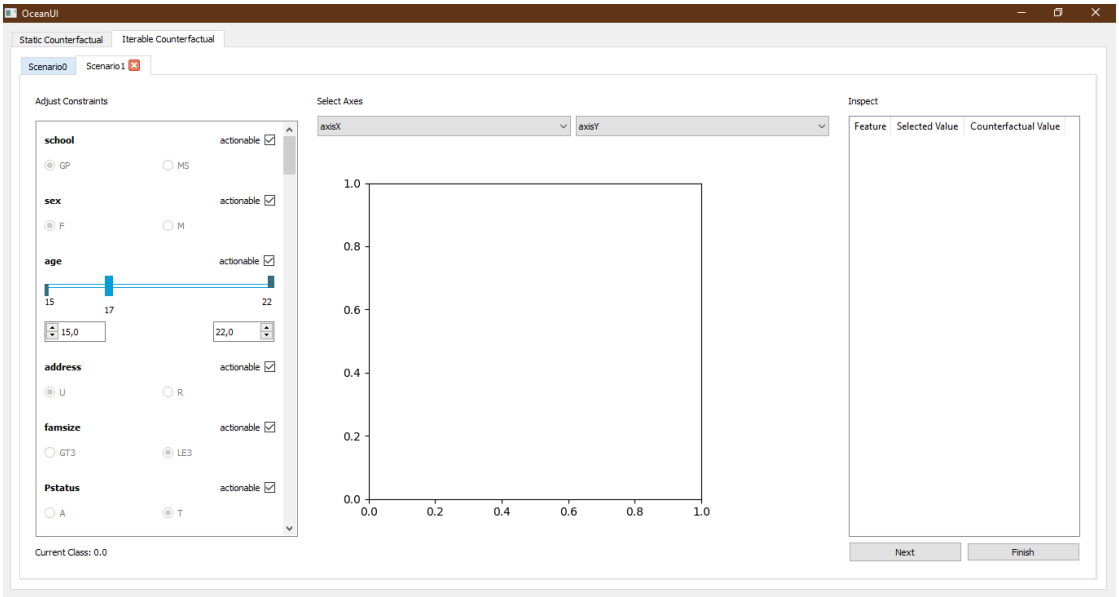

Figura 3.9: Interface de geração iterativa primeira versão cenário 1

por meio do tamanho do ponto plotado: pontos maiores estão mais distantes de um contrafactual do que pontos menores (Figura 3.10).

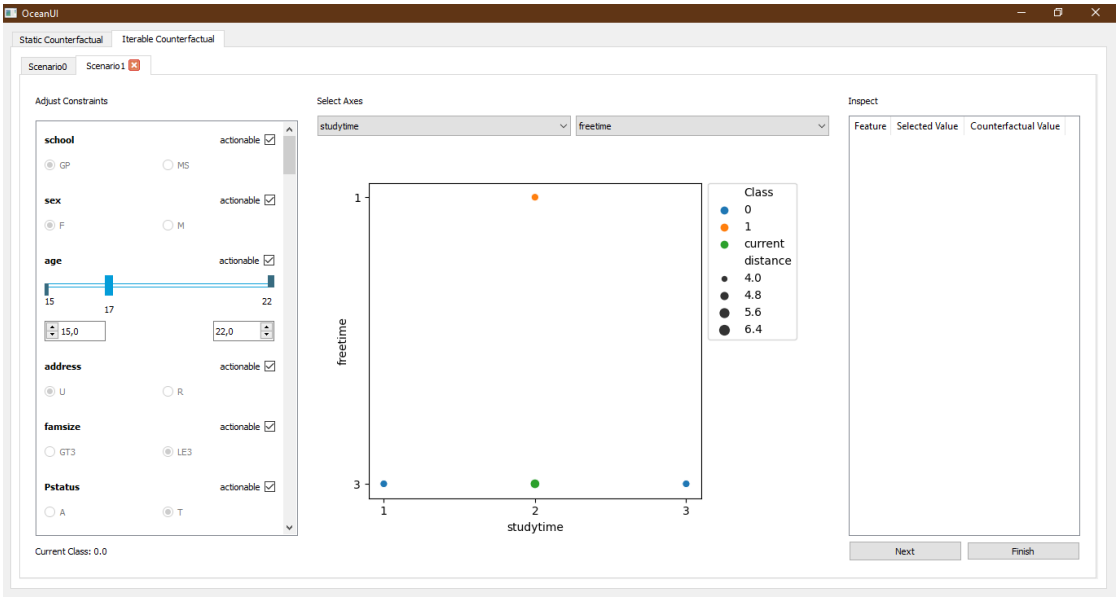

Figura 3.10: Interface de geração iterativa primeira versão gráfico do cenário

Além das informações que o gráfico apresenta, o usuário pode ver informações sobre o ponto fornecido e manipulá-lo para fazer a "caminhada" até um contrafactual. Para ver as informações sobre o ponto e quais mudanças ele já sofreu, basta clicar sobre ele e a tabela à direita é preenchida com os valores de entrada e os valores correntes. Para facilitar a comparação, todos os pares de valores de mesma *feature* diferentes entre si são mostrados com um fundo azul. E para executar o processo de "caminhada" basta clicar sobre o ponto corrente para selecioná-lo e então clicar em uma nova coordenada (x, y), para que o ponto corrente seja atualizado no próximo cenário (a tabela à direita já é atualizada no cenário atual para indicar a intenção de mudança do usuário e permitir a ele ver o impacto desta potencial mudança), Figura 3.11. Em seguida, o usuário pode instanciar um novo cenário usando a opção *"Next"* ou finalizar o estudo usando a opção *"Finish"*.

Uma vez que o usuário já terminou o estudo e deseja ver uma comparação final entre os pontos de entrada e suas alterações, ele pode usar a opção *"Finish"*. Esta opção instancia um cenário com uma tabela idêntica à utilizada nos cenários de gráfico, Figura 3.12.

3.1.3 Geração Iterativa Versão Final

Testes durante o desenvolvimento mostraram que o OCEAN não consegue encontrar contrafactuais quando se impõem muitas restrições simultane-

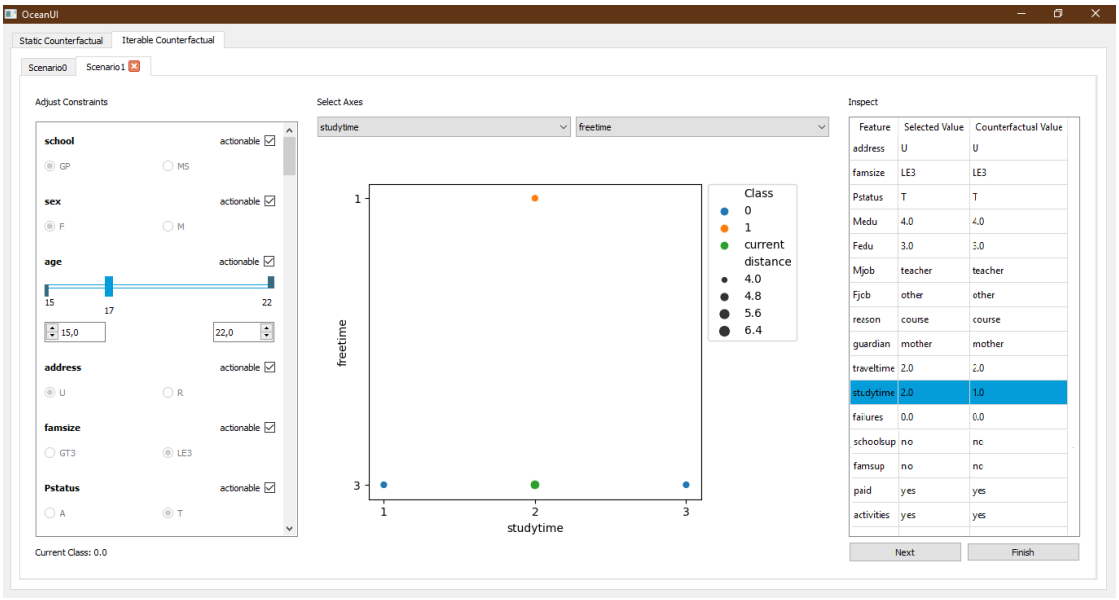

Figura 3.11: Interface de geração iterativa primeira versão gráfico clicado

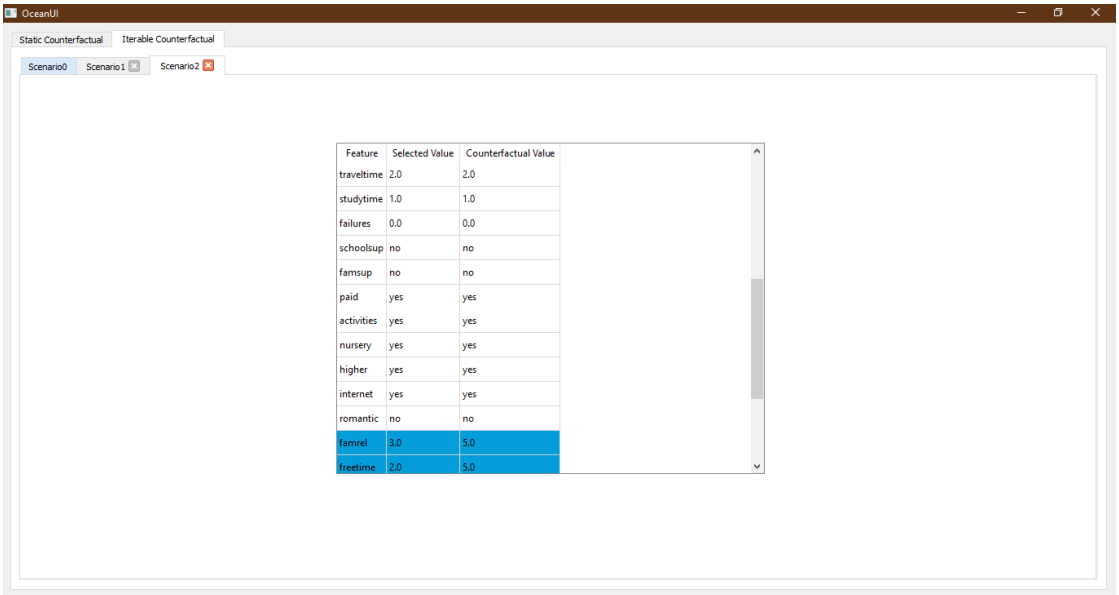

Figura 3.12: Interface de geração iterativa primeira versão cenário final

amente (o que de certa forma é esperado, pois o acumulo de restrições pode fazer o problema ser impossível de resolver usando os métodos implementados pelo OCEAN), portanto remover o *checkBox "actionable"* foi uma alternativa para desencorajar os usuários a realizar tal ação, pois agora seria necessário alterar as restrições apenas pelos ranges ou por meio de seleção de valores permitidos, devido a isto optamos por remover-lo dos componentes. Dessa forma, a personalização da explicação contrafactual fica sob a responsabilidade das restrições. No mais, os componentes permanecem inalterados.

Inicialmente, ao abrir o sistema na versão final de geração iterativa, o usuário se depara com a interface ilustrada na Figura 3.13. Similar à primeira versão iterativa, o "caminhamento" ocorre por cenários; o cenário de entrada é o *"Scenario0"*. À medida que o usuário vai utilizando as funcionalidade disponíveis, novos cenários vão sendo instanciados em abas.

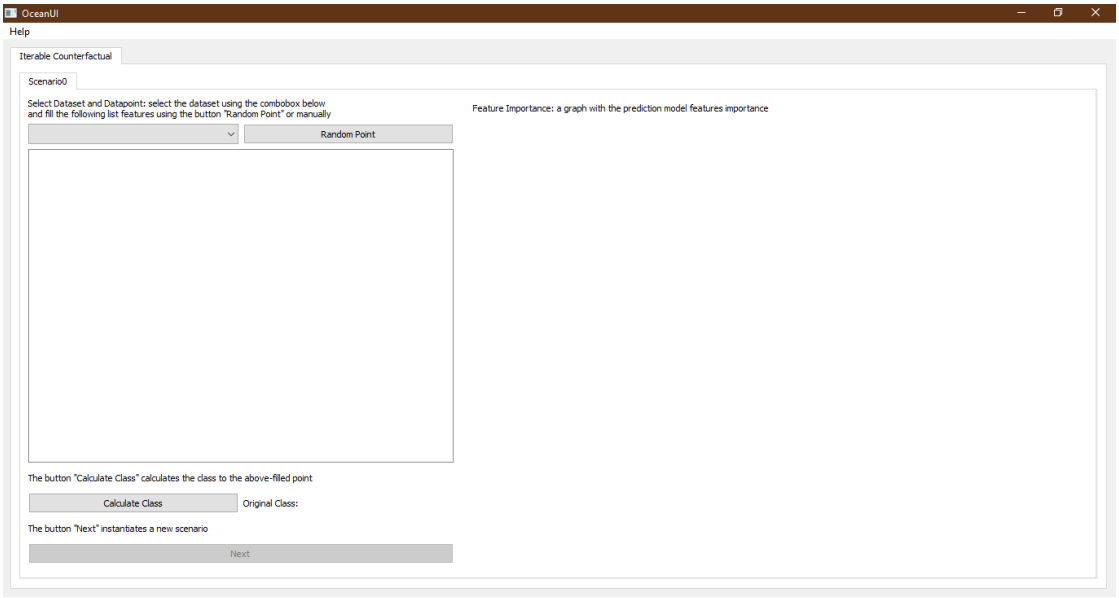

Figura 3.13: OceanUI em estado inicial

Nesta tela inicial o usuário seleciona um *dataset* para estudo e opta entre o preenchimento manual ou um ponto qualquer do *dataset* com classe 0, igual ao que está disponível nas versões anteriores de geração de contrafactual. Porém, diferentemente das versões anteriores, ao selecionar um *dataset* nesta versão o usuário recebe um gráfico, ao lado da lista de componentes de *features*, indicando a importância de cada *feature* para o modelo de predição (Figura 3.14). Com isto, o usuário já pode obter uma visão geral de como é o comportamento do modelo de predição, ou seja, se uma *feature* é mais importante ela tende a impactar mais a predição caso ela seja alterada.

Voltando às similaridades com a versão de geração instantânea, nesta versão também temos a opção de cálculo de classe (Figura 3.15, logo em

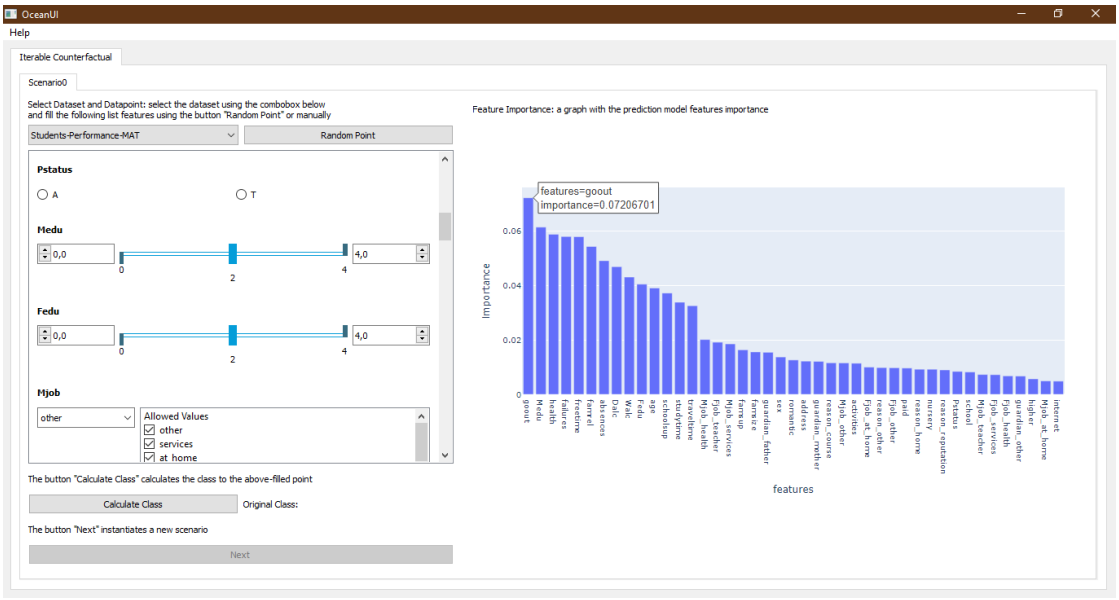

Figura 3.14: OceanUI dataset selecionado

seguida do preenchimento do ponto desejado. Nesta etapa de preenchimento do ponto de entrada, o usuário também pode informar suas restrições iniciais e tentar ir para o próximo cenário usando a opção *"Next"*. Caso não seja possível encontrar um contrafactual com as restrições indicadas (por exemplo, porque o usuário definiu um conjunto excessivamente restritivo), o sistema retorna esta informação ao usuário, vide Figura 3.16. E, caso haja algum contrafactual que atenda a todas as restrições, então o sistema instancia um novo cenário em uma nova aba (*"Scenário1"* vide Figura 3.17).

Neste novo cenário, temos a possibilidade de escolher quais *features* serão plotadas no gráfico e podemos também manipular o ponto corrente (segmentos de reta azul) via gráfico, bastando apenas clicar sobre o valor e arrastar até o valor desejado. Ao clicar sobre o valor corrente, nos é mostrado um componente de *feature* correspondendo à *feature* clicada e um gráfico de distribuição dos seus valores ao longo do *dataset*. Assim, à medida que o usuário manipular o ponto corrente, ele saberá se o valor modificado é ou não é um *outlier*. Além disso, é possível impor restrições por meio do componente mostrado, entretanto não é possível alterar o valor corrente por meio deste componente, vide Figura 3.18.

Como o contrafactual mostrado no gráfico é calculado no momento de instanciação do cenário, ele não é editado até que outro cenário seja instanciado. Por isso, o gráfico mantém os valores plotados inalterados, mesmo quando novas restrições tiverem sido aplicadas sobre o cenário corrente. Sendo assim, as restrições são refletidas no gráfico no cenário seguinte, desde que não seja contraditório a nenhum outro segmento de reta referente aos valores do

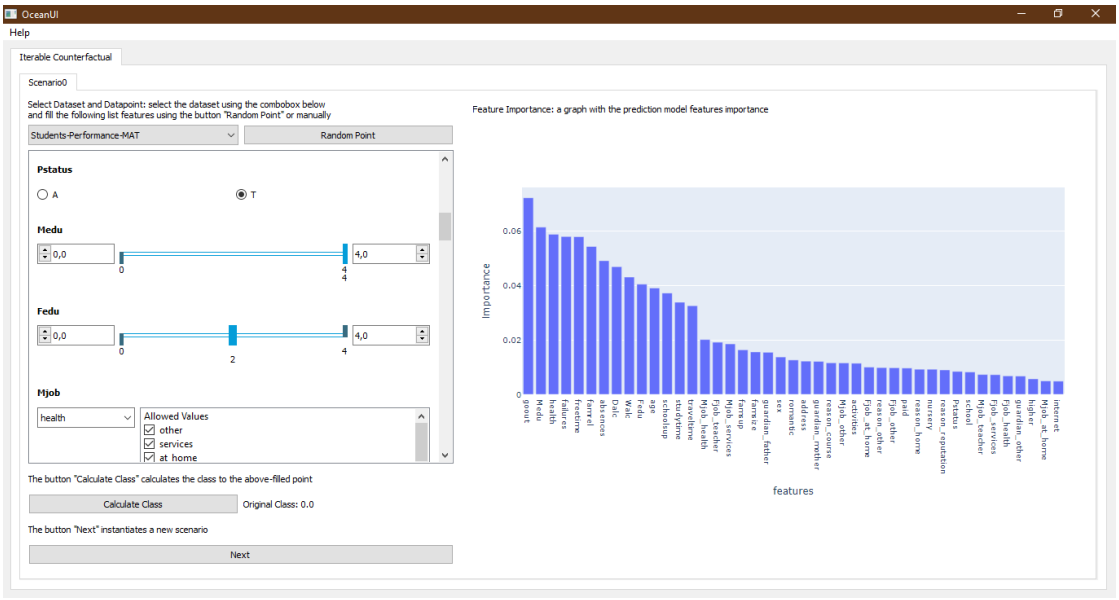

Figura 3.15: OceanUI ponto de entrada preenchido e classe calculada

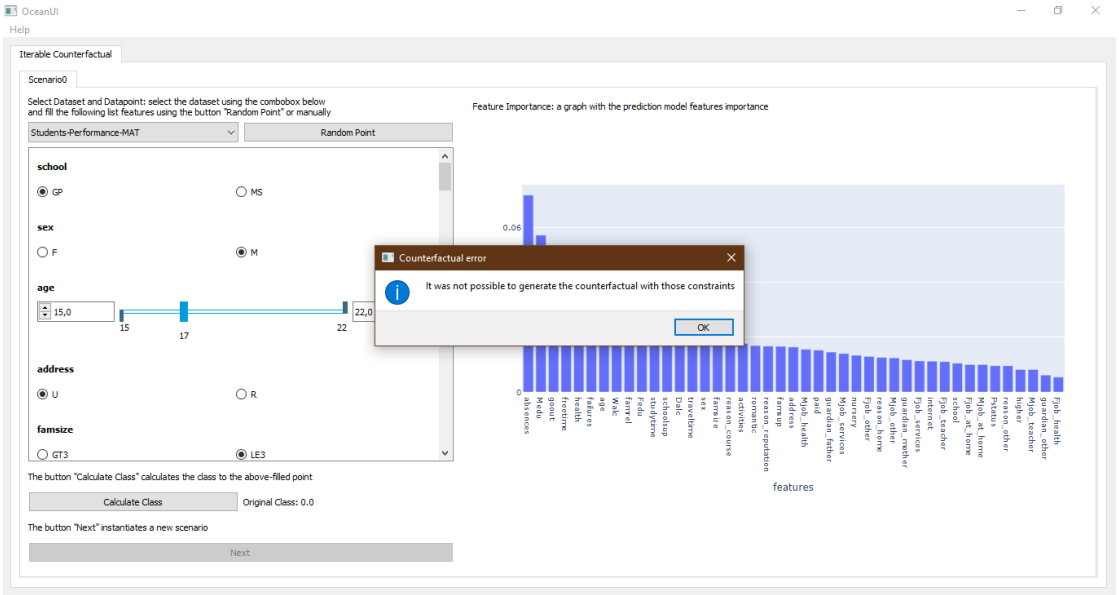

Figura 3.16: OceanUI erro por excesso de restrições

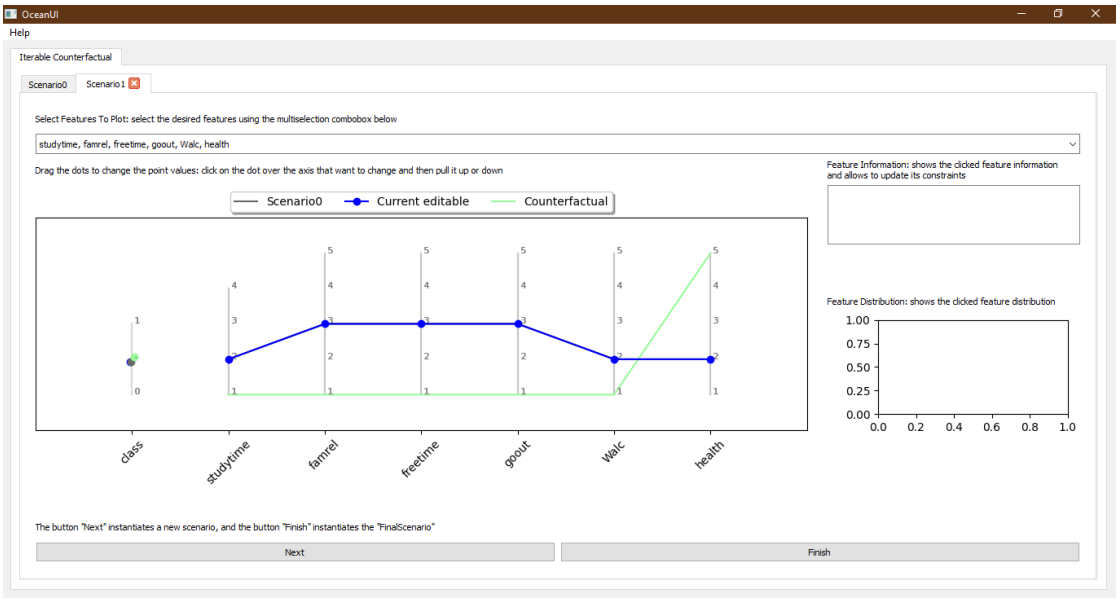

Figura 3.17: OceanUI novo cenário instanciado

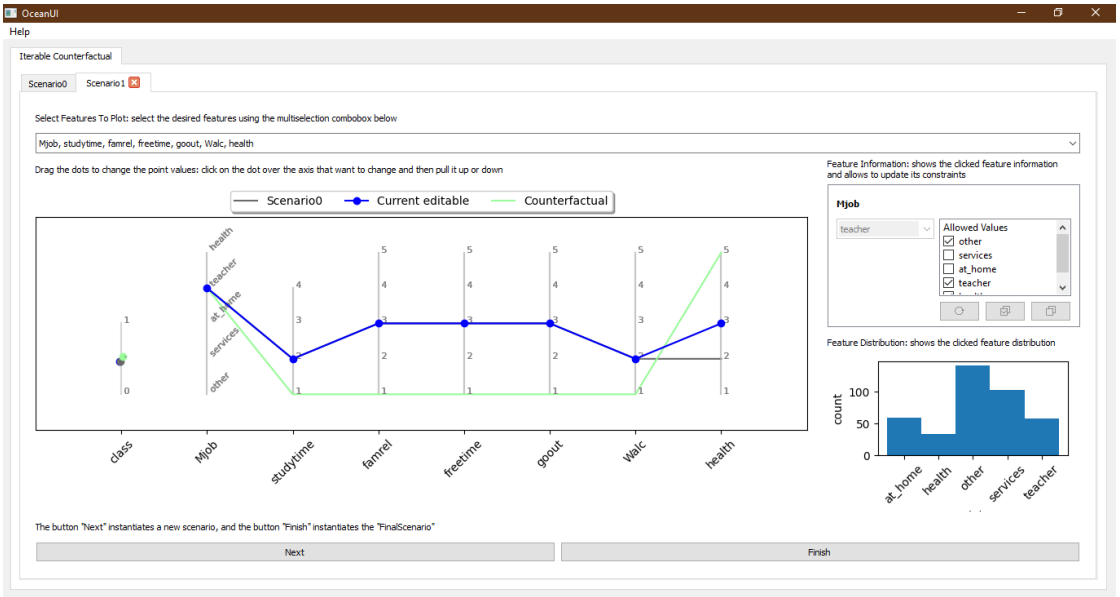

Figura 3.18: OceanUI cenário editado

cenário de entrada e nem do cenário anterior.

Por exemplo, se para uma determinada *feature* numérica o valor mínimo foi definido como sendo três, e temos o ponto do cenário inicial ou o ponto do cenário anterior passando sobre o valor dois, então o gráfico conterá o valor dois plotado no eixo desta *feature* para garantir a consistência dos pontos citados anteriormente. E para as *features* categóricas, todos os possíveis valores também são mantidos no gráfico, pelo mesmo motivo. Entretanto os componentes de *feature* continuam refletindo as mudanças impostas.

Para garantir que não haverá confusões sobre os valores permitidos, caso o usuário mova o valor para um não permitido pelas restrições, todo o eixo fica vermelho para indicar a inconsistência, vide Figura 3.19.

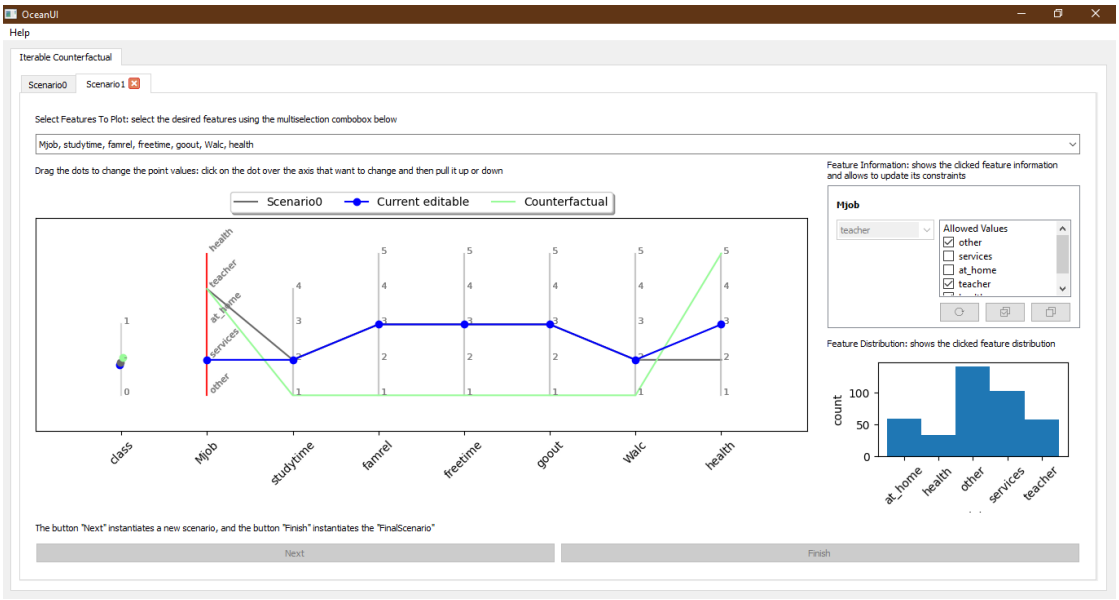

Figura 3.19: OceanUI restrição contraditória ao valor editado

Como citado acima, é possível instanciar novos cenários com a opção *"Next"*, permitindo assim manter um histórico de alterações até que se obtenha um contrafactual adequado. A proposta de manter um histórico pode ajudar a se ter um entendimento melhor do processo de geração de contrafactual e dos passos percorridos. Assim, caso seja necessário alterar algum passo, o usuário pode partir de cenário anterior ao cenário que houve um possível equivoco ou nova demanda de restrição e então continuar até um novo contrafactual. Ao manipular o ponto corrente até um contrafactual, o segmento de reta do ponto corrente muda da cor azul para a cor verde, para facilitar a identificação por parte do usuário, vide Figura 3.20.

Uma vez terminados os estudos, o usuário pode usar a opção *"Finish"* para instanciar o cenário final. Nesse cenário, temos o valor da classe do ponto de entrada e do contrafactual gerado pelo usuário e uma tabela listando as

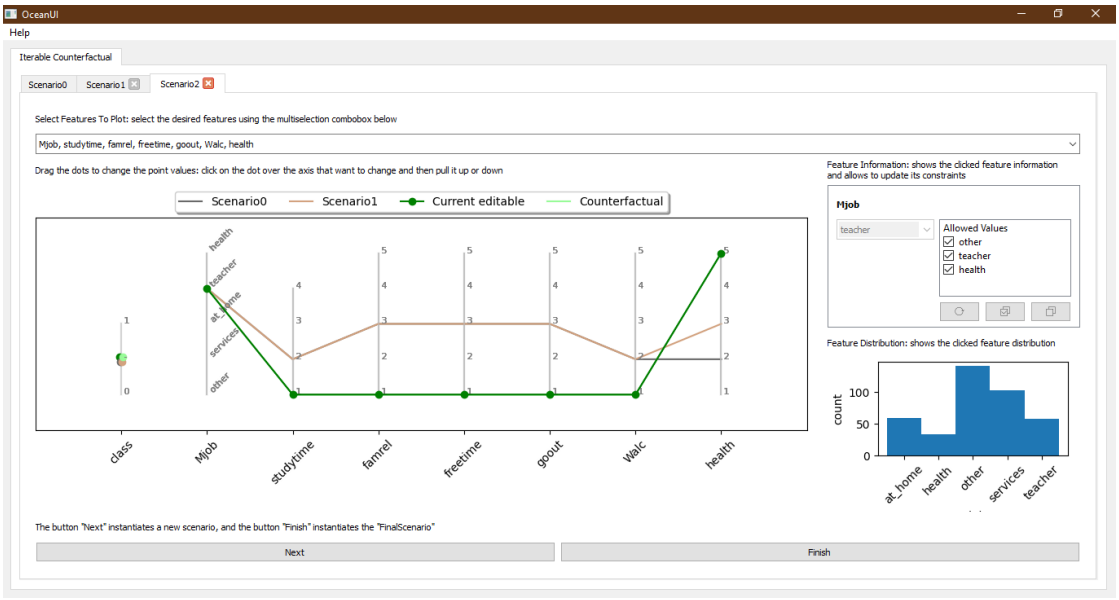

Figura 3.20: OceanUI cenário seguinte instanciado com contrafactual encontrado

features, os valores do ponto inicial e os valores do contrafactual gerado pelo usuário, juntamente com um destaque nos valores diferentes para facilitar a identificação dos valores que precisam ser alterados para se obter a classificação desejada, vide Figura 3.21.

Com isto terminamos este capítulo de concepção e desenvolvimento do OceanUI.

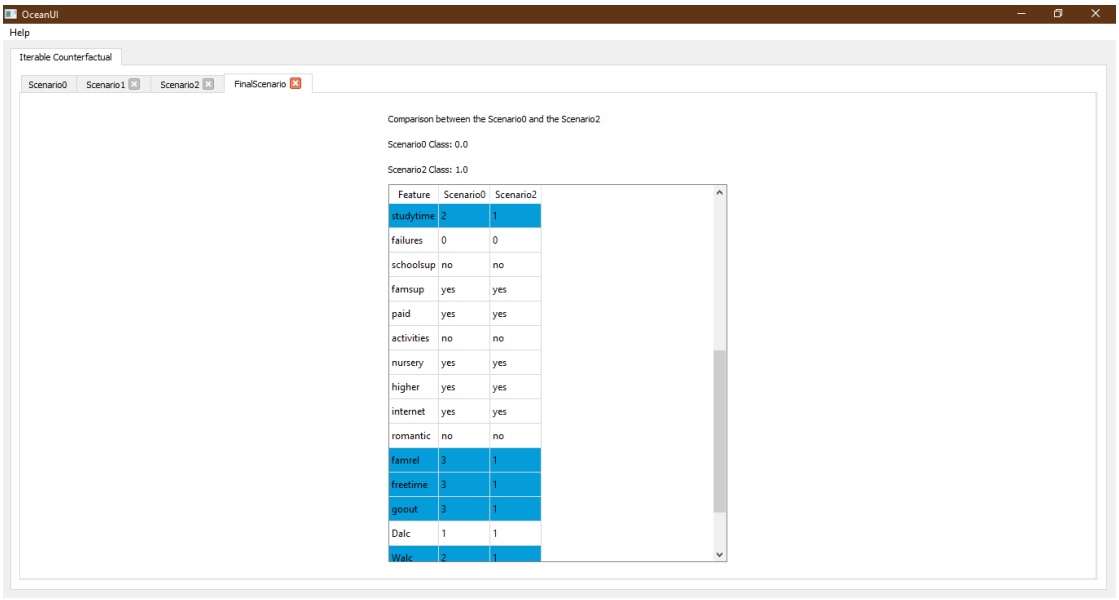

Figura 3.21: OceanUI cenário final para comparação

4
Avaliação

Este capítulo descreve como avaliamos nossa proposta, apresenta os resultados do estudo conduzido e discute suas implicações para ajustes e continuidade do projeto.

Para avaliar nossa proposta, conduzimos um teste de usabilidade juntamente com um de comparação com a interface VIDALab (Piesanen and Kerrigan, 2020), visto que dentre as referências principais do nosso trabalho, a interface do VIDALab é a que mais se assemelha à que propomos. O teste trouxe como resultado informações de como nosso sistema funciona na prática (ou o mais próximo possível disto, uma vez que o teste foi feito em um ambiente controlado) e alguns de seus pontos fortes e fracos. Com isso, nós podemos definir uma direção do que considerar para estudos posteriores e avanços no projeto.

4.1
Planejamento da Avaliação

A avaliação consiste em um único teste, onde avaliamos tanto a usabilidade do OceanUI quanto uma comparação entre ele e o sistema VIDALab, permitindo aos participantes voluntários do teste identificar os pontos fortes e fracos de cada sistema.

4.1.1
Objetivo

Os principais objetivos do teste foram os seguintes (os objetivos a seguir são referentes ao OceanUI):

- avaliar o quão fácil de usar e de entender é a forma de preenchimento dos dados e restrições
- avaliar o quanto a inclusão de restrições é percebida como útil pelos usuários
- avaliar se o "caminho" até encontrar um contrafactual é percebido como claro pelos usuários
- avaliar a opinião sobre o uso do gráfico para a geração do contrafactual

Esses aspectos nos auxiliam a avaliar a interface como um todo.

4.1.2 Procedimento

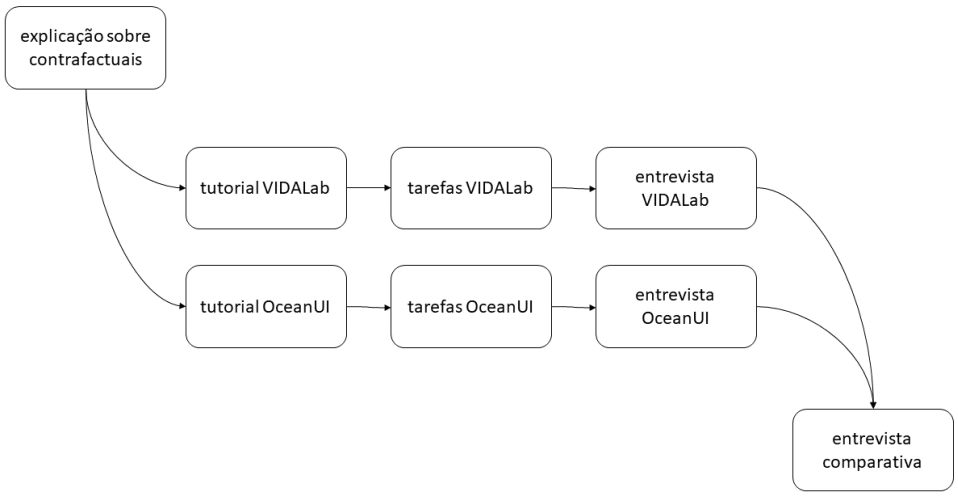

Figura 4.1: Diagrama dos passos do estudo

Para executarmos o teste, inicialmente passamos uma explicação rápida e exemplificada do que se trata Explicação Contrafactual para os participantes que não tinham este conhecimento prévio. Logo em seguida, os usuários (em sessões individuais) liam a documentação correspondente do sistema que iriam utilizar e executavam algumas tarefas predefinidas (as tarefas eram passadas verbalmente). Com isso, eles conseguiam navegar por todo o sistema e testar todas as funcionalidades disponíveis.

Após o teste de cada sistema, fizemos uma entrevista com o usuário para conseguirmos extrair informações sobre suas percepções e experiências durante o uso do sistema. E ao final do teste das duas interfaces, também fizemos outra rodada de entrevista, focando na comparação entre os sistemas.

Além disso, observamos durante toda a sessão o que os participantes falavam e o que eles faziam, para entendermos se poderia estar havendo alguma parte do teste ou dos sistemas que estivesse guiando os participantes para uma direção equivocada.

Vale observar que para reduzir vieses com relação à ordem de uso dos sistemas, nós conduzimos os testes de forma alternada, de modo que metade do grupo de teste executou as tarefas começando pelo OceanUI e logo em seguida pelo VIDALab e outra metade começou pelo VIDALab e depois utilizou o OceanUI.

Para simplificar o teste para os usuários, utilizamos um único dataset, o German-Credit[1] com *binning* por frequência nas *features "Age", "CreditAmount"* e *"Duration"*. Também fornecemos dois pontos predefinidos para dar início às tarefas. Nós havíamos investigado previamente tais pontos e sabíamos que eles apresentavam um comportamento adequado: para encontrar uma explicação contrafactual seria necessário que o usuário fizesse ajustes ou calibragem de peso durante a execução das tarefas.

4.1.3 Material

Para o OceanUI, preparamos um tutorial com as informações necessárias, tanto em texto quanto em imagens, para o usuário aprender como utilizar o sistema. E para o VIDALab utilizamos o tutorial fornecido pelos desenvolvedores. Vale comentar que ambos os materiais são semelhantes, pois exemplificam a utilização dos respectivos sistemas, e mostram como deve ser o fluxo de uso, entretanto a diferença entre eles está no fato do VIDALab usar Jupyter, desta forma alguns dos passos do tutotial deste sistema trata de células Jupyter e não apenas interface.

Para o teste, fizemos uma apresentação definindo do que se trata Explicação Contrafactual, fornecemos um documento com as informações sobre o *dataset* e também os pontos selecionados.

4.1.4 Execução

O teste foi pensado e preparado para ser feito de forma remota. Selecionamos uma máquina padrão (intel Core $i5$ e $8GB$ de memória RAM) e configuramos ambos os sistemas, de modo que os usuários não tiveram que lidar com as partes de clonagem de repositório GitHub e/ou com instalação de bibliotecas para fazê-los funcionar. Uma vez que o ambiente estava devidamente configurado, os usuários tiveram acesso à máquina usando software de controle remoto.

4.2 Resultados

Pelo fato da ideia de a explicação contrafactual normalmente estar associada a sistemas de predição e suas respostas, o perfil de usuários deveria estar relacionado a pessoas que normalmente utilizam sistemas de predição ou que pelo menos possuem alguma experiência com isto.

[1] `https://www.kaggle.com/uciml/german-credit`

Devido a isto, todos os participantes (dez no total) foram pessoas da área de Ciência da Computação, onde quatro eram alunos de graduação e seis eram alunos de mestrado.

Dada a escassez de tempo para uma seleção mais ampla de voluntários e de uma disponibilidade limitada de pessoas, nenhum dos participantes havia trabalhado com explicações contrafactuais anteriormente. Seis já possuíam experiência com Jupyter notebook e três não possuíam nenhuma experiência. Quatro não possuíam experiência com gráfico de coordenas paralelas, dois possuíam pouca experiência, um possuía experiência razoável e os três restantes possuíam muita experiência. E todos eles possuíam certa experiência com modelos de predição.

Além disto, os testes tiveram uma duração média de de uma hora e trinta minutos, cerca de quinze a vinte minutos acima do esperado e observado no teste piloto. Pois cinco dos participantes realizaram todas as atividades em cerca de sessenta e cinco minutos ou menos (dentro do esperado), dois precisaram de cerca de setenta e cinco minutos, enquanto os outros três precisaram de um intervalo de tempo entre oitenta e cinco a 100 minutos.

Para facilitar a apresentação dos resultados, vamos dividi-los em subseções: a primeira trata dos resultados relacionados aos objetivos principais; a segunda trata dos resultados comparativos entre o OceanUI e VIDALab; e, por fim, temos os resultados que não foram englobados nas seções anteriores mas ainda são dignos de observação.

4.2.1 Resultados Objetivos Principais

Considerando os quatro objetivos principais listados anteriormente, obtivemos os seguintes resultados (os objetivos a seguir são referentes ao OceanUI):

Em relação à forma de preenchimento dos dados e restrições, foi observado com os participantes P1, P2, P5, P6, P8 e P10, que é uma forma simples e fácil de aprender e permite facilmente ao usuário informar ao sistema as suas necessidades. Por exemplo, algumas das coisas ditas pelos participantes foram as seguintes:

- "as restrições pelas janelas de gráfico é algo bem intuitivo, e as variações de componentes (por tipo de *feature*) também ajuda, pois deixa simples a manipulação" (P1)
- "a tela de entradas é bem interessante, pois é simples de entender e usar" (P2)
- "a manipulação é simples, tanto edição do ponto corrente, quanto inclusão de restrição" (P5)

– “é bastante interativo, bem intuitivo e a forma de incluir as restrições é bem tranquila” (P6)

– “a forma de preenchimento é fácil, pois posso arrastar os valores dentro de um range dado e também marcar ou desmarcar valores das listas, e as restrições são bem tranquilas de incluir e funcionam como filtros” (P8)

– “é flexível para incluir restrições, pois basta clicar no ponto e então informar o que for necessário” (P10)

Porém foi observado com P3, P7, P8 e P9 que, em um primeiro momento, até entender o funcionamento dos componentes, de entrada de valores e restrições, é um pouco confuso utilizar o componente de *features* numéricas e a inclusão de restrições as vezes é ambígua. Por exemplo, algumas das coisas ditas pelos participantes foram as seguintes:

– “nem todas as restrições ficam claras, pois nem sempre aparece uma indicação que foram aceitas e isso dificulta o entendimento” (P3)

– “inicialmente o componente para *features* discretas é confuso, pois precisa arrastar o valor até a posição final (ele se move por valores inteiros em vez de decimais)” (P7)

– “ao acrescentar restrições sobre as *features* na janela de gráfico, não é claro se a restrição foi ou não considerada” (P8)

– “o componente de *features* numéricas é ambíguo, pois apresenta tanto um *slider* quanto um *doubleSpinBox* para movimentar os limites superior e inferior” (P9)

Em relação à percepção da utilidade das restrições, foi observado com os participantes P1, P2, P4, P6 e P10 que é útil para auxiliar na geração de contrafactual, pois evita que valores desnecessários sejam considerados e isso acelera o processo de geração. Por exemplo, algumas das coisas ditas pelos participantes foram as seguintes:

– “é uma forma de se chegar mais rápido na resposta, porque evita opções de valores desnecessários” (P1)

– “quando só queremos soluções viáveis podemos restringir e eliminar valores desnecessários” (P2)

– “a pessoa tem mais poder de manipulação, não fica presa apenas ao que o sistema oferece, ela pode mudar e guiar melhor o processo” (P4)

– “foi bem intuitivo incluir restrições no OceanUI, principalmente à medida que as regras de negócio se modificavam” (P6)

– “isso ajuda a afunilar o processo, se você tiver muitos valores e precisar remover alguns, basta informar isso” (P10)

Além disto, não foi observado nenhum comentário negativo em relação à funcionalidade de restrições. Os participantes foram unânimes em relação ao acréscimo de poder e benefício que esta funcionalidade garante ao percurso até o contrafactual, de modo que as críticas foram mais direcionadas aos componentes de inclusão de restrição e ao gráfico.

Em relação à percepção da clareza do percurso até encontrar um contrafactual, foi observado com os participantes P1, P2, P3, P6 e P9 que tende a ser longo, uma vez que podemos instanciar vários cenários com pequenas modificações em cada um deles. No entanto, de uma forma geral ainda é claro, devido ao histórico dos cenários anteriores e as informações dispostas na interface.

– “a forma de manipulação pelo gráfico juntamente com a sugestão deixam o processo de encontrar um contrafactual mais fácil e claro” (P1)
– “a geração de cenários é intuitiva e o histórico dessas gerações permite uma boa comparação” (P2)
– “a usabilidade é clara e o uso de gráfico interativo facilita o processo de geração de contrafactual” (P3)
– “a navegação em abas ajuda a entender o fluxo” (P6)
– “as explicações ao longo da interface ajudam a lembrar as funcionalidade e facilitam o fluxo, embora o fluxo seja mais longo” (P9)

Porém foi observado com os participantes P3, P4 e P6, que algumas interações não estavam claras o suficiente, pois em alguns momentos os usuários entenderam que era mais fácil e intuitivo voltar ao *“Scenario0”* para fazer as aplicações de restrição em vez de usar os cenários com gráfico interativo para tal propósito. E ao final do estudo, quando já havia chegado no contrafactual, alguns participantes esperavam que o botão *“Next”* deveria ser usado para exibir a aba de comparação entre o *“Scenário0”* e o cenário onde o contrafactual foi encontrado.

– “o tutorial deveria mencionar sobre a organização em abas e que a instanciação dos cenários que cria elas da esquerda para a direita, além disto a ideia do botão *"Finish"* não ficou bem definida, pois esta opção está habilitada em todos os cenários e não apenas no cenário cuja classe mudou para a classe desejada” (P3)
– “pareceu mais correto voltar ao *"Scenario0"* para incluir as restrições, em vez de usar o gráfico, e então gerar novos cenários a partir dele” (P4)

– “em vez de aplicar as restrições usando o gráfico, foi mais intuitivo voltar no *"Scenario0"* todas as vezes para aplicar as restrições e então instanciar os novos cenários. E para apresentar o comparativo final, ter que clicar em *"Finish"* ao invés de *"Next"* dificultou bastante, já que é mais natural continuar o processo com *"Next"*, pois *"Finish"* da uma ideia de finalizar o sistema como um todo” (P6)

Em relação à opinião sobre o uso do gráfico para a geração do contrafactual, foi observado com os participantes P2, P3, P4, P5, P6, P7, P8 e P9 que ele é bem instrutivo e muito útil, pois as informações que ele apresenta, juntamente com a possibilidade de acompanhar em tempo real o impacto das mudanças, além de fornecer informações relevantes, facilita e agiliza o processo de geração de contrafactual.

– “o gráfico facilita a manipulação e análise dinâmicas, pois é possível acompanhar em "tempo real" o quanto as mudanças afetam a predição” (P2)
– “o gráfico facilitou pois mostra o quão longe o usuário está da sugestão e isso ajuda a direcionar o que deve/pode ser alterado e à medida que se altera o ponto corrente, já é possível ver como a classificação está se comportando, dessa forma, rapidamente se sabe o efeito da mudança” (P3)
– “a linha do contrafactual de sugestão no gráfico é bem útil, pois já dá uma ideia de onde eu devo/posso mover os valores de modo a chegar onde quero” (P4)
– “a manipulação por meio do gráfico é mais agradável do que se fosse mexer diretamente com os números, essa interação é legal e bem amigável” (P5)
– “o gráfico ajudou tornando o programa mais interativo e facilitando a passagem de valores” (P6)
– “o gráfico é útil, pois a manipulação do ponto corrente é clara, além de já apresentar tudo o que está sendo alterado” (P7)
– “o gráfico é importante, pois facilita a utilização sem precisar montar várias simulações” (P8)
– “a manipulação do ponto é simples, uma vez que só precisa arrastar” (P9)

Porém foi observado com os participantes P1, P2, P3, P5 e P8 que em um primeiro momento o gráfico é complexo, pois gráficos de coordenadas paralelas

não são comumente utilizados (o que acaba gerando essa estranheza inicial). Além disto, é por meio de cliques sobre o gráfico que o usuário tem acesso aos componentes de *feature* para aplicar as restrições, mas como o gráfico exige que se clique exatamente sobre um ponto, isso gera um certo desconforto, pois esperavam que qualquer clique no eixo, no ponto, ou no nome da *feature* já acionassem estes componentes. Também foi observado que o eixo de informação da classe dos pontos é confuso, pois não informa explicitamente o limiar de mudança de classe e é disposto no inicio, sendo que seria mais agradável visualmente se ele estivesse ao final do gráfico.

- "depois que entendi o gráfico, tudo ficou mais claro, porém o eixo da classe ainda é confuso, talvez se ele fosse maior ou se tivesse o valor 0.5 isto poderia facilitar a visualização" (P1)
- "o gráfico é complexo em um primeiro momento mas depois de entender como ele funciona, fica muito fácil manipular o ponto e testar o impacto das mudanças em "tempo real" . Entretanto o eixo de classe não ficou claro, pois a ausência de valores intermediários dificulta o entendimento" (P2)
- "é estranho ter que clicar exatamente no ponto para exibir o componente da *feature* para restrição, além disto nem todas as restrições ficam claras, no gráfico, que foram aceitas e isto dificulta o entendimento. Em relação ao eixo da classe, não é tão claro que acima de 0.5 já era classe 1, e também, o eixo de classe deveria ser o último, uma vez que é o resultado da predição (deixá-lo no início é visualmente estranho)" (P3)
- "o gráfico não necessariamente atualiza à medida que acrescenta as restrições, isso é um pouco confuso" (P5)
- "ter que clicar exatamente no ponto para aparecer a janela de restrições não é tão intuitivo, pois inicialmente cliquei no eixo, depois no nome da *feature*, isso me deixou confuso. Além disto, não ficou claro, inicialmente, que o contrafactual de sugestão poderia ser um guia para direcionar a geração do contrafactual personalizado" (P8)

4.2.2 Resultados Comparativos

Considerando a comparação com o VIDALab, obtivemos os seguintes resultados.

Em relação ao sistema mais fácil de usar, foi observado que:

– seis participantes consideram o OceanUI mais fácil, pois é uma interface mais direta, intuitiva e autocontida (não precisa de programas de terceiros, como o Jupyter), além de possuir exibição e manipulação por meio de gráfico (o que facilita o entendimento e é um diferencial). Também consideram a forma de indicar a entrada e as restrições como um ponto positivo se comparado ao VIDALab pois as restrições são explicitas em vez de pesos. E consideram a usabilidade mais clara, pois os comentários ao longo da interface ajudam a entender o fluxo

– um participante considera o VIDALab mais fácil, pois possui uma interface minimalista. Também consideram a forma de exibição do contrafactual mais clara, mesmo tendo que ficar mudando de células do Jupyter para acessá-la.

– um participante considera que a forma de entrada e restrições do OceanUI são mais fáceis e explicitas, enquanto a saída do VIDALab é mais fácil e simples

– dois participantes consideram o VIDALab mais fácil, pois possui uma usabilidade mais simples e limpa, enquanto a manipulação sobre o gráfico do OceanUI se sobressai

Em relação ao sistema mais útil, foi observado que:

– sete participantes consideram o OceanUI mais útil, pois possui e apresenta mais informações, seja com histórico, acompanhamento da classe à medida que se muda o ponto, exibição de frequência de valores das features ou importância das mesmas. Além disto, a manipulação por meio do gráfico e a inclusão explicita de restrições tornam a análise e o processo de geração de contrafactual mais eficiente. E consideram que por apresentar uma interface amigável, isto o torna mais acessível para um publico maior, aumentando sua utilidade

– um participante considera o VIDALab mais útil, pois permite uma integração mais fácil e direta com outros sistemas Python

– um participante considera ambos os sistemas úteis, porém para situações diferentes, por exemplo em casos em que o usuário precise de algo mais visual então usaria o OceanUI, entretanto se for necessário apenas gerar algumas possibilidades de contrafactual, então usaria o VIDALab

– um participante não soube opinar

De forma geral, dentre os participantes que consideram o OceanUI mais fácil e útil, se comparado ao VIDALab, comentaram que se houvesse uma

demanda por geração de contrafactual, poderiam utilizá-lo no dia-a-dia devido a sua facilidade de manipulação pelo gráfico, acompanhamento de impacto em tempo real na predição, inclusão de restrições, as informações que ele exibe e o histórico de cenários que ele possui. Outro ponto positivo para o OceanUI é que o gráfico sugere apenas o conjunto de *features* sugeridas para serem alteradas, o que também ajuda a direcionar a geração de contrafactual.

Quanto aos usuários que preferiram o VIDALab, eles comentaram que, se houvesse uma demanda por geração de contrafactual, poderiam utilizá-lo devido à sua flexibilidade dos pesos, pois isto garante um poder maior a nível de *feature*. Além disso, permite gerar várias explicações simultâneas, aumentando assim a faixa de análise e por consequência ajudando num entendimento maior dos dados estudados. Também mencionaram sua aparente facilidade de integração com demais sistemas Python, o que facilita a implementação de tarefas automatizadas.

4.2.3 Resultados Gerais

Como observações gerais sobre o OceanUI que não foram listadas acima, pois não se encaixam adequadamente nos tópicos citados anteriormente, obtivemos os seguintes resultados:

- a possibilidade de selecionar as *features* que podem ser alteradas iria resumir o número de restrições que precisamos acrescentar no OceanUI, similar ao que é feito no VIDALab
- para alguns usuários, não pareceu claro o suficiente o que os pontos de cenários anteriores representavam no gráfico, pois foi sugerido que se juntasse os cenários para comparação do percurso em direção ao contrafactual
- foi ressaltado por mais de um participante que a exibição exata do valor de porcentagem é um recurso interessante de se visualizar no gráfico, considerando que isso pode ajudar a ter uma visão mais clara de como o ponto corrente está se aproximando de um contrafactual
- não está claro para todos os usuários que quando o ponto corrente muda de cor é quando se muda a classe
- para alguns usuários o contrafactual de sugestão não está claro, alguns acham que devem mudar o contrafactual em vez de mudar o ponto corrente rumo ao contrafactual

- não é claro quando se recalcula classe na aba do *"Scenario0"*, pois não há nenhuma mensagem informando se deu certo ou errado, ou se houve alguma mudança
- poderia incluir mensagens claras para quando o usuário muda o ponto corrente e chega em um contrafactual, pois alguns usuários continuam alterando o ponto sem perceber que já possuem a resposta
- poderia ter a possibilidade de fechar o componente de inclusão de restrição, pois só é possível trocar o componente corrente, mas em momento algum é possível ocultá-lo
- apenas um dos usuários utilizaram ativamente o gráfico de importância de *features*
- apenas dois dos usuários utilizaram ativamente o gráfico de distribuição dos dados de *feature* no *dataset*
- comentaram sobre a importância de ter telas mais escuras para aumentar o conforto de pessoas com problemas de visão, além de possibilidades de zoom

Vale comentar que algumas das observações listadas acima foram feitas nos momentos iniciais de utilização do sistema e, à medida que se executava as tarefas dadas e se obtinha um melhor entendimento da interface e das suas funcionalidades, algumas dessas dúvidas eram sanadas de forma natural.

4.2.4 Discussão

Como fizemos uma separação em seções para apresentar os resultados obtidos, também faremos a mesma separação para deixar melhor apresentável a nossa discussão.

4.2.4.1 Discussão sobre os Objetivos Principais

Dados os resultados que tivemos, percebemos que em relação a forma de preenchimento dos dados e restrições, foi claro e simples para a maioria dos usuários porém pode ser ajustado. Por exemplo, simplificar o componente para *features* discretas/numéricas removendo os *sliders* de mínimo e máximo, pois esta parte não está sendo intuitiva para os usuários, uma vez que eles apenas usam os *spinBoxes* para alterar os limites. Além disto, devemos melhorar o "arrastar" do *slider*, atualmente quando usamos o *slider* para valores inteiros, ele só move o ponto quando "caminha" pelo menos uma unidade inteira, logo,

uma opção de melhoria pode ser permitir o “arrastar” em decimais e quando o usuário soltar, o *slider* vai para o inteiro mais próximo.

Em relação à percepção da utilidade das restrições, foi notado que esta funcionalidade é útil e importante, pois além de acelerar o processo de geração de contrafactual, ela também garante mais poder na mão dos usuários, de modo que obtenham contrafactuais mais aplicáveis a cada individuo. Mas como parte de trabalhos futuros podemos estudar a possibilidade de selecionar um grupo de teste onde os usuários já conheçam sobre explicações contrafactuais e também possuem alguma experiência com sistemas de geração dessas explicações, com isso poderemos ver uma utilização mais refinada das restrições para então termos um entendimento melhor e mais profundo sobre o impacto dela no processo de geração de contrafactuais.

Em relação à percepção da clareza do “caminho” até encontrar um contrafactual, chegamos a conclusão de que é claro porém pode ser ramificado. Vimos isto acontecendo com os usuários que acrescentaram as restrições por meio dos cenários com gráfico e os que voltaram ao *"Scenário0"* para incluir as restrições, uma vez que ambos são válidos. Entretanto ao voltar ao *"Scenário0"* perde-se parte do histórico que esperava-se manter, pois as modificações sobre o ponto corrente só ocorrem ao final do processo.

Devido a isto, podemos estudar novas formas de definição do fluxo, por exemplo, se é válido deixar livre a possibilidade de abertura de novos cenários a partir de qualquer cenário já instanciado (como temos atualmente), ou se devemos eliminar a possibilidade de interação sobre um cenário assim que a opção *"Next"* for usada. Desta forma as possíveis ramificações seriam evitadas e o fluxo seria totalmente linear.

Em relação à opinião sobre o uso do gráfico para a geração do contrafactual, ficou claro que ele é útil, instrutivo, facilita e agiliza o processo de geração de contrafactuais, além de ter o facilitador de não precisar gerar várias simulações para saber o quão próximo está de um contrafactual, pois esta análise é feita em "tempo real". Entretanto uma possibilidade de melhoria é exibir com cores diferentes os valores de categóricos que não são permitidos para um determinado cenário, para evitar qualquer confusão devido a exibição desses valores.

Ainda relacionado com o ponto anterior, foi sugerido simplificar o gráfico deixando apenas o ponto corrente e o contrafactual sugerido. Os usuários alegaram que como já tem as abas com os cenários, não é necessário que haja essa redundância de informação, entretanto isso deve ser analisado com mais cautela, uma vez que é possível instanciar novos cenários a partir de qualquer cenário já instanciado. Se uma mudança nesse nível for feita sem um estudo

prévio, isso pode gerar confusão em alguns usuários pois podem se perder no fluxo de geração de contrafactuais.

Além disto, sobre o contrafactual sugerido, foi observado que não ficou tão claro para alguns usuários que se trata apenas de uma sugestão e que não necessariamente precisa-se parear os pontos corrente e contrafactual, pois em alguns exemplos o Ocean está gerando contrafactuais com mais sugestões de mudança de *features* do que realmente está sendo necessário na prática. Tal fenômeno pode ser devido a redução do número de árvores e de nós das árvores para o modelo de predição, uma vez que isso deixaria os testes muito lentos. Portanto, testes futuros devem considerar usar um número maior de árvores e cada uma delas com mais nós, para avaliar se desta forma a precisão do contrafactual sugerido também aumenta.

4.2.4.2 Discussão sobre os Dados Comparativos

Em relação aos dados comparativos comentados pelos voluntários, algo que deve ser estudado é a possibilidade de simplificação das telas de cenário 1 à n, pois alguns usuários comentaram que ela é informativa e bem completa, porém apresenta muita informação e esse excesso de informação acaba confundindo em um primeiro momento. Uma alternativa para simplificar seria mover as partes de *"Feature information"* e *"Distribution"* para uma janela popup, de modo que ela aparecesse apenas quando fosse clicado em um eixo para editar ou inclusão de restrições. Outra alternativa seria remover completamente a parte de *"Distribution"*, uma vez que está sendo pouco utilizada, apenas dois dos usuários viram utilidade nessa funcionalidade. Tais observações se devem ao fato da interface do VIDALab ser minimalista.

Em contra posição aos voluntários que sentiram que a interface do OceanUI é excessivamente informativa, tivemos alguns que sugeriram acrescentar um modal de informação nos componentes para auxiliar a utilização. A ideia é que esses modais sejam um lembrete breve de como cada um dos componentes de *feature* funcionam. Pois até se obter uma certa experiência de uso, estes lembretes evitariam ter de voltar no tutorial para tirar eventuais dúvidas.

4.2.4.3 Discussão sobre os Dados Gerais

Em relação as dados gerais, podemos estudar para versões futuras a possibilidade de configuração de metadados (informações gerais sobre as *features* e seus valores), pois nem sempre as *features* e os valores apresentados no gráfico são claros e às vezes representam intervalos ou valores categóricos

que não possuem significado semântico direto com a realidade. Logo, a inclusão destes metadados pode auxiliar o estudo, reduzir a necessidade de consulta em materiais complementares e enriquecer o gráfico e os componentes de restrições.

Outra observação pertinente é a necessidade de se incluir um botão para "fechar" o componente que exibe *"Feature information"* e/ou quando clicasse em uma área vazia do gráfico ele também fosse fechado.

5
Considerações finais

A principal contribuição deste trabalho está no fato de facilitar a geração de contrafactuais até mesmo por pessoas que não são da área de Informática e que não estão familiarizadas com programação de uma forma geral. Com a versão de geração instantânea, o usuário apenas precisa selecionar um *dataset*, entrar com seus dados e pressionar um botão para obter a informação do que precisaria ser alterado para ele alcançar seu objetivo. Já com a versão de geração iterativa, além de selecionar o *dataset* e entrar com seus dados, o usuário pode, de forma iterativa e embasada nos gráficos fornecidos, acompanhar e guiar a geração do contrafactual por manipulação direta dos valores e/ou inclusão de restrições.

Para avaliar essa interface, fornecemos o sistema juntamente com cenários de uso para um grupo de voluntários utilizar, avaliar e dar *feedback* de melhorias. Além disso, fazemos uma avaliação comparativa com o sistema desenvolvido por Piesanen and Kerrigan (2020), escolhido devido à similaridade de sua interface com o nosso projeto. A ideia dos cenários foi fornecer um guia de utilização para entender, por meio da avaliação e do feedback dos usuários, o quão fácil de entender, fácil de usar, informativo e perceptivelmente útil é o nosso sistema. Com o estudo realizado, concluímos que desenvolvemos componentes, de entrada de dados e restrições, fáceis de entender e de usar; vimos que a possibilidade de impor restrições é útil para acelerar e refinar o processo de geração de contrafactuais; observamos que, ao utilizar cenários, deixamos o fluxo mais organizado e geramos um histórico de mudanças; e, por fim, vimos que o gráfico de coordenadas paralelas é um facilitador do percurso, pois simplifica a manipulação dos pontos e deixa claro o que pode ou deve ser alterado para se obter o contrafactual.

Como trabalhos futuros, esperamos aperfeiçoar o OceanUI levando em conta as observações dos usuários durante o teste e também pretendemos aplicar novas rodadas de teste variando o grupo de voluntários para incluir tanto pessoas mais experientes com a utilização de contrafactuais, quanto pessoas tangentes à área de computação que não necessariamente tenham conhecimentos profundos sobre o tema. Dessa forma, podemos obter informações e sugestões de melhorias relacionadas ao processo de geração de contrafactuais

e também sobre o quanto o OceanUI é claro para vários tipos de público.

Atualmente o OceanUI foi desenvolvido para apresentar apenas uma explicação contrafactual por cenário, portanto um próximo passo pode ser a possibilidade de exibição de um conjunto de explicações contrafactuais, para dar mais opções para o usuário, similar ao que é feito no AdViCE (Gomez et al., 2021). E ao seguir por este caminho, podemos pensar em questões de escalabilidade do nosso projeto. Como o OceanUI funciona como um mediador entre o OCEAN e o usuário, a parte de cálculo complexo fica apenas no OCEAN e a comunicação acontece apenas por chamadas e retornos de funções, portanto a escalabilidade fica mais dependente em quantos cálculos de explicações contrafactuais são necessárias. Devido a isto, esta exibição de múltiplos contrafactuais pode exigir um tempo a mais de espera por parte do usuário, até que todos os elementos da interface sejam carregados e possam ser exibidos.

Ainda sobre o OceanUI, um outro trabalho em potencial é a possibilidade de inclusão de um gráfico similar ao Code (2021), onde este gráfico seria uma projeção de todo o *dataset* estudado, evidenciando a fronteira de decisão do modelo. De modo que a medida que o usuário alterasse o ponto corrente, o ponto correspondente na projeção "caminhasse" para mais perto ou mais longe desta fronteira de decisão. Esta visualização extra poderia auxiliar o usuário a entender como suas mudanças se comportam em relação ao *dataset*.

Também consideramos continuar os trabalhos sobre o OCEAN, para tentar mitigar os problemas observados (não conseguir gerar contrafactuais quando recebe muitas restrições simultâneas).

Além disto, como o OceanUI foi planejado para ser usado exclusivamente com o OCEAN, consideramos também como trabalhos futuros a possibilidade de desenvolvimento de uma interface similar ao OceanUI porém que aceite outros geradores de explicações contrafactuais. E como sugestão de desenvolvimento, pensamos em usar os novos geradores como *APIs*, apenas chamando funções para a comunicação, e se necessário, desenvolver uma "camada" extra no código atual para facilitar esta abordagem.

E para propósitos de ensino sobre ética e explicabilidade em algoritmos, pensamos na possibilidade de desenvolvimento de uma interface capaz de exibir modelos de predição de forma gráfica. Tomemos como exemplo um modelo de árvore de decisão, esta interface poderia exibir a árvore treinada, de modo que ao entrar com um ponto, a interface iria "desenhar o caminho" que este ponto toma até chegar na predição. E a medida que este ponto fosse sendo alterado, o "caminho" seria atualizado.

Referências Bibliográficas

Adebayo, J., Gilmer, J., Goodfellow, I., and Kim, B. (2018). Local explanation methods for deep neural networks lack sensitivity to parameter values. *arXiv preprint arXiv:1810.03307*.

Artelt, A. and Hammer, B. (2019). Efficient computation of counterfactual explanations of lvq models. *arXiv preprint arXiv:1908.00735*.

Artelt, A. and Hammer, B. (2020). Convex density constraints for computing plausible counterfactual explanations. In *International Conference on Artificial Neural Networks*, pages 353–365. Springer.

Bates, J., Cameron, D., Checco, A., Clough, P., Hopfgartner, F., Mazumdar, S., Sbaffi, L., Stordy, P., and de la Vega de León, A. (2020). Integrating fate/critical data studies into data science curricula: where are we going and how do we get there? In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 425–435.

Bibal, A., Lognoul, M., De Streel, A., and Frénay, B. (2021). Legal requirements on explainability in machine learning. *Artificial Intelligence and Law*, 29(2):149–169.

Bogina, V., Hartman, A., Kuflik, T., and Shulner-Tal, A. (2021). Educating software and ai stakeholders about algorithmic fairness, accountability, transparency and ethics. *International Journal of Artificial Intelligence in Education*, pages 1–26.

Burkart, N. and Huber, M. F. (2021). A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, 70:245–317.

Carvalho, D. V., Pereira, E. M., and Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832.

Cheng, F., Ming, Y., and Qu, H. (2020). Dece: Decision explorer with counterfactual explanations for machine learning models. *IEEE Transactions on Visualization and Computer Graphics*.

Code, P. (2021). What-if tool. `https://pair-code.github.io/what-if-tool/`. Último acesso: 14/04/2022.

Dandl, S., Molnar, C., Binder, M., and Bischl, B. (2020). Multi-objective counterfactual explanations. *arXiv preprint arXiv:2004.11165*.

Das, A. and Rad, P. (2020). Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*.

Fernández, R. R., de Diego, I. M., Aceña, V., Fernández-Isabel, A., and Moguerza, J. M. (2020). Random forest explainability using counterfactual sets. *Information Fusion*, 63:196–207.

Fuchs, D. J. (2018). The dangers of human-like bias in machine-learning algorithms. *Missouri S&T's Peer to Peer*, 2(1):1.

Gomez, O., Holter, S., Yuan, J., and Bertini, E. (2020). Vice: visual counterfactual explanations for machine learning models. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, pages 531–535.

Gomez, O., Holter, S., Yuan, J., and Bertini, E. (2021). Advice: Aggregated visual counterfactual explanations for machine learning model validation. In *2021 IEEE Visualization Conference (VIS)*, pages 31–35. IEEE.

Goodman, B. and Flaxman, S. (2017). European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine*, 38(3):50–57.

Guidotti, R., Monreale, A., Giannotti, F., Pedreschi, D., Ruggieri, S., and Turini, F. (2019). Factual and counterfactual explanations for black box decision making. *IEEE Intelligent Systems*, 34(6):14–23.

Kasinidou, M., Kleanthous, S., Orphanou, K., and Otterbacher, J. (2021). Educating computer science students about algorithmic fairness, accountability, transparency and ethics. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*, pages 484–490.

Kulesza, T., Burnett, M., Wong, W.-K., and Stumpf, S. (2015). Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th international conference on intelligent user interfaces*, pages 126–137.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35.

Molnar, C. (2020). *Interpretable Machine Learning*. Lulu. com.

Mothilal, R. K., Sharma, A., and Tan, C. (2020). Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617.

Parmentier, A. and Vidal, T. (2021). Optimal counterfactual explanations in tree ensembles.

Piesanen and Kerrigan (2020). Vida lab. `https://sites.google.com/nyu.edu/counterfactual-explorer/`. Último acesso: 14/04/2022.

Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., and Flach, P. (2020). Face: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350.

Pratap, A., Agarwall, A., and Meyarivan, T. (2002). Fast and elitist multiobjective genetic algorithm. *IEEE Trans. EVolut. Comput*, 6:182.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Russell, C. (2019). Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 20–28.

Stepin, I., Alonso, J. M., Catala, A., and Pereira-Fariña, M. (2021). A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access*, 9:11974–12001.

Sun, W., Nasraoui, O., and Shafto, P. (2020). Evolution and impact of bias in human and machine learning algorithm interaction. *Plos one*, 15(8):e0235502.

Van Looveren, A. and Klaise, J. (2019). Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584*.

Verma, S., Dickerson, J., and Hines, K. (2020). Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*.

White, A. and Garcez, A. d. (2019). Measurable counterfactual local explanations for any classifier. *arXiv preprint arXiv:1908.03020*.

A
Requisitos

Este anexo detalha os requisitos das diversas versões do sistema: a versão de geração instantânea e as versões de geração iterativa de contrafactuais.

A.1
Requisitos da Opção de Geração Instantânea

A.1.1
Requisitos Funcionais do Back-end

RF 1: inicialização da interface

- Descrição: o sistema inicializa a interface considerando os parâmetros padrão e instanciando os componentes necessários
- Entrada: início do evento
- Processo: o sistema verifica os *datasets* disponíveis para preencher o componente de seleção de *datasets*, instancia os componentes necessários para mostrar as informações das *features* e instancia toda a interface
- Saída: interface

RF 2: seleção de ponto aleatório

- Descrição: o sistema seleciona um ponto aleatório do *dataset*
- Entrada: requisição de um ponto aleatório
- Processo: o sistema toma o *dataset* selecionado, analisa seu tamanho e seleciona um valor aleatório considerando o índice e os pontos cuja classificação seja 0
- Saída: o ponto aleatório selecionado

RF 3: cálculo da classe

- Descrição: o sistema recebe um ponto e retorna a classe correspondente
- Entrada: ponto
- Processo: o sistema usa o modelo RF treinado para predizer a classe para o ponto fornecido pelo usuário

– Saída: classe predita

RF 4: cálculo da explicação contrafactual

– Descrição: o sistema recebe um ponto e retorna a explicação contrafactual correspondente
– Entrada: ponto
– Processo: o sistema usa cálculo de contrafactual (OCEAN) para gerar a explicação
– Saída: um ponto que representa a explicação contrafactual

A.1.2 Requisitos Não Funcionais do Back-end

RNF 1: linguagem: Python
RNF 2: principais pacotes: sklearn, gurobipy, pyTest

A.1.3 Requisitos Funcionais do Front-end

RF 1: seleção de *dataset*

– Descrição: o usuário seleciona um *dataset* de uma lista fornecida
– Entrada: evento de seleção
– Processo: dada uma lista de nomes de *datasets*, o usuário seleciona um deles para ser aberto e usado pelo *back-end*
– Saída: o sistema mostra uma lista com as *features* do *dataset*; caso a *feature* seja numérica, o sistema mostra informações de valores mínimo e máximo; caso a *feature* seja categórica ou binária, o sistema mostra todas a opções possíveis; e também oferece um meio para o usuário informar para o sistema o seu próprio valor

RF 2: seleção de ponto aleatório

– Descrição: o usuário seleciona uma opção da interface para receber um ponto aleatório
– Entrada: evento de seleção
– Processo: quando o usuário requisita um ponto aleatório, o *front-end* requisita do *back-end*
– Saída: um ponto aleatório do *dataset* selecionado

RF 3: cálculo da classe

- Descrição: o usuário seleciona uma opção de cálculo de classe para um determinado ponto
- Entrada: evento de seleção
- Processo: quando o usuário requisita o cálculo de classe para um determinado ponto, o *back-end* é chamado para executar o cálculo
- Saída: a classe correspondente

RF 4: geração de contrafactual

- Descrição: o usuário seleciona uma opção para cálculo de explicação contrafactual
- Entrada: evento de seleção
- Processo: quando o usuário requisita o cálculo de contrafactual para um determinado ponto, o *back-end* é chamado para executar o cálculo, levando em conta as restrições impostas pelo usuário sobre cada *feature*
- Saída: uma comparação entre o ponto original e o ponto retornado como contrafactual

A.1.4 Requisitos Não Funcionais do Front-end

RNF 1: plataforma: o sistema será desenvolvido para Desktop
RNF 2: linguagem: Python
RNF 3: pacote: pyQt
RNF 4: atualização de mínimo, máximo e valor corrente para *features* numéricas

- Descrição: o usuário pode editar os valores limites e o valor corrente para serem usados na geração do contrafactual

RNF 5: atualização de valores permitidos valores não permitidos e valor corrente para *features* categóricas

- Descrição: o usuário indica quais valores são permitidos, quais não são e qual é o valor corrente para serem usados na geração do contrafactual

RNF 6: atualização de valor corrente para *features* binárias

- Descrição: devem ser mostradas as opções possíveis, de modo que o usuário possa escolher uma delas

A.2
Requisitos da Opção de Geração Iterativa Versão 1

A.2.1
Requisitos Funcionais do Back-end

RF 1: inicialização da interface

- Descrição: o sistema inicializa a interface considerando os parâmetros padrão e instanciando os componentes necessários
- Entrada: início do evento
- Processo: o sistema verifica os *datasets* disponíveis para preencher o componente de seleção de *datasets*, instancia os componentes necessários para mostrar as informações das *features* e instância toda a interface
- Saída: interface

RF 2: seleção de ponto aleatório

- Descrição: o sistema seleciona um ponto aleatório do *dataset*
- Entrada: requisição de um ponto aleatório
- Processo: o sistema toma o *dataset* selecionado, analisa seu tamanho e seleciona um valor aleatório considerando o índice e os pontos cuja classificação seja 0
- Saída: o ponto aleatório selecionado

RF 3: cálculo da classe

- Descrição: o sistema recebe um ponto e retorna a classe correspondente
- Entrada: ponto
- Processo: o sistema usa o modelo RF treinado para predizer a classe para o ponto fornecido pelo usuário
- Saída: classe predita

RF 4: cálculo da explicação contrafactual

- Descrição: o sistema recebe um ponto e retorna a explicação contrafactual correspondente
- Entrada: ponto
- Processo: o sistema usa cálculo de contrafactual (OCEAN) para gerar a explicação
- Saída: um ponto que representa a explicação contrafactual

RF 5: geração de novo cenário

- Descrição: o sistema toma o ponto selecionado pelo usuário, as alterações sobre este ponto e gera um gráfico correspondente
- Entrada: requisição de novo cenário
- Processo: o sistema toma o ponto selecionado, aplica as alterações sobre ele usando $RF6$ e gera um gráfico usando $RF7$
- Saída: novo cenário

RF 6: alteração do ponto do usuário

- Descrição: o sistema recebe a atualização do ponto do usuário e recalcula a classe e a distância até um contrafactual mais próximo
- Entrada: evento de alteração
- Processo: o sistema recebe a modificação do ponto selecionado para recalcular classe usando $RF3$ e distância até um contrafactual
- Saída: classe e distância até um contrafactual

RF 7: geração de gráfico

- Descrição: o sistema toma o ponto selecionado pelo usuário e gera um gráfico com a vizinhança deste ponto levando em conta duas *features* escolhidas
- Entrada: requisição de gráfico
- Processo: o sistema usa o ponto selecionado, a vizinhança deste ponto e gera um gráfico $2D$ onde os eixos são *features* escolhidas pelo usuário. Os pontos devem ser representados com tamanhos diferentes para indicar a distância até um contrafactual mais próximo e com cores diferentes para indicar a classe de cada um deles
- Saída: gráfico

RF 8: seleção de *features*

- Descrição: o sistema recebe duas *features* para atualizar gráfico
- Entrada: requisição de gráfico
- Processo: o sistema usa $RF7$ levando em conta as duas *features* recebidas
- Saída: gráfico

A.2.2
Requisitos Não Funcionais do Back-end

RNF 1: linguagem: Python

RNF 2: principais pacotes: sklearn, gurobipy, pyTest

A.2.3
Requisitos Funcionais do Front-end

RF 1: seleção de *dataset*

- Descrição: o usuário seleciona um *dataset* de uma lista fornecida
- Entrada: evento de seleção
- Processo: dada uma lista de nomes de *datasets*, o usuário seleciona um deles para ser aberto e usado pelo *back-end*
- Saída: o sistema mostra uma lista com as *features* do *dataset*; caso a *feature* seja numérica, o sistema mostra informações de valores mínimo e máximo; caso a *feature* seja categórica ou binária, o sistema mostra todas a opções possíveis; e também oferece um meio para o usuário informar para o sistema o seu próprio valor

RF 2: seleção de ponto aleatório

- Descrição: o usuário seleciona uma opção da interface para receber um ponto aleatório
- Entrada: evento de seleção
- Processo: quando o usuário requisita um ponto aleatório, o *front-end* requisita do *back-end*
- Saída: um ponto aleatório do *dataset* selecionado

RF 3: cálculo da classe

- Descrição: o usuário seleciona uma opção de calculo de classe para um determinado ponto
- Entrada: evento de seleção
- Processo: quando o usuário requisita o cálculo de classe para um determinado ponto, o *back-end* é chamado para executar o cálculo
- Saída: a classe correspondente

RF 4: geração de contrafactual

- Descrição: o usuário seleciona uma opção para calculo de explicação contrafactual

- Entrada: evento de seleção
- Processo: quando o usuário requisita o cálculo de contrafactual para um determinado ponto, o *back-end* é chamado para executar o cálculo, levando em conta as restrições impostas pelo usuário sobre cada *feature*
- Saída: uma comparação entre o ponto original e o ponto retornado como contrafactual

RF 5: geração de novo cenário

- Descrição: o usuário seleciona uma opção para gerar um novo cenário
- Entrada: evento de seleção
- Processo: quando o usuário requisita novo cenário, o *back-end* é chamado para executar as alterações sobre o ponto selecionado, gerar novo gráfico e instanciar um novo cenário com as novas informações
- Saída: cenário

RF 6: alteração do ponto do usuário

- Descrição: o usuário informa que um determinado valor em uma *feature* deve ser alterado
- Entrada: evento de seleção
- Processo: quando o usuário seleciona uma *feature*, o *back-end* é chamado para recalcular a classe e a distância até um contrafactual
- Saída: classe e distância até um contrafactual

RF 7: alteração de *features*

- Descrição: o usuário seleciona duas *features* que devem ser consideradas como eixos de gráfico
- Entrada: evento de seleção
- Processo: quando o usuário seleciona duas *features*, o *back-end* é chamado para atualizar gráfico
- Saída: gráfico

A.2.4
Requisitos Não Funcionais do Front-end

RNF 1: plataforma: o sistema será desenvolvido para *Desktop*
RNF 2: linguagem: Python
RNF 3: pacote: pyQt
RNF 4: atualização de mínimo, máximo e valor corrente para *features* numéricas

- Descrição: o usuário pode editar os valores limites e o valor corrente para serem usados na geração do contrafactual

RNF 5: atualização de valores permitidos valores não permitidos e valor corrente para *features* categóricas

- Descrição: o usuário indica quais valores são permitidos, quais não são e qual é o valor corrente para serem usados na geração do contrafactual

RNF 6: atualização de valor corrente para *features* binárias

- Descrição: devem ser mostradas as opções possíveis, de modo que o usuário possa escolher uma delas

A.3
Requisitos da Opção de Geração Iterativa Versão Final

A.3.1
Requisitos Funcionais do Back-end

RF 1: inicialização da interface

- Descrição: o sistema inicializa a interface considerando os parâmetros padrão e instanciando os componentes necessários
- Entrada: início do evento
- Processo: o sistema verifica os *datasets* disponíveis para preencher o componente de seleção de *datasets*, instancia os componentes necessários para mostrar as informações das *features* e instancia toda a interface
- Saída: interface

RF 2: seleção de ponto aleatório

- Descrição: o sistema seleciona um ponto aleatório do *dataset*
- Entrada: requisição de um ponto aleatório

- Processo: o sistema toma o *dataset* selecionado, analisa seu tamanho e seleciona um valor aleatório considerando o índice e os pontos cuja classificação seja 0
- Saída: o ponto aleatório selecionado

RF 3: cálculo da classe

- Descrição: o sistema recebe um ponto e retorna a classe correspondente
- Entrada: ponto
- Processo: o sistema usa o modelo RF treinado para predizer a classe para o ponto fornecido pelo usuário
- Saída: classe predita

RF 4: cálculo da explicação contrafactual

- Descrição: o sistema recebe um ponto e retorna a explicação contrafactual correspondente
- Entrada: ponto
- Processo: o sistema usa cálculo de contrafactual (OCEAN) para gerar a explicação
- Saída: um ponto que representa a explicação contrafactual

RF 5: geração de novo cenário

- Descrição: o sistema toma o ponto selecionado pelo usuário, as alterações sobre este ponto e gera um gráfico correspondente
- Entrada: requisição de novo cenário
- Processo: o sistema toma o ponto selecionado, aplica as alterações sobre ele usando $RF6$ e gera um gráfico usando $RF7$
- Saída: novo cenário

RF 6: alteração do ponto do usuário

- Descrição: o sistema recebe a atualização do ponto do usuário e recalcula a classe
- Entrada: evento de alteração
- Processo: o sistema recebe a modificação do ponto selecionado para recalcular classe usando $RF3$
- Saída: classe

RF 7: geração de gráfico

- Descrição: o sistema toma o ponto selecionado pelo usuário, ponto do cenário anterior e ponto contrafactual para gerar um gráfico correspondente
- Entrada: requisição de gráfico
- Processo: o sistema usa o ponto selecionado, o ponto do cenário anterior e o ponto contrafactual calculado usando $RF4$, para gerar um gráfico de coordenadas paralelas usando um número n de *features* como eixos
- Saída: gráfico

RF 8: seleção de *features*

- Descrição: o sistema recebe as *features* selecionadas para atualizar gráfico
- Entrada: requisição de gráfico
- Processo: o sistema usa $RF7$ levando em conta as *features* recebidas
- Saída: gráfico

A.3.2 Requisitos Não Funcionais do Back-end

RNF 1: linguagem: Python
RNF 2: principais pacotes: sklearn, gurobipy, pyTest

A.3.3 Requisitos Funcionais do Front-end

RF 1: seleção de *dataset*

- Descrição: o usuário seleciona um *dataset* de uma lista fornecida
- Entrada: evento de seleção
- Processo: dada uma lista de nomes de *datasets*, o usuário seleciona um deles para ser aberto e usado pelo *back-end*
- Saída: o sistema mostra uma lista com as *features* do *dataset*; caso a *feature* seja numérica, o sistema mostra informações de valores mínimo e máximo; caso a *feature* seja categórica ou binária, o sistema mostra todas a opções possíveis; e também oferece um meio para o usuário informar para o sistema o seu próprio valor

RF 2: seleção de ponto aleatório

- Descrição: o usuário seleciona uma opção da interface para receber um ponto aleatório

- Entrada: evento de seleção
- Processo: quando o usuário requisita um ponto aleatório, o *front-end* requisita do *back-end*
- Saída: um ponto aleatório do *dataset* selecionado

RF 3: cálculo da classe

- Descrição: o usuário seleciona uma opção de calculo de classe para um determinado ponto
- Entrada: evento de seleção
- Processo: quando o usuário requisita o cálculo de classe para um determinado ponto, o *back-end* é chamado para executar o cálculo
- Saída: a classe correspondente

RF 4: geração de contrafactual

- Descrição: o usuário seleciona uma opção para calculo de explicação contrafactual
- Entrada: evento de seleção
- Processo: quando o usuário requisita o cálculo de contrafactual para um determinado ponto, o *back-end* é chamado para executar o cálculo, levando em conta as restrições impostas pelo usuário sobre cada *feature*
- Saída: uma comparação entre o ponto original e o ponto retornado como contrafactual

RF 5: geração de novo cenário

- Descrição: o usuário seleciona uma opção para gerar um novo cenário
- Entrada: evento de seleção
- Processo: quando o usuário requisita novo cenário, o *back-end* é chamado para executar as alterações sobre o ponto selecionado, gerar novo gráfico e instanciar um novo cenário com as novas informações
- Saída: cenário

RF 6: alteração do ponto do usuário

- Descrição: o usuário informa que um determinado valor em uma *feature* deve ser alterado
- Entrada: evento de seleção
- Processo: quando o usuário seleciona uma *feature*, o *back-end* é chamado para recalcular a classe e atualizar o gráfico

– Saída: gráfico

RF 7: alteração de *features*

– Descrição: o usuário seleciona duas *features* que devem ser consideradas como eixos de gráfico
– Entrada: evento de seleção
– Processo: quando o usuário seleciona duas *features*, o *back-end* é chamado para atualizar gráfico
– Saída: gráfico

A.3.4 Requisitos Não Funcionais do Front-end

RNF 1: plataforma: o sistema será desenvolvido para Desktop
RNF 2: linguagem: Python
RNF 3: pacote: pyQt
RNF 4: atualização de mínimo, máximo e valor corrente para *features* numéricas

– Descrição: o usuário pode editar os valores limites e o valor corrente para serem usados na geração do contrafactual

RNF 5: atualização de valores permitidos valores não permitidos e valor corrente para *features* categóricas

– Descrição: o usuário indica quais valores são permitidos, quais não são e qual é o valor corrente para serem usados na geração do contrafactual

RNF 6: atualização de valor corrente para *features* binárias

– Descrição: devem ser mostradas as opções possíveis, de modo que o usuário possa escolher uma delas

B
Casos de Uso

Este anexo detalha os casos de uso tanto da versão de geração instantânea, quanto das versões de geração iterativa de contrafactuais.

B.1
Caso de Uso de Geração Instantânea

Fluxo:

1. usuário abre a interface
2. usuário seleciona um *dataset*
3. usuário escolhe um ponto aleatório ou preenche manualmente os valores
4. usuário pode informar suas restrições
5. usuário seleciona *"Calculate Class"* e/ou *"Generate Counterfactual"*
6. o sistema mostra o resultado
7. usuário pode retornar para o passo 3 para testar outro ponto ou para o passo 4 para incluir novas restrições

B.2
Caso de Uso de Geração Iterativa Versão 1

Fluxo:

1. usuário abre a interface
2. usuário seleciona um *dataset*
3. usuário escolhe um ponto aleatório ou preenche manualmente os valores
4. usuário pode informar suas restrições
5. usuário seleciona a(s) *feature(s)* que deve(m) ser usada(s) para plotar o gráfico com as distâncias correspondentes para uma determinada amostragem de pontos

6. o sistema mostra/atualiza o gráfico em função da(s) *feature(s)* selecionada(s)

7. usuário seleciona um ponto do gráfico para atualizar o valor da *feature* correspondente

8. usuário pode retornar para o passo 3 para testar outro ponto ou para o passo 4 para incluir novas restrições

B.3 Caso de Uso de Geração Iterativa Versão Final

Fluxo:

1. usuário abre a interface

2. usuário seleciona um *dataset*

3. usuário pode analisar um gráfico de importância das *features* para o modelo treinado

4. usuário escolhe um ponto aleatório ou preenche manualmente os valores

5. usuário pode informar suas restrições

6. usuário instância novo cenário

   - se as restrições forem muito estritas o novo cenário não será instanciado
   - usuário atualiza as restrições
   - usuário instância novo cenário

7. sistema apresenta, neste novo cenário, um gráfico em função das *features* sugeridas de serem alteradas pelo contrafactual calculado

8. usuário manipula o ponto corrente por meio do gráfico

9. usuário pode informar novas restrições por meio do gráfico

10. usuário pode voltar ao passo 6 ou instanciar o cenário final

11. sistema apresenta, neste cenário final, as classes do ponto inicial e do ponto que gerou o cenário final, juntamente com uma tabela de comparação entre eles

C
Material do Estudo

Este anexo detalha o material fornecido para os usuários durante o teste, que consistiu em um Termo de Consentimento Livre e Esclarecido (TCLE – seção C.1), um questionário de caracterização do perfil do participante (seção C.2), uma tabela com as informações do *dataset* (seção C.3), um roteiro das tarefas que o participante deveria realizar (seção C.4) e um roteiro de entrevista para captura de opiniões sobre o sistema (seção C.5).

C.1
Termo de Consentimento Livre e Esclarecido

O TCLE é apresentado nas páginas a seguir.

Termo de Consentimento Livre e Esclarecido

Natureza da Pesquisa

Eu, Moisés Henrique Pereira, aluno de Mestrado do Departamento de Informática da PUC-Rio, responsável pelo estudo comparativo de duas ferramentas de geração e visualização de explicações contrafactuais (OceanUI e VidaLab), sob coordenação da Professora Simone Diniz Junqueira Barbosa, do Departamento de Informática da PUC-Rio e do Professor Thibaut Victor Gaston Vidal, do Departamento de Matemática e Engenharia Industrial Polytechnique Montréal, lhe convido a participar como voluntário neste estudo.

Nossa pesquisa visa investigar melhores formas de geração e exibição de explicações contrafactuais por meio de uma interface, levando em consideração a acionabilidade das explicações e também formas de deixá-las mais personalizadas para cada usuário. O objetivo do estudo não é avaliar o desempenho das pessoas, mas sim o quão adequadas estão as ferramentas OceanUI e VidaLab. Através desta pesquisa espera-se identificar problemas e oportunidades de melhoria nos sistemas.

Benefícios

Os benefícios esperados envolvem, uma ferramenta de geração de contrafactuais aperfeiçoada e artigos científicos sobre os diversos aspectos da pesquisa. No entanto, não há benefícios a curto prazo esperados para os participantes do estudo.

Riscos e desconfortos

Identificamos alguns riscos mínimos associados à participação nesta pesquisa:

1. **Desconforto físico:** cansaço ou aborrecimento caso a sessão seja longa (acima de 1 hora). Vamos minimizar o tempo necessário à realização das atividades, focando nas questões mais relevantes ao objetivo do estudo.
2. **Constrangimento por causa da gravação de áudio e vídeo:** Iremos gravar áudio e vídeo somente mediante o seu consentimento. Além disso, uma vez que o material coletado seja processado, ele será descartado.
3. **Quebra da segurança digital dos dados armazenados:** Os dados coletados serão armazenados em ambiente seguro (mídia ou máquina sem acesso à internet ou em área protegida por senha). Além disto, o material coletado será desassociado da sua identidade, para garantir o seu anonimato e privacidade.
4. **Qualquer tipo de incômodo ou constrangimento:** Você pode interromper a pesquisa a qualquer momento e sem qualquer prejuízo, penalização ou constrangimento. Em nenhum lugar ficará registrado que você iniciou sua participação no estudo e optou por interrompê-la.

Garantia de anonimato, privacidade e sigilo dos dados

Esta pesquisa se pauta no respeito à privacidade, ao sigilo e ao anonimato dos participantes. Todos os dados brutos serão acessados somente pelos pesquisadores envolvidos nesta pesquisa e anonimizados para análise ou divulgação. O uso que faremos

dos dados coletados durante o teste é estritamente limitado a atividades científicas, didáticas e de desenvolvimento de ferramentas que apoiem atividades de consumo e produção de visualizações de informação. Qualquer imagem, vídeo ou áudio divulgado será disfarçado para impedir a identificação dos participantes que nela aparecem.

Divulgação dos resultados

Os dados agregados e análises realizadas poderão ser publicados em publicações científicas e didáticas. Ao divulgarmos os resultados da pesquisa, nos comprometemos em preservar seu anonimato e privacidade, ocultando ou disfarçando toda informação (seja em texto, imagem, áudio ou vídeo) que possa revelar sua identidade. As informações brutas coletadas não serão divulgadas.

Acompanhamento, assistência e esclarecimentos

Todo material coletado será arquivado por no mínimo cinco anos. A Professora Simone Diniz Junqueira Barbosa, do Departamento de Informática da PUC-Rio, será a responsável por arquivar o material da pesquisa ao longo deste período e até o seu descarte. A qualquer momento, durante a pesquisa e até seis meses após o seu término, você poderá solicitar mais informações sobre o estudo ou cópias dos materiais divulgados. Caso você observe algum comportamento que julgue antiético ou prejudicial a você, você pode entrar em contato para que sejam tomadas as medidas necessárias. Ao final deste termo você encontra as formas de contato conosco ou com a Câmara de Ética em Pesquisa da PUC-Rio.

Ressarcimento de despesa eventual

Ao aceitar este termo, você não abre mão de nenhum direito legal. Se, por algum motivo, você tiver despesas decorrentes de sua participação nesse estudo, com transporte e/ou alimentação, você será reembolsado adequadamente pelos pesquisadores, conforme acordado no momento do recrutamento.

Liberdade de recusa, interrupção, desistência e retirada de consentimento

Sua participação nesta pesquisa é voluntária. Sua recusa não trará nenhum prejuízo a você, nem à sua relação com os pesquisadores ou com a universidade. A qualquer momento você pode interromper ou desistir da pesquisa, sem que incorra nenhuma penalização ou constrangimento. Você não precisará sequer justificar ou informar o motivo da interrupção ou desistência. Caso você mude de ideia sobre seu consentimento durante a sessão de estudo, basta comunicar sua decisão aos pesquisadores responsáveis, que então descartarão seus dados.

Consentimento

Eu, participante abaixo assinado(a), confirmo que:

1. Recebi informações detalhadas sobre a natureza e objetivos da pesquisa descrita neste documento e tive a oportunidade e esclarecer eventuais dúvidas;
2. Estou ciente de que minha participação é voluntária e posso abandonar o estudo a qualquer momento, sem fornecer uma razão e sem que haja quaisquer

consequências negativas. Além disso, caso eu não queira responder a uma ou mais questões, tenho liberdade para isto;

3. Estou ciente de que minhas respostas serão mantidas confidenciais. Entendo que meu nome não será associado aos materiais de pesquisa e não será identificado nos materiais de divulgação que resultem da pesquisa;
4. Estou ciente de que a minha participação não acarretará qualquer ônus e que as atividades previstas na pesquisa não representam nenhum risco para mim ou para qualquer outro participante;
5. Estou ciente de que sou livre para consentir ou não com a pesquisa, conforme as opções que marco abaixo:

Sobre a coleta e uso de dados:

☐ **Não autorizo** o uso das informações coletadas descritas neste documento.

☐ **Autorizo** o uso das informações coletadas conforme as condições descritas neste termo.

Sobre a gravação de áudio:

☐ **Não autorizo** a gravação em áudio do que eu disser durante o estudo.

☐ **Autorizo** a gravação em áudio do que eu disser durante o estudo.

Sobre a gravação de vídeo:

☐ **Não autorizo** a gravação em vídeo das atividades que eu realizar.

☐ **Autorizo** a gravação em vídeo das atividades que eu realizar.

Rio de Janeiro, ____ de ____ de 2022

Pesquisador: Moises Henrique Pereira _____

Participante: _____

_____

Contatos: (1) Profa. Simone Diniz Junqueira Barbosa, Departamento de Informática, PUC-Rio - simone@inf.puc-rio.br - +55 21 3527-1500 ext. 4353; +55 21 992-413-651. (2) Câmara de Ética em Pesquisa da PUC-Rio: Rua Marquês de São Vicente, 225, Prédio Kennedy, 2º andar – Gávea – RJ.

C.2
Questionário de Caracterização do Perfil do Participante

O questionário de perfil de usuário é apresentado nas páginas a seguir.

Perfil de usuário

Este formulário é para coletar informações sobre o perfil dos voluntários que estão testando os sistemas OceanUI e VidaLab.

*Obrigatório

1. Qual a sua escolaridade? *

   Marcar apenas uma oval.

   - ◯ graduação
   - ◯ mestrado
   - ◯ doutorado
   - ◯ Outro: _____

2. De 1 a 5, tenho familiaridade com sistemas de predição (como usuário). *

   Marcar apenas uma oval.

	1	2	3	4	5	
discordo totalmente	◯	◯	◯	◯	◯	concordo totalmente

3. Você sabe do que se trata uma explicação contrafactual? Se sim, por favor, descreva. *

_____

_____

_____

_____

_____

4. Já usou algum sistema de geração de explicações contrafactuais? *

Marcar apenas uma oval.

- sim
- não

5. De 1 a 5, tenho familiaridade com utilização de gráficos de coordenadas paralelas. *

Marcar apenas uma oval.

	1	2	3	4	5	
discordo totalmente						concordo totalmente

C.3 Informações do Dataset

Por questões de simplicidade, selecionamos apenas o *dataset "German-Credit"*. Este *dataset* contém informações que ajudam na concessão de empréstimo levando em conta a predição de se o usuário é um bom ou mal pagador (mais detalhes podem ser encontrados em: `https://www.kaggle.com/uciml/german-credit`).

Porém, diferentemente do *dataset* encontrado no link acima, o que utilizamos passou por um tratamento de inclusão de um cabeçalho informando o tipo e a acionabilidade de cada *feature* e também pelo processo de *binning* por frequência em algumas *features*. Cada uma das *features* presentes nesse *dataset* pode ser vista na tabela a seguir, juntamente com seus respectivos tipos, significados e possíveis valores:

Feature	**Tipo**	**Significado**	**Possíveis valores**
Age	numérica	intervalo de idade	0 - [19, 24) 1 - [24, 27) 2 - [27, 30) 3 - [30, 34) 4 - [34, 39) 5 - [39, 48) 6 - [48, 75]
Sex	binária	sexo	"female- feminino "male- masculino
Job	categórica	trabalho	0 - não qualificado e não residente 1 - não qualificado e residente 2 - qualificado 3 - altamente qualificado
Housing	categórica	moradia	"own- casa própria "rent- aluguel "free- concessão
SavingAccounts	numérica	poupança	1 - classe baixa 2 - classe média 3 - classe média alta 4 - classe alta
CheckingAccount	numérica	conta corrente	1 - classe baixa 2 - classe média 3 - classe alta
CreditAmount	numérica	intervalo de quantidade de crédito	0 - [0, 1049) 1 - [1049, 1374) 2 - [1374, 1987) 3 - [1987, 2671) 4 - [2671, 3676) 5 - [3676, 6187) 6 - [6187, 18424]
Duration	numérica	intervalo de tempo em meses para pagar	0 - [6, 10) 1 - [10, 12) 2 - [12, 18) 3 - [18, 21) 4 - [21, 24) 5 - [24, 36) 6 - [36, 72]
Purpose	categórica	objetivo do empréstimo	"car- compra de carro "radio/TV- compra de rádio ou TV "furniture/equipment- compra de móveis "business- aplicações "education- educação "repairs- reformas "vacation/others- férias ou outros propósitos "domestic appliances- eletrodomésticos

C.4
Roteiro de Tarefas

Tarefas OceanUI:

1. leia o tutorial
2. os dados correspondentes já estão preenchidos, vide Ponto1
3. descubra se esta pessoa teria seu empréstimo aprovado ou reprovado
4. vá para o gráfico de coordenadas paralelas
5. lhe foi informado que a *feature "Purpose"* não pode ser *"vacation"*, passe isso para o sistema
6. lhe foi informado que a *feature "CreditAmount"* precisa ser no mínimo 1, passe isso para o sistema
7. gere um novo *"Scenario"*
8. manipule o ponto corrente até que o empréstimo deva ser aprovado
9. informe a lista de comparação com as sugestões que a pessoa pode usar

Tarefas VIDALab:

1. leia a documentação
2. no notebook do VIDA Lab aberto, os dados correspondentes já estão preenchidos, vide Ponto2
3. encontre um contrafactual
   - se o usuário não informar os pesos, o sistema apresenta um bug
   - todos os pesos devem ser mudados
   - número do conjunto contrafactual pode ser mudado
4. mostre este contrafactual em lista
5. o resultado obtido sugere mudanças que não podem ser executadas na prática
6. encontre um contrafactual
   - se o usuário não informar os pesos, o sistema apresenta um bug
   - todos os pesos devem ser mudados
   - número do conjunto contrafactual pode ser mudado

7. mostre este contrafactual em lista

8. o resultado obtido sugere mudanças que não podem ser executadas na prática

Ponto1:

Feature	**Valor**
Age	4
Sex	male
Job	2
Housing	own
SavingAccounts	1
CheckingAccount	2
CreditAmount	5
Duration	5
Purpose	business

Ponto2:

Feature	**Valor**
Age	3
Sex	male
Job	2
Housing	rent
SavingAccounts	1
CheckingAccount	1
CreditAmount	3
Duration	5
Purpose	radio/TV

C.5
Roteiro de entrevista para Captura de Opiniões sobre o Sistema

Ao testar o OceanUI:

1. O que achou ao utilizar o OceanUI?
2. O que o sistema facilitou durante a utilização?
3. O que o sistema dificultou durante a utilização?
4. Quais foram as partes mais úteis do sistema?
5. O que mais lhe agradou?
6. O que menos gostou?
7. Me diga o que achou da visualização dos pontos. Em que situações a visualização te ajudou? O que acha que ela atrapalhou?
8. Me diga o que achou da inclusão das restrições. Em que situações a inclusão de restrições te ajudou? O que acha que ela atrapalhou?
9. Você sente que faltou alguma funcionalidade? Qual seria e por quê?
10. Teria alguma sugestão de melhoria para a ferramenta?

Ao testar o VIDALab:

1. O que achou ao utilizar o VIDALab?
2. O que o sistema facilitou durante a utilização?
3. O que o sistema dificultou durante a utilização?
4. Quais foram as partes mais úteis do sistema?
5. O que mais lhe agradou?
6. O que menos gostou?
7. Você sente que faltou alguma funcionalidade? Qual seria e por quê?
8. Teria alguma sugestão de melhoria para a ferramenta?

Comparativo ao final dos dois testes:

1. Em quais situações você preferiria utilizar o VIDALab? Por quê?
2. Em quais situações você preferiria utilizar o OceanUI? Por quê?
3. Qual dos sistemas achou mais fácil de usar? Por quê?
4. Qual dos sistemas achou mais útil? Por quê?