NATACHA TARGINO RODRIGUES SIMÕES BRASILEIRO

# USO DE DADOS DE PROVENIÊNCIA EM PIPELINES DE APRENDIZADO DE MÁQUINA: UM CASO DE USO COM SELEÇÃO DE ATRIBUTOS.

**UNIVERSIDADE FEDERAL DE PERNAMBUCO**
*posgraduacao@cin.ufpe.br*
*www.cin.ufpe.br/~posgraduacao*

Recife
2025

NATACHA TARGINO RODRIGUES SIMÕES BRASILEIRO

# USO DE DADOS DE PROVENIÊNCIA EM PIPELINES DE APRENDIZADO DE MÁQUINA: UM CASO DE USO COM SELEÇÃO DE ATRIBUTOS.

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de doutor(a) em Ciência da Computação.
**Área de Concentração:** Banco de Dados

**Orientadora:** Ana Carolina Brandão Salgado.
**Co-Orientadora:** Damires Yluska de Souza Fernandes.

Recife
2025

.Catalogação de Publicação na Fonte. UFPE - Biblioteca Central

Brasileiro, Natacha Targino Rodrigues Simões.
Uso de dados de proveniência em pipelines de aprendizado de máquina: um caso de uso com seleção de atributos / Natacha Targino Rodrigues Simões Brasileiro. - Recife, 2025.
122f.: il.

Tese (Doutorado)-Universidade Federal de Pernambuco, Centro de Informática - CIn, Programa de Pós-Graduação em Ciência da Computação, 2025.
Orientação: Ana Carolina Brandão Salgado.
Coorientação: Damires Yluska de Souza Fernandes.

1. Aprendizado de máquina; 2. Dados de proveniência; 3. Seleção de atributos; 4. Ontologias. I. Salgado, Ana Carolina Brandão. II. Fernandes, Damires Yluska de Souza. III. Título.

UFPE-Biblioteca Central

**Natacha Targino Rodrigues Simões Brasileiro**

**“Uso de Dados de Proveniência em Pipelines de Aprendizado de Máquina: Um Caso de Uso com Seleção de Atributos”**

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Banco de Dados

Aprovada em: 27/02/2025.

_____

**Orientadora: Profa. Dra. Ana Carolina Brandão Salgado**

**BANCA EXAMINADORA**

_____

Prof. Dr. Fernando da Fonseca de Souza
Centro de Informática / UFPE

_____

Prof. Dr. Marcelo de Sousa Oliveira
Centro de Informática / UFPB

_____

Prof. Dr. Diego Ernesto Rosa Pessoa
Unidade Acadêmica de Informática / IFPB

_____

Prof. Dr. Dimas Cassimiro do Nascimento Filho
Universidade Federal do Agreste de Pernambuco

_____

Profa. Dra. Andrêza Leite de Alencar
Departamento de Computação / UFRPE

*À minha querida avó, Lúcia Maria Targino Moreira Rodrigues (in memoriam).*

# Agradecimentos

Agradeço, em primeiro lugar, a Deus, por me conceder sabedoria e força para superar cada desafio ao longo da minha trajetória. Sou imensamente grata por sentir Sua presença constante, guiando meus passos e me fortalecendo a todo momento.

Aos meus pais, Régia e Jandui, por me ensinarem valores que levarei para toda a vida, por estarem sempre ao meu lado, celebrando minhas conquistas e me guiando nos desafios. Agradeço por cada gesto de amor, pelos conselhos e por todas as orações feitas. À minha querida avó, Lúcia, por todo o apoio e carinho que sempre me dedicou, especialmente em meus estudos. Embora não esteja mais aqui para testemunhar este momento, seu amor e sabedoria permanecerão para sempre em meu coração. Aos meus irmãos, Shimena, Sayonara, Quênia e Caio, agradeço por todo o companheirismo, por sempre estarem presentes em minha vida, compartilhando os momentos com alegria. Aos meus sobrinhos, Augusto e Sofia, que eu amo tanto, por serem a coisa mais fofa e por trazerem tanta alegria e amor para as nossas vidas. As risadas compartilhadas, as palavras de incentivo e os atos de carinho tornam nossa família ainda mais especial!

Minha profunda gratidão às minhas orientadoras, Ana Carolina Salgado e Damires Souza, cujo suporte foi fundamental para o desenvolvimento deste trabalho e para o meu crescimento acadêmico. Agradeço por cada ensinamento e por me guiarem com tanto compromisso e dedicação. Sou especialmente grata por terem sido persistentes comigo entre tantos percalços nessa caminhada tão longa. Vocês foram fundamentais para que eu chegasse até aqui e levarei comigo cada aprendizado.

Também agradeço aos meus colegas Gabrielle, Neto, Jéssyca, Jáder, Marcelo, Douglas, Michael, Léuson e Karine, que fizeram toda a diferença no dia a dia no CIn. Nos períodos em que cursamos as disciplinas e no convívio no laboratório, principalmente antes da pandemia, compartilhamos vários momentos de aprendizado, superação, alegrias e aperreios. Sou grata por cada um de vocês estar ao meu lado nessa caminhada.

Por último, expresso minha gratidão a todos que colaboraram, me deram suporte ao longo deste processo e tornaram a realização deste trabalho possível.

# Resumo

Com o aumento exponencial dos dados e o desenvolvimento de sistemas inteligentes baseados em Aprendizado de Máquina (AM), surgem novas oportunidades e desafios. A eficácia desses sistemas depende da compreensão dos princípios do AM, principalmente na utilização de algoritmos supervisionados, que aprendem a partir de dados rotulados para realizar tarefas de previsão. Nesse contexto, dados de proveniência oferecem uma oportunidade de rastrear e entender decisões feitas durante as execuções anteriores de *pipelines* de AM, promovendo a transparência e rastreabilidade desses processos. Embora a literatura explore o uso de dados de proveniência em AM, sua aplicação em atividades de seleção de atributos ainda é pouco explorada, apesar do potencial para automatizar ajustes e melhorar a avaliação dos modelos. O presente trabalho propõe uma abordagem focada em dados de proveniência de execuções de *pipelines* de AM, com o objetivo de explorar o papel desses dados na reexecução e ajuste de atividades de seleção de atributos em *pipelines* de AM. Especificamente, investigam-se duas questões de pesquisa: (1) como dados de proveniência capturados durante a execução de um *pipeline* de AM podem ser utilizados para registrar e viabilizar a reexecução consistente de atividades específicas em momentos futuros, e (2) como as informações obtidas a partir dos dados de proveniência de execuções anteriores de *pipelines* de AM podem auxiliar na realização de ajustes na seleção de atributos, de forma a contribuir para a melhoria da avaliação dos modelos de AM. A solução apresentada envolve a captura de dados de proveniência durante a execução de *pipelines* e a estruturação semântica desses dados usando uma extensão da Ontologia PROV (W3C). A estruturação visa otimizar a reutilização das informações para ajustar e melhorar a avaliação dos modelos de AM. A abordagem permite ajustar a seleção de atributos com base em execuções anteriores, promovendo o aprimoramento contínuo do modelo. Para avaliar a proposta, foi desenvolvido um protótipo que automatiza esse processo. Em experimentos com diferentes tarefas de treinamento de modelos de AM, foi observado que os ajustes baseados em dados de proveniência resultaram em melhorias nas métricas de acurácia e *F1-score* dos modelos de AM gerados. Os resultados indicam que o uso de dados de proveniência contribui para otimizar o processo de treinamento, especialmente ao considerar a reexecução e o ajuste das atividades. As principais contribuições deste trabalho incluem a definição da ontologia PROVX, a qual permite modelar e gerenciar os dados de proveniência dos *pipelines* de AM, e a proposta de uma estratégia de seleção de atributos que facilita o aprimoramento dos modelos com base nesses dados.

**Palavras-chave:** Aprendizado de Máquina, Dados de Proveniência, Seleção de Atributos, Ontologias.

# Abstract

With the exponential growth of data and the development of intelligent systems based on Machine Learning (ML), new opportunities and challenges have emerged. The effectiveness of these systems depends on a comprehensive understanding of ML principles, particularly the use of supervised algorithms that learn from labeled data to perform predictive tasks. In this context, provenance data offers a valuable opportunity to trace and understand decisions made during previous executions of ML pipelines, thereby promoting transparency and traceability in these processes. Although the literature explores the use of provenance data in ML, its application to feature selection activities remains underexplored, despite its potential to automate tuning processes and improve model evaluation. This study proposes an approach focused on provenance data derived from ML pipeline executions, aiming to investigate the role of such data in the reexecution and adjustment of feature selection activities within ML pipelines. Specifically, two research questions are addressed: (1) how provenance data captured during the execution of an ML pipeline can be used to record and enable the consistent re-execution of specific activities at later stages, and (2) how information obtained from provenance data of previous ML pipeline executions can support adjustments to feature selection in order to enhance ML model evaluation. The proposed solution involves the capture of provenance data during pipeline executions and the semantic structuring of this data using an extension of the PROV Ontology (W3C). This structuring aims to optimize the reuse of information to refine and improve ML model evaluation. The approach enables feature selection adjustments based on prior executions, thereby supporting the continuous enhancement of the model. To evaluate the proposal, a prototype was developed to automate the process. In experiments involving different ML model training tasks, it was observed that adjustments based on provenance data led to improvements in accuracy and F1-score metrics of the resulting models. The results indicate that the use of provenance data contributes to the optimization of the training process, particularly when considering the re-execution and adjustment of specific activities. The main contributions of this work include the definition of the PROVX ontology, which enables the modeling and management of provenance data from ML pipelines, and the proposal of a feature selection strategy that facilitates model enhancement based on such data.

**Keywords:** Machine Learning, Provenance Data, Feature Selection, Ontologies.

# Lista de Figuras

# Lista de Quadros

# Lista de Tabelas

# Lista de Acrônimos

| | |
|---|---|
| **AM** | Aprendizado de Máquina |
| **CRISP-DM** | *CRoss-Industry Standard Process for Data Mining* |
| **CSV** | *Comma-separated values* |
| **IA** | Inteligência Artificial |
| **KNN** | K-Nearest-Neighbors |
| **NoSQL** | *Not Only SQL* |
| **OPM** | *Open Provenance Model* |
| **OWL** | Ontology Web Language |
| **PROV** | PROV Family of Documents |
| **PROV-DM** | *Provenance Data Model* |
| **PROV-O** | *Provenance Ontology* |
| **RDF** | *Resource Description Framework* |
| **RFE** | *Recursive Feature Elimination* |
| **SPARQL** | *Protocol and Resource Description Framework Query Language* |
| **SVM** | *Support Vector Machines* |
| **W3C** | *World Wide Web Consortium* |
| **WIR** | *Workflow Intermediate Representation* |
| **XAI** | *Explainable Artificial Intelligence* |
| **URIs** | *Uniform Resource Identifiers* |

# Sumário

# 1 INTRODUÇÃO

Neste capítulo, é descrito o contexto no qual este trabalho está inserido. Inicialmente, é apresentada a motivação do trabalho, seguida da definição do problema. Também são estabelecidos os objetivos, uma visão geral da solução proposta, a metodologia de pesquisa e as contribuições esperadas. Por fim, a organização geral do documento é exposta.

## 1.1 CONTEXTO E MOTIVAÇÃO

Com o crescimento exponencial dos dados produzidos ao longo dos anos, técnicas foram desenvolvidas com o objetivo de extrair informações valiosas (OMITAOMU et al., 2021). A adoção de metodologias baseadas em Inteligência Artificial (IA), especialmente em Aprendizado de Máquina (AM), tem gerado oportunidades e desafios para uma melhor utilização desses dados, incluindo o desenvolvimento de sistemas inteligentes capazes de realizar tarefas, como previsões e tomadas de decisões baseadas nas previsões obtidas.

Compreender os princípios que norteiam o AM, incluindo os algoritmos e as etapas relacionadas ao treinamento de modelos de AM, é essencial para melhorar a eficácia dessas técnicas. Os algoritmos de aprendizado supervisionado, em particular, desempenham um papel crucial nesse processo. Sendo uma das categorias mais comuns e utilizadas em AM, o aprendizado supervisionado faz uso de algoritmos que aprendem a partir de dados rotulados para realizar tarefas de previsão (JIANG et al., 2020).

Para o treinamento de modelos de AM, podemos considerar as fases fundamentais para o desenvolvimento do modelo: coleta de dados, preparação de dados, treinamento do modelo, avaliação do modelo e otimização (TEUBL et al., 2023). Nessas fases, podemos ter uma série de tarefas específicas, como na fase de preparação de dados, que pode incluir a limpeza e transformação dos mesmos, e a seleção de atributos relevantes. Os resultados obtidos por esses modelos dependem dos dados coletados e das atividades realizadas sobre eles, as quais influenciam diretamente os resultados dos modelos de AM. Alguns trabalhos, como os de CELDRAN et al. (2023) e SCHELTER et al. (2023), indicam a relevância do conhecimento detalhado sobre um *pipeline* de AM, em nível de execução e configuração. Esse conhecimento compreende as atividades realizadas para o treinamento de modelos de AM, como o entendimento do modelo gerado, os parâmetros utilizados e a detecção de problemas nos dados, de forma a tornar o processo mais transparente e rastreável.

A partir do treinamento de modelos de AM, pode ser interessante a obtenção de dados de proveniência que permitam rastrear e facilitar o entendimento das decisões tomadas durante

a execução do *pipeline*. A proveniência se refere a qualquer informação que descreva o processo de geração de um produto final (como um dado ou até um objeto físico), podendo ser obtida para diferentes finalidades, como a reprodutibilidade, a rastreabilidade e a depuração de processos (HERSCHEL et al., 2017). Dados de proveniência podem incluir informações sobre a origem de artefatos, como o responsável e a data de criação de um conjunto de dados, além de aspectos sobre o *script* e execução dos *pipelines* de AM, como parâmetros utilizados, duração da execução e status das operações realizadas. Por exemplo, ao executar um *pipeline* para classificação de risco de crédito, no qual o rótulo indica se um cliente é um "bom" ou "mau" pagador, os dados de proveniência podem registrar todo o pré-processamento dos dados. Esses registros também podem incluir detalhes do modelo gerado, como algoritmo, parâmetros e atributos utilizados, permitindo acesso a um histórico das execuções realizadas com esse *pipeline* específico.

Considerando o contexto de execução de *pipelines* de AM, dados de proveniência podem ser capturados para ajudar na análise dos resultados obtidos a partir de um modelo de AM e do processo aplicado para identificar fatores que impactaram o desempenho do modelo. Ao relacionar os dados de proveniência com os resultados gerados pelo modelo de AM, também é possível aplicar técnicas que reflitam na melhoria dos resultados dos modelos de AM. Como ilustração, o trabalho realizado por LOURENÇO et al. (2020) utiliza dados de proveniência extraídos de um histórico de execuções com o objetivo de identificar falhas, como a geração de resultados errôneos, e aplicar a otimização de hiperparâmetros utilizados no treinamento de modelos de AM. Ao considerar esse processo iterativo, é possível ajustar e avaliar o modelo a partir de dados de proveniência, utilizando informações sobre a execução do *pipeline* e os resultados obtidos, o que torna os dados de proveniência relevantes para futuras execuções do *pipeline*.

Para estruturar esses dados de proveniência, utiliza-se, em geral, uma ontologia com o objetivo de padronizar a representação, facilitar sua interpretação e reutilização. Entre as ontologias existentes, a mais comumente adotada é a PROV, proposta pelo *World Wide Web Consortium* (W3C) (LEBO et al., 2013). Essa ontologia se destaca por sua flexibilidade na modelagem dos processos, podendo ser estendida para representar de forma mais contextualizada as atividades em domínios específicos de AM, conforme demonstrado por SCHLEGEL & SATTLER (2023), BUTT & FITCH (2020) e SOUZA et al. (2021).

Os dados de proveniência das atividades realizadas na fase de preparação de dados também ajudam a entender o tratamento realizado sobre eles e a possível influência que cada operação exerceu no resultado do modelo preditivo (OLIVEIRA et al., 2024). Em um *pipeline*

de classificação de atrasos de voos, por exemplo, os dados de proveniência podem indicar que a exclusão de atributos com baixa variabilidade ou alta correlação com outros atributos contribuiu para melhorias de métricas de avaliação, como a *F1-score*, nos modelos de AM.

Ao considerar que a melhoria e adequação dos dados realizados na fase de preparação desempenham um papel importante na análise e otimização do processo de treinamento do modelo de AM, torna-se relevante associar essas atividades aos dados de proveniência. Isso se deve ao fato de que eles permitem registrar de forma estruturada todo o processo aplicado para a obtenção dos atributos utilizados nos modelos de AM. Também se torna possível realizar ajustes mais precisos ao longo do tempo, como na identificação de problemas nos dados e o ajuste dos parâmetros das atividades desempenhadas, aprimorando os resultados obtidos a partir dos modelos.

As melhorias observadas a partir do uso dos dados de proveniência entre execuções de um *pipeline* no contexto da fase de preparação de dados ainda necessitam de investigações mais aprofundadas. Além disso, são necessárias avaliações experimentais que comprovem sua efetividade em diferentes cenários e domínios. Isso destaca a necessidade de desenvolver métodos para detalhar as atividades relacionadas à preparação dos dados, como realizado em OLIVEIRA et al. (2024) e CHAPMAN et al. (2020), mas também para analisar e aplicar dados de proveniência, de modo que este uso possa refletir de forma positiva na avaliação dos modelos de AM. Além disso, embora trabalhos na literatura indiquem que os dados de proveniência podem contribuir para o desenvolvimento de modelos de AM, sua aplicação ao longo desse processo ainda não foi plenamente explorada.

No trabalho realizado por SCHELTER et al. (2024), dados provenientes de execuções anteriores são utilizados para identificar automaticamente correções que devem ser realizadas no *pipeline*. De modo semelhante, os dados de proveniência também podem ser utilizados para otimizar atividades específicas executadas ao longo do *pipeline*, como a seleção de atributos. Ao rastrear e correlacionar os atributos mais eficazes em execuções, torna-se possível identificar os atributos que impactam positivamente a avaliação do modelo, o que pode contribuir diretamente para a qualidade dos modelos gerados (OLIVEIRA et al., 2024).

A utilização eficaz dos dados de proveniência, muitas vezes, encontra obstáculos na dificuldade de rastrear como determinados atributos foram selecionados ou descartados ao longo do *pipeline* de AM e como reexecutar essas atividades de forma que a consistência do processo seja mantida. Isso pode, por sua vez, refletir diretamente na avaliação do modelo, afetando as métricas avaliativas, impactando a acurácia, precisão e outras medidas de desempenho essenciais para a qualidade das previsões. O desenvolvimento de métodos

baseados em dados de proveniência pode contribuir significativamente para a documentação e interpretação dos modelos, além de possibilitar ajustes mais precisos e melhorias em diferentes contextos de aplicação, conforme evidenciado em trabalhos como os de OLIVEIRA et al. (2024), SCHELTER et al. (2024) e CHAPMAN et al. (2020).

## 1.2 QUESTÕES DE PESQUISA

Dado o cenário exposto, este trabalho visa compreender como dados de proveniência obtidos na execução de *pipelines* de AM podem ser utilizados para a auxiliar em sua melhoria. Nesse sentido, delineiam-se as seguintes questões de pesquisa:

**Q1:** Como dados de proveniência capturados durante a execução de um *pipeline* de aprendizado de máquina podem ser utilizados para registrar e viabilizar a reexecução consistente de atividades específicas em momentos futuros?

Ao considerar um processo em que os dados de proveniência, obtidos a partir da execução do *pipeline* de aprendizado de máquina, são estruturados semanticamente, é possível acessar o histórico de atividades. Isso proporciona maior compreensão, confiança e reutilização dos dados em diferentes execuções (SOUZA et al., 2021), melhorando a análise e o aprimoramento contínuo dos modelos de aprendizado de máquina. A partir dos dados de proveniência também é possível identificar se, para uma execução atual de um *pipeline* existem execuções anteriores relacionadas, desde que seja utilizado o mesmo *pipeline* e conjunto de dados, o que também contribui para a consistência das atividades que podem ser reexecutadas.

**Q2:** Como as informações obtidas a partir dos dados de proveniência de execuções anteriores de *pipelines* de aprendizado de máquina podem auxiliar na realização de ajustes na seleção de atributos, de forma a contribuir para a melhoria da avaliação dos modelos de AM?

Em um contexto iterativo, novas execuções de um *pipeline* de aprendizado de máquina podem se beneficiar de execuções anteriores, alterando o processo, promovendo a sua reexecução e ajustando-o com base nos dados de proveniência obtidos. Isso se aplica tanto à reexecução de atividades quanto à introdução ou ajuste de atividades, como a seleção de atributos. Embora a literatura explore o uso de dados de proveniência em atividades dos *pipelines* de AM, a aplicação desses dados na seleção de atributos ainda é pouco explorada, o que motiva o foco do presente trabalho nessa atividade. Ao analisar os atributos utilizados em execuções passadas e seu impacto na avaliação do modelo, é possível ajustar e refinar a escolha de atributos em futuras execuções, selecionando aqueles mais relevantes para o modelo, o que tende a melhorar a sua avaliação.

## 1.3 OBJETIVOS

O principal objetivo deste trabalho de doutorado é propor uma abordagem que considere desde a estruturação semântica de dados de proveniência relacionados à execução de *pipelines* de AM até a sua utilização de modo a promover a melhoria dos modelos treinados. Para atingir esse objetivo, definem-se alguns objetivos específicos:

- Estruturar e organizar os dados de proveniência gerados em *pipelines* de AM, utilizando ontologias para promover uma representação semântica que favoreça a interpretação, reutilização e integração com diferentes ferramentas;
- Investigar como os dados de proveniência provenientes de execuções anteriores de treinamento de modelos de AM podem ser analisados para orientar e automatizar ajustes na atividade de seleção de atributos, com ênfase em técnicas como a Eliminação Recursiva de Atributos (RFE, do inglês *Recursive Feature Selection*);
- Propor e desenvolver uma abordagem que, a partir da captura de dados de proveniência, realize a estruturação semântica desses dados, possibilitando sua utilização tanto na reexecução de atividades específicas dos *pipelines* quanto no aprimoramento da seleção de atributos, com o objetivo de dar suporte para uma possível melhoria na avaliação dos modelos gerados; e
- Avaliar a eficácia da abordagem proposta por meio de experimentos, utilizando métricas como acurácia e *F1-score*, comparando os modelos gerados com e sem o suporte dos dados de proveniência.

## 1.4 VISÃO GERAL DA SOLUÇÃO PROPOSTA

Considerando o objetivo deste trabalho, a estruturação semântica dos dados de proveniência relacionados à execução de *pipelines* de AM e seu uso têm como propósito a melhoria dos modelos treinados. Essa estruturação semântica inclui a organização e categorização das informações sobre os artefatos utilizados e os dados obtidos durante a execução do *pipeline* de AM. Assim, todos os dados de proveniência relevantes estarão disponíveis para análise posterior.

Além disso, a realização de ajustes nas atividades a partir dos dados de proveniência pode refletir positivamente nos resultados dos modelos de AM. Por exemplo, ao analisar os modelos de AM em execuções anteriores, torna-se possível identificar quais atributos têm maior influência nos resultados do modelo, possibilitando a seleção dos atributos mais adequados.

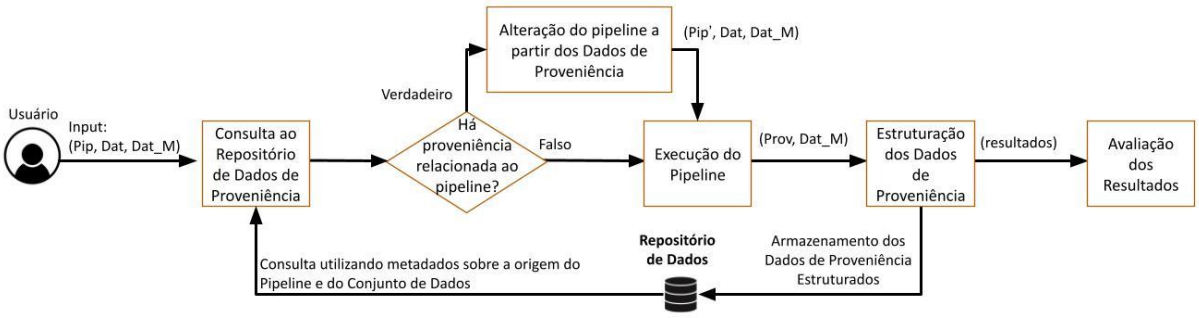
Dessa forma, a solução proposta busca facilitar a interpretação e a transparência dos modelos de AM, além de promover melhorias contínuas com base na análise e na reexecução de atividades bem-sucedidas.

Neste trabalho, a seleção de atributos é realizada a partir da aplicação da técnica de RFE (KUHN & JOHSON, 2013), visando aprimorar o desempenho dos modelos gerados. A RFE foi escolhida por permitir a seleção de um conjunto de atributos de forma automatizada e padronizada, levando em consideração a contribuição dos atributos para o modelo gerado, independentemente de particularidades específicas de cada *pipeline*. Além disso, a escolha pela RFE em vez de outras técnicas também automatizadas, como o Lasso (HASTIE et al., 2015), se dá pelo fato de ser mais adequada para modelos não lineares, permitindo levar em consideração as interações não lineares entre as variáveis, como nos modelos utilizados nos *pipelines* dos experimentos, que envolvem algoritmos de Árvore de Decisão e Random Forest.

A Figura 1 ilustra as etapas de execução da solução proposta, que abrange desde a utilização de dados de proveniência de execuções anteriores para ajustar o *pipeline* de AM até a estruturação semântica dos dados de proveniência da execução atual. O processo inicia a partir do momento em que o usuário fornece o *pipeline* de AM (*Pip*), o conjunto de dados (*Dat*) a ser utilizado e o conjunto de metadados associado a esse conjunto de dados (*Dat_M*).

Na primeira etapa, verifica-se, no repositório de dados, se há alguma execução anterior relacionada à execução atual do *pipeline*. Com base nessa verificação, se houver execuções anteriores desse *pipeline*, os dados de proveniência são utilizados para identificar atributos que possam melhorar os resultados do modelo por meio da aplicação da técnica RFE. A partir da identificação desses atributos mais relevantes, a atividade de seleção de atributos é atualizada, resultando em um *pipeline* ajustado (*Pip'*), que segue para a etapa de execução. Por outro lado, se não houver dados de proveniência relacionados ao *pipeline*, ou se o usuário não desejar alterar o *pipeline* com os dados de proveniência, o fluxo segue para a execução do *pipeline*, sem alterações.

Figura 1 – Etapas de Execução da solução proposta.




Fonte: O Autor.

A partir da execução do *pipeline*, é realizada a captura dos dados de proveniência e, na etapa seguinte, esses dados de proveniência e os metadados do conjunto de dados são estruturados semanticamente de forma automatizada utilizando a extensão da Ontologia PROV do W3C (LEBO et al., 2023), denominada PROVX, proposta neste trabalho. Após esta etapa, a proveniência estruturada semanticamente é armazenada no repositório de dados em formato RDF. Em seguida, é disponibilizada ao usuário para a realização de consultas do tipo SPARQL. Por fim, os resultados da execução atual são avaliados e comparados com os resultados das execuções anteriores, permitindo avaliar os ajustes realizados e os respectivos impactos nos resultados.

## 1.5 METODOLOGIA DE PESQUISA

O tema central desta tese, relacionado ao uso de dados de proveniência em *pipelines* de AM, foi inicialmente investigado por meio de um estudo exploratório. Esse estudo teve como propósito fornecer uma visão das abordagens existentes, possibilitando a identificação de lacunas e limitações nas soluções previamente propostas. Posteriormente, foi realizado um levantamento direcionado sobre os trabalhos que estendem a Ontologia/Modelo PROV no contexto da execução de *pipelines* de AM, com o intuito de compreender as estratégias adotadas e identificar aspectos ainda não explorados. A partir dessa análise, foi possível delimitar o escopo da pesquisa, que se concentra na proposição de uma abordagem que permita a reexecução e o ajuste de atividades de um *pipeline* de AM, utilizando dados de proveniência estruturados semanticamente.

Com base no cenário identificado, foram formuladas duas perguntas de pesquisa (Seção 1.2), as quais nortearam o desenvolvimento da investigação. A pesquisa adota uma metodologia quantitativa experimental, com o objetivo de avaliar, de forma controlada, os efeitos da aplicação de dados de proveniência na atividade de seleção de atributos em *pipelines* de AM. Além disso, foi adotada uma abordagem exploratória, considerando a natureza ainda pouco explorada da aplicação de dados de proveniência na atividade de seleção de atributos em *pipelines* de AM.

A proposta deste trabalho foi estruturada em três fases principais: (i) modelagem conceitual e definição da solução, incluindo a extensão da Ontologia PROV, denominada PROVX, e a estratégia de ajuste baseada em execuções anteriores; (ii) desenvolvimento do protótipo nFlowX, que considera a captura, estruturação e utilização dos dados de proveniência; e (iii) avaliação experimental da abordagem. Para a avaliação, foram definidos cenários e conduzidos experimentos com diferentes conjuntos de dados, conforme descrito na Seção 5. Os

resultados obtidos foram analisados por meio de métricas quantitativas e comparados com cenários de referência (baseline), de modo a evidenciar as contribuições da proposta.

## 1.6 CONTRIBUIÇÕES ESPERADAS

Estas são as principais contribuições desta tese:

- Uma abordagem que, a partir da obtenção de dados de proveniência da execução de *pipelines* de AM, estrutura esses dados considerando elementos essenciais do contexto, como artefatos, atores e o processo aplicado, e os disponibiliza para consulta;
- A definição de uma extensão de ontologia (PROVX) baseada na PROV-O para modelar e gerenciar dados de proveniência em *pipelines* de aprendizado de máquina, considerando as características e o detalhamento de execuções iterativas;
- Uma abordagem para reexecutar atividades com o objetivo de auxiliar na melhoria da avaliação dos *pipelines* de AM;
- Uma prova de conceito, representada por um protótipo (nFlowX), desenvolvido para validar a abordagem proposta; e
- Avaliação da abordagem proposta por meio de experimentos que analisam a melhoria na avaliação dos modelos de AM, resultante do uso de dados de proveniência para a reexecução de atividades e para ajustes na seleção de atributos utilizados no modelo, considerando métricas como acurácia e *F1-score*.

## 1.7 ORGANIZAÇÃO DO DOCUMENTO

O restante deste documento está organizado da seguinte forma:

- O Capítulo 2 detalha os conceitos e técnicas fundamentais que envolvem a abordagem proposta;
- O Capítulo 3 apresenta uma visão geral dos trabalhos relacionados, considerando os trabalhos que exploram a execução de *pipelines* de aprendizado de máquina, com ênfase especial no gerenciamento e uso de dados de proveniência, e os trabalhos que estendem a Ontologia/Modelo PROV do W3C considerando o contexto de execução de *pipelines* de aprendizado de máquina;
- O Capítulo 4 descreve a abordagem proposta, considerando a formalização de algumas definições e cada fase que compõe a abordagem, sendo detalhada a extensão PROVX;
- O Capítulo 5 detalha os experimentos realizados neste trabalho a partir do protótipo nFlowX, sendo avaliados os ganhos que a abordagem proposta promove; e

- O Capítulo 6 expõe as considerações finais acerca do trabalho, considerando as contribuições alcançadas, limitações e trabalhos futuros.

# 2 FUNDAMENTAÇÃO

Neste capítulo, são apresentados os conceitos fundamentais que compõem o escopo deste trabalho. Primeiramente, são introduzidos conceitos sobre aprendizado de máquina e sobre algumas técnicas que podem ser utilizadas no contexto de aprendizado supervisionado. Em seguida, na Seção 2.2, são incluídas definições acerca de dados de proveniência. Na Seção 2.3, são expostas definições e características principais das ontologias e seu uso para representar dados de proveniência. Por fim, são apresentadas as considerações sobre o capítulo.

## 2.1 APRENDIZADO DE MÁQUINA

Na Inteligência Artificial, diversas técnicas permitem a análise de dados para realizar previsões e recomendações, como o Aprendizado de Máquina e o Processamento de Linguagem Natural. O Aprendizado de Máquina, um dos ramos da IA, fornece algumas técnicas, que contribuem para o processo de descoberta de conhecimento por meio de algoritmos capazes de alcançar resultados notáveis mesmo em tarefas mais complexas (OLIVEIRA et al. 2024). A partir desses algoritmos, são gerados modelos que aprendem automaticamente a partir dos dados e podem ser classificados em três tipos principais (LUDERMIR, 2021): Supervisionado, Não Supervisionado e por Reforço. Cada uma dessas abordagens pode ser aplicada de acordo com o cenário apresentado, sendo (ALPPAYDIN, 2010):

- Aprendizado Supervisionado: técnica de aprendizado em que o objetivo é aprender um mapeamento da entrada para uma saída, utilizando dados de treinamento rotulados, cujos valores corretos são fornecidos por um supervisor;
- Aprendizado Não-Supervisionado: essa técnica de aprendizado de máquina utiliza os dados de entrada para identificar os padrões que ocorrem com mais frequência do que outros, utilizando conjuntos de dados não-rotulados; e
- Aprendizado por Reforço: técnica de aprendizado de máquina em que um agente interage com um ambiente, recebendo recompensas ou penalidades por suas ações. O agente aprende a seguir a melhor política a partir de suas tentativas e erros, tendo como objetivo obter a sequência de ações que melhore a recompensa total.

Popularmente utilizados, os algoritmos de AM supervisionado podem ser aplicados em tarefas de classificação (utilizando um rótulo como alvo) ou tarefas de regressão (prevendo valores contínuos a partir de um conjunto de variáveis de entrada) (BURKART & HUBER, 2021). Alguns desses algoritmos induzem modelos interpretáveis e transparentes por sua

natureza simples, por exemplo, regressão logística e árvores de decisão (SCHNEIDER et al., 2023).

Existe uma variedade de algoritmos supervisionados para a classificação, na qual a escolha do algoritmo mais apropriado depende de uma série de fatores, como o objetivo do modelo, o poder de processamento dos dados, a necessidade de interpretação dos resultados e os recursos computacionais disponíveis (BURKART & HUBER, 2021). Alguns desses algoritmos, como Árvores de Decisão e Naive Bayes, são amplamente utilizados devido à sua simplicidade e a sua possível interpretação. Além dessas abordagens tradicionais, também se destaca o *Deep Learning* (Aprendizado Profundo), um subcampo do aprendizado de máquina baseado em redes neurais artificiais com múltiplas camadas (LECUN et al., 2015). Apesar de necessitar de maior poder computacional e geralmente resultar em modelos menos interpretáveis, essa técnica é notável por seu alto desempenho em grandes volumes de dados. Por isso, é amplamente utilizada em tarefas como reconhecimento de imagens e processamento de linguagem natural.

PATIL et al. (2024) detalha alguns dos algoritmos de classificação:

- Árvore de Decisão: Um modelo com estrutura de árvore usado para tomar decisões e prever resultados, dividindo os dados em ramos com base nos valores das características;
- *K-Nearest-Neighbors* (KNN): Um algoritmo simples e não paramétrico utilizado para classificação e regressão, que compara novos pontos de dados com os pontos mais próximos no conjunto de treinamento;
- *Naive Bayes:* Um classificador probabilístico baseado no teorema de Bayes, que assume independência entre os atributos; e
- *Support Vector Machines* (SVM): Um algoritmo de aprendizado supervisionado que encontra uma superfície de decisão (hiperplano) ótima para classificar os pontos de dados em diferentes classes.

Estes algoritmos permitem classificar uma nova entrada em uma das categorias de uma variável Y. Conforme exemplificado na Tabela 1, na qual uma amostra de dados composta por características sociodemográficas de clientes será utilizada para prever o comportamento de um futuro solicitante de crédito com base na variável Y, classificando-o em bom ou mau pagador (SICSÚ et al., 2023).

Tabela 1 - Dados para um problema de classificação

| Cliente | Idade | UF | Resid. | Est. Civil | Instrução | Renda | Y=status |
|---|---|---|---|---|---|---|---|
| 1 | 33 | SP | Própria | Casado | Fundamental | 1200 | Bom |
| 2 | 52 | PA | Própria | Solteiro | Fundamental | 6500 | Bom |
| 3 | 65 | SP | Própria | Solteiro | Superior | 7832 | Mau |
| ... | | | | | | | |

Fonte: SICSÚ et al. (2023)

Para avaliar os resultados dos modelos de aprendizado de máquina, devem ser consideradas métricas que indicam o desempenho dos modelos treinados, como as métricas de acurácia, precisão, cobertura e *F1-score* (ELMRABIT et al., 2020), estabelecidas da seguinte forma:

(i) Acurácia (*Accuracy*) se refere à fração de previsões corretas em relação ao número total de instâncias, conforme fórmula seguinte:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}$$

Onde:

VP são os verdadeiros positivos;

VN são os verdadeiros negativos;

FP são os falsos positivos;

FN são os falsos negativos.

(ii) Precisão (*Precision*) corresponde à fração de previsões corretas entre todas as instâncias classificadas como positivas.

$$Precisão = \frac{VP}{VP + FP}$$

Onde:

VP verdadeiros positivos;

FP falsos positivos.

(iii) Cobertura (*Recall*) é a fração de resultados verdadeiros positivos entre todos os exemplos que realmente são positivos.

$$Cobertura = \frac{VP}{VP + FN}$$

Onde:

VP verdadeiros positivos;

FN falsos negativos.

(iv) *F1-score* representa a média harmônica entre os valores de Precisão e de Cobertura.

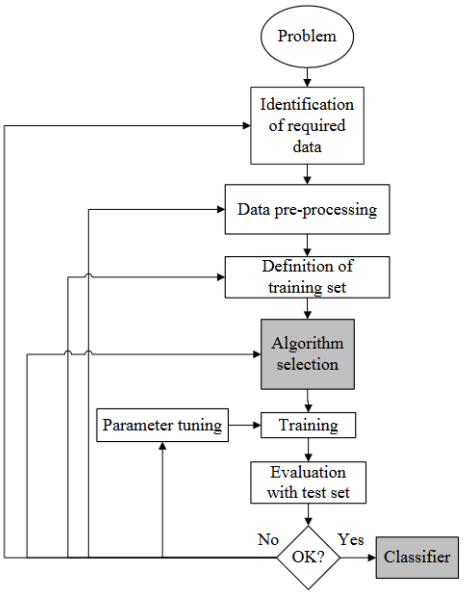
$$F1 - score = 2 * \frac{Precisão * Cobertura}{Precisão + Cobertura}$$

### 2.1.1 *Pipeline* para Treinamento de Modelos de AM

Um *pipeline* de AM consiste em uma série de etapas ordenadas utilizadas para automatizar o *workflow* para o treinamento do modelo (SAMUEL et al. 2021). O desenvolvimento de um *pipeline* para modelos de AM é um processo composto por etapas com o objetivo de treinar o modelo com os dados para que ele seja capaz de prever valores ou classes (CHERAGHI et al., 2021; MARSLAND, 2015). Este *pipeline* deve incluir etapas como: coleta e preparação de dados, treinamento do modelo, teste e avaliação (TEUBL et al., 2023; CHERAGHI et al., 2021), as quais podem ser definidas por metodologias específicas para o desenvolvimento do processo.

Em relação às atividades importantes para o *pipeline* de AM, podemos considerar o trabalho de KOTSIANTIS (2007), no qual é definido um processo de AM para um problema do mundo real (Figura 2). A partir da definição do problema, esse processo inicia com a coleta do conjunto de dados. Em seguida, ocorre a etapa de pré-processamento de dados, na qual são utilizadas técnicas como o tratamento de dados ausentes, detecção de *outliers*, seleção de instâncias e atributos, que permitem reduzir a dimensionalidade dos dados e melhorar a eficácia dos algoritmos.

Figura 2: Atividades *pipeline* de AM




Fonte: KOTSIANTIS, 2007.

Em relação à etapa de modelagem de AM, são realizadas atividades como a definição do conjunto de dados de treinamento (ex.: divisão do conjunto de treinamento utilizando dois terços dos dados e a avaliação do desempenho utilizando o restante dos dados), a escolha do algoritmo e as atividades de treinamento e teste do modelo de AM. A escolha do algoritmo de AM é uma etapa importante desse processo e, a partir da realização de testes preliminares, o classificador poderá ser utilizado ou não.

A avaliação de um classificador normalmente é baseada na acurácia, que é calculada pela porcentagem de predições corretas dividida pelo número total de predições. Se a taxa de erro for insatisfatória, será necessário revisar as etapas anteriores do processo de treinamento do modelo, considerando aspectos como o tamanho do conjunto de dados de treino, a seleção de atributos relevantes e a redução da dimensionalidade, a adequação do algoritmo e o ajuste de parâmetros. A seleção adequada de atributos é fundamental para identificar as variáveis que realmente influenciam os resultados do modelo (HAMDARD & LODIN, 2023). Isso permite eliminar atributos que são irrelevantes ou redundantes, o que melhora a eficiência do modelo e pode reduzir a taxa de erro, concentrando o aprendizado nos atributos mais significativos.

Outros trabalhos também destacam a importância de seguir um conjunto estruturado de etapas ao desenvolver um *pipeline* de AM, visando a eficiência e a precisão do modelo. Segundo TEUBL et al. (2023) e CHERAGHI et al. (2021), um *pipeline* típico deve incluir as etapas que considerem atividades para a coleta e organização dos dados, seguida da limpeza e preparação para remover ruídos e inconsistências dos dados. Após essa preparação, o modelo deve ser treinado com os dados preparados e, em seguida, testado e avaliado para verificar sua eficácia e precisão. Sendo uma abordagem sistemática composta por fases essenciais para transformar os dados brutos em predições úteis.

Entre as áreas relacionadas ao AM, temos a mineração de dados. A mineração de dados é o processo de encontrar padrões nas informações contidas em grandes bases de dados, sendo uma área de pesquisa com interseção com várias disciplinas, incluindo estatística, banco de dados, reconhecimento de padrões e IA (FAYYAD, 2003). Algumas das metodologias estabelecidas e detalhadas para o desenvolvimento de projetos de extração de conhecimento de Mineração de Dados são consideradas e aplicadas dentro do contexto do aprendizado de máquina. O modelo CRISP-DM (CHAPMAN et al. 2000), que, embora tenha sido inicialmente projetado para gerenciar projetos de mineração de dados, teve sua utilidade expandida com a propagação de tecnologias de AM. Estudos como os de SINGH et al. (2022) e PURBASARI et al. (2021), exemplificam o uso do CRISP-DM no desenvolvimento de projetos de AM.

#### *2.1.1.1 CRISP-DM*

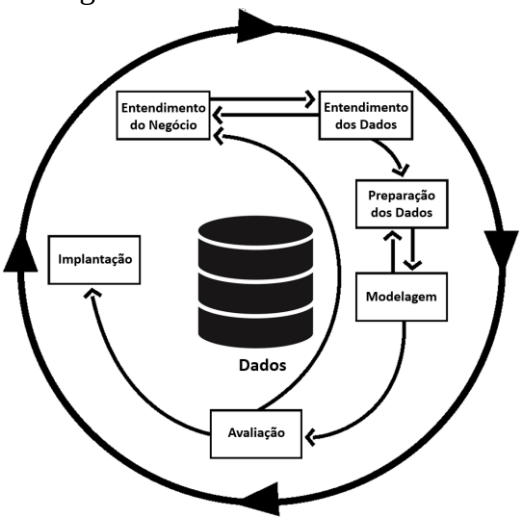
O CRISP-DM (*CRoss-Industry Standard Process for Data Mining*) surgiu em 1996, proposto por um consórcio de três empresas do mercado de mineração de dados, formado pela Daimler AG[1] (na época Daimler-Benz), SPSS Inc.[2] da IBM (na época ISL) e a NCR[3] (CHAPMAN et al. 2000). Esse modelo de processo foi projetado para a indústria, de forma independente de ferramentas e aplicativos, e possui um ciclo de vida para projetos de mineração de dados composto por seis fases (Figura 3), conforme descrito a seguir:

- Entendimento do Negócio: fase que visa definir os objetivos e requisitos do projeto, de forma que, a partir desses objetivos e requisitos, possa ser definido um problema de negócio e um plano para que os objetivos possam ser alcançados;
- Entendimento dos Dados: fase iniciada com a coleta dos dados, passa pela descrição e exploração dos mesmos, permitindo identificar problemas de qualidade e formação de hipóteses sobre informações ocultas;
- Preparação dos Dados: realiza a preparação dos dados que serão utilizados na fase de modelagem, podendo ser realizadas atividades (sem ordem estabelecida) de seleção, limpeza, construção, integração e formatação de dados;
- Modelagem: ocorre a seleção e aplicação de técnicas em busca da extração de conhecimento, como, por exemplo, segmentação, classificação e/ou predição dos dados. Dependendo da técnica utilizada, pode haver requisitos específicos em relação ao formato de dados, sendo possível retornar à fase de preparação de dados, se houver necessidade;
- Avaliação: momento em que é realizada a avaliação detalhada do modelo construído na fase de modelagem, visando identificar se os objetivos de negócio foram alcançados; e
- Implantação: utiliza os resultados da fase de avaliação para determinar uma estratégia de implantação, de forma que fique claro quais ações precisam ser realizadas para o uso dos modelos de conhecimento criados. Essa fase pode resultar na geração de um relatório como um resumo geral ou até mesmo em uma apresentação final e abrangente acerca dos resultados obtidos.

[1] Daimler AG <https://www.daimler.com/en/>
[2] SPSS Inc. <https://www.ibm.com/br-pt/analytics/spss-statistics-software>
[3] NCR <https://www.ncr.com/>

As fases não são obrigatórias, e é possível reexecutar fases que já foram realizadas, tendo em vista que se trata de um processo iterativo. As setas entre as fases do CRISP-DM (Figura 3) indicam as dependências mais importantes e frequentes, e o círculo externo simboliza a natureza cíclica do próprio processo de mineração de dados, em que a solução implantada pode ser utilizada em futuras iterações (CHAPMAN et al., 2000). A Figura 4 apresenta as tarefas genéricas (em negrito) e saídas (em itálico) de cada fase do modelo de referência CRISP-DM.

Figura 3 – Fases do CRISP-DM.




Fonte: Adaptado de CHAPMAN et *al.*, 2000.

Figura 4 – Tarefas e Saídas das Fases do CRISP-DM.

| Entendimento do Negócio | Entendimento dos Dados | Preparação dos Dados | Modelagem | Avaliação | Implantação |
|---|---|---|---|---|---|
| **Determina os objetivos do negócio**<br>-Background<br>-Objetivos de Negócios<br>-Critérios de sucesso empresarial<br><br>**Avalia a situação**<br>-Requisitos de inventário de recursos, suposições e restrições<br>-Riscos e contingências<br>-Terminologia<br>-Custos e benefícios<br><br>**Determina as metas de mineração de dados**<br>-Metas de mineração de dados<br>-Critérios de sucesso de mineração de dados<br><br>**Produz o Plano de Projeto**<br>-Plano de projeto<br>-Avaliação inicial de ferramentas e técnicas | **Coleta os Dados Iniciais** -Relatório inicial de coleta de dados<br><br>**Descreve os dados**<br>-Relatório de descrição de dados<br><br>**Explora os dados**<br>-Relatório de exploração de dados<br><br>**Verifica a qualidade dos dados**<br>-Relatório de Qualidade de Dados | **Seleciona os dados**<br>-Justificativa para inclusão / exclusão<br><br>**Limpa os Dados**<br>-Relatório de limpeza de dados<br><br>**Constrói os Dados**<br>-Atributos Derivados<br>-Registros Gerados<br><br>**Integra os Dados**<br>-Dados mesclados<br>-Dados de formato<br><br>**Formata os Dados**<br>-Dados formatados<br><br>Conjunto de Dados<br>Descrição do conjunto de dados | **Seleciona as técnicas de modelagem**<br>-Técnica de Modelagem<br>-Suposições de modelagem<br><br>**Gera o Design de Teste**<br>- Design de Teste<br><br>**Constrói o Modelo**<br>- Configurações de parâmetro<br>- Modelos<br>- Descrições do modelo<br><br>**Avalia o modelo**<br>-Avaliação de modelo<br>-Configurações de parâmetro revisadas | **Avalia os resultados**<br>- Avaliação dos resultados da mineração de dados escreva Critérios de sucesso empresarial<br>- Modelos Aprovados<br><br>**Processo de revisão**<br>-Revisão do Processo<br><br>**Determina as próximas etapas**<br>-Lista de Decisão de Ações Possíveis | **Plano de implantação**<br>-Plano de preparação<br><br>**Monitoramento e manutenção do plano**<br>-Plano de Monitoramento e Manutenção<br><br>**Produz relatório final**<br>- Relatório final<br>- Apresentação final<br><br>**Revisão do Projeto**<br>- Documentação de experiência |

Fonte: Adaptado de CHAPMAN et *al.*, 2000.

### 2.1.2 Atividade de Seleção de Atributos em *pipelines* de AM

A seleção de atributos em AM é uma técnica de redução da dimensionalidade dos dados que tem o objetivo de selecionar o conjunto de atributos mais relevantes dos atributos originais, removendo atributos irrelevantes, redundantes ou ruidosos (MIAO et al., 2016). Essa atividade é realizada com o objetivo de melhorar o desempenho dos modelos de AM, pois, por meio dela, é possível manter os atributos mais impactantes para o problema de AM em questão (LI et al. 2017). Ao descartar atributos irrelevantes ou redundantes, o treinamento do modelo consegue focar nos atributos mais essenciais, podendo refletir no aumento da métrica de precisão, melhorar o tempo de treinamento e até prevenir o *overfitting*, que comprometeria a capacidade do modelo generalizar e ter um bom desempenho em novos dados.

Entre os métodos utilizados para a seleção de atributos, pode-se destacar (MIAO, et al., 2016; LI et al. 2017):

- Métodos *Filter*: avaliam os atributos de forma independente do modelo de aprendizado, utilizando métricas estatísticas para classificar a importância desses atributos;
- Métodos *Wrapper*: utilizam um modelo de aprendizado específico para avaliar a qualidade de subconjuntos de atributos. Neste caso, é realizada uma busca iterativa para encontrar o subconjunto que melhore o desempenho do modelo, podendo ser computacionalmente caro, devido à necessidade de treinar o modelo várias vezes; e
- Métodos *Embedded*: realizam a seleção de atributos como parte do processo de treinamento do modelo, sendo mais eficiente em termos de tempo e desempenho.

Uma das técnicas amplamente utilizadas em algoritmos de classificação, como Árvores de Decisão e Random Forest, é a Eliminação Recursiva de Atributos, que se classifica como um método do tipo *Wrapper*. Essa técnica realiza a seleção dos atributos a partir da importância de cada atributo dentro do modelo, em que os atributos menos relevantes são removidos um por vez (KUHN & JOHSON, 2013). Nos algoritmos de classificação baseados em árvores de decisão, as importâncias dos atributos são calculadas com base na redução da impureza (como o índice de Gini ou a entropia) durante as divisões realizadas na construção da árvore, refletindo o impacto de cada atributo nas previsões do modelo (SVETNIK et al., 2003).

Para reduzir a quantidade de atributos utilizados no modelo, podemos considerar o seguinte processo (PRIYATNO & WIDIYANINGTYAS, 2024): o modelo de AM é treinado no conjunto de dados, e as importâncias dos atributos são calculadas; os atributos que apresentarem menor importância são eliminados um a um, até que seja alcançada a quantidade desejada de atributos ou então verificar se há uma certa quantidade de atributos que otimizam o desempenho

do modelo. Apesar de haver diversas técnicas para seleção de atributos, a RFE se destaca no uso em algoritmos de classificação devido ao seu processo automatizado e à capacidade de adaptação ao modelo específico, sem a necessidade de intervenção manual, permitindo a identificação otimizada do conjunto de atributos mais relevantes para a tarefa preditiva.

## 2.2 DADOS DE PROVENIÊNCIA

De acordo com BUNEMAN et al. (2001), a proveniência de dados pode ser definida como a "linhagem" ou o "*pedigree*" que descreve a origem de um dado e os processos aplicados sobre o mesmo. A proveniência de dados envolve todo o histórico do dado, desde a sua criação, considerando também a fonte de origem do dado e quem o gerencia, podendo ser utilizada para diversas finalidades, conforme descrito no trabalho de GOBLE (2002):

- Confiabilidade e qualidade: considera informações sobre os processos aplicados ao dado e a sua origem;
- Justificativa e auditoria: utilizam o registro histórico e preciso da fonte e do método utilizado, de forma que seja reprodutível e que também possa identificar possíveis erros ocorridos na geração dos dados;
- Reuso, reprodução e repetição: possibilita a repetição e validação de experimentos, mas deve utilizar as mesmas condições, desde os dados até as ferramentas e algoritmos; e
- Propriedade, segurança, crédito e direitos autorais: se referem à origem e à propriedade do dado, considerando também quando este passar por alterações.

Além da proveniência dos dados, também é possível capturar a proveniência de um fluxo de atividades, como a de uma tarefa computacional, cujos dados de proveniência se enquadram na seguinte classificação (FREIRE et al. 2008, ZHAO et al. 2006):

- Proveniência Prospectiva: considera aspectos de um procedimento ou fluxo de trabalho (ex.: *script* ou fluxo de trabalho) de uma tarefa computacional, sendo as etapas que devem ser seguidas para gerar um produto de dados; e
- Proveniência Retrospectiva: captura as etapas que foram executadas para derivar um produto de dados específico, também captura outras informações de ambiente de forma mais detalhada, como o tempo de execução e recursos utilizados.

Além desses dois tipos de proveniência, no trabalho de HERSCHEL et al. (2017) também é indicada a proveniência do tipo evolutiva, que representa a proveniência em cada execução de um processo iterativo, refletindo alterações que podem ter ocorrido em pelo menos um dos seguintes elementos: dados de entrada, fluxo de trabalho ou contexto de

execução (ex.: parâmetros). Com a captura desse tipo de proveniência é possível comparar as execuções realizadas, o que permite identificar qual delas apresenta os melhores resultados (PIMENTEL et al. 2016).

Os mecanismos que permitem a captura de dados de proveniência em tarefas computacionais podem acessar detalhes relevantes, como as etapas e suas respectivas atividades, informações de execução e anotações especificadas pelo usuário. Entre as ferramentas genéricas que permitem capturar a proveniência a partir da execução de *scripts* temos: Vamsa (NAMAKI et al., 2020), Variolite (KERY et al., 2017) e noWorkflow (PIMENTEL et al., 2017). No entanto, Vamsa não captura informações de tempo de execução e Variolite não disponibiliza a evolução de proveniência. O MLflow[4] é outra ferramenta popular para o gerenciamento de experimentos e captura de dados de proveniência, com foco principalmente em parâmetros, métricas, artefatos e código. Porém, não oferece um rastreamento tão detalhado quanto o noWorkflow[5] (Seção 5.1.1), especialmente no que se refere à captura mais aprofundada dos dados de proveniência.

O noWorkflow permite explorar os dados de proveniência de forma mais completa, considerando informações de tempo de execução e o histórico detalhado das execuções. Além disso, o noWorkflow funciona diretamente com scripts Python, sem necessidade de modificar o código, nem de ambientes específicos para sua execução, sendo uma boa solução para a rastreabilidade e a reprodutibilidade dos experimentos.

## 2.3 REPRESENTAÇÃO DE DADOS DE PROVENIÊNCIA

Nesta seção, são apresentados alguns tópicos referentes à representação dos dados de proveniência, abordando as definições e conceitos acerca de ontologias. Em seguida, a seção é finalizada explorando a modelagem dos dados de proveniência, na qual são detalhados os modelos utilizados para documentar esse tipo de dado.

### 2.3.1 Ontologias

Uma ontologia define uma especificação formal de uma conceitualização compartilhada (BORST, 1997), permitindo assim, descrever conceitos, promover o conhecimento e o compartilhamento dos dados, e sua interpretação por seres humanos ou por máquinas (GRUBER, 1993). As ontologias podem modelar tanto um domínio geral quanto um domínio

[4] https://mlflow.org/
[5] https://github.com/gems-uff/noworkflow

específico, proporcionando a representação do conhecimento, viabilizando a integração de informações de diversas fontes de dados, facilitando a comunicação e a compreensão entre diferentes sistemas e usuários.

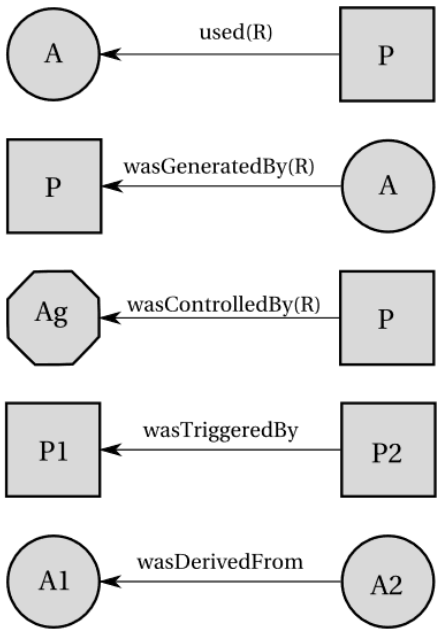
O uso de ontologias permite relacionar as informações por meio de sua estrutura de conhecimento, e estabelecer associações que exigiriam compreensão humana, por meio de classes e propriedades. Com o uso de ontologias é possível estabelecer definições formais de termos e suas equivalências, geralmente utilizando taxonomias e regras de inferência (BERNERS-LEE et. al. 2001). Para as ontologias, estão relacionadas tecnologias como o OWL (*Ontology Web Language*) e o RDF (*Resource Description Framework*). O OWL é a linguagem padrão proposta pelo W3C (2012), utilizada para representar um conhecimento rico e complexo sobre coisas, grupos de coisas e relações entre coisas, sem ambiguidades, permitindo atribuir semântica e definir relações. Já o RDF é o modelo organizado em triplas (sujeito + predicado + objeto) para intercâmbio de dados na Web, fornecendo uma estrutura organizada para os dados, permitindo atribuir significado, e representando e relacionando recursos (W3C, 2014).

Com o uso das ontologias, é possível atribuir significado comum mesmo para termos que sejam de bases diferentes, o que é viabilizado a partir do uso de termos definidos por vocabulários, que permitem estabelecer um padrão para os termos utilizando identificadores únicos (URIs, do inglês *Uniform Resource Identifiers*), que ajudam a padronizar a representação dos conceitos e a reduzir ambiguidades, facilitando o compartilhamento de conhecimento. Um exemplo disso pode ser observado no vocabulário FOAF, identificado pela URI "http://xmlns.com/foaf/0.1/" que utiliza URIs para identificar suas classes e propriedades, como, por exemplo, a propriedade "name" apresenta sua URI correspondente "http://xmlns.com/foaf/0.1/name" (LAUFER, 2015).

### 2.3.2 Modelagem de Dados de Proveniência

Para a representação dos dados de proveniência, existem alguns modelos propostos que são baseados em ontologias e permitem a organização desses dados. Por exemplo, o OPM (*Open Provenance Model*), que é um modelo aberto de proveniência de dados que permite descrever dependências entre artefatos (*Artifact*), processos (*Process*) e agentes (*Agent*) (MOREAU et al., 2011).

Figura 5 – Relacionamentos entre os elementos do OPM.




Fonte: Moreau et al., 2011.

Conforme observado na Figura 5, os artefatos (A), que são objetos usados ou gerados por um processo, os processos (P), que correspondem a ações executadas que consomem e/ou geram artefatos, e os agentes (Ag), que são entidades responsáveis por influenciar a execução dos processos, representam os nós dos grafos. Já as arestas, são dependências (relacionamentos) entre esses nós, que apresentam os seguintes tipos: *used* (artefato usado por um processo), *was generated by* (artefatos gerados por processos), *was triggered by* (processo iniciado por outro processo), *was derived from* (artefato derivado de artefato) e *was controlled by* (processo controlado pelo agente).

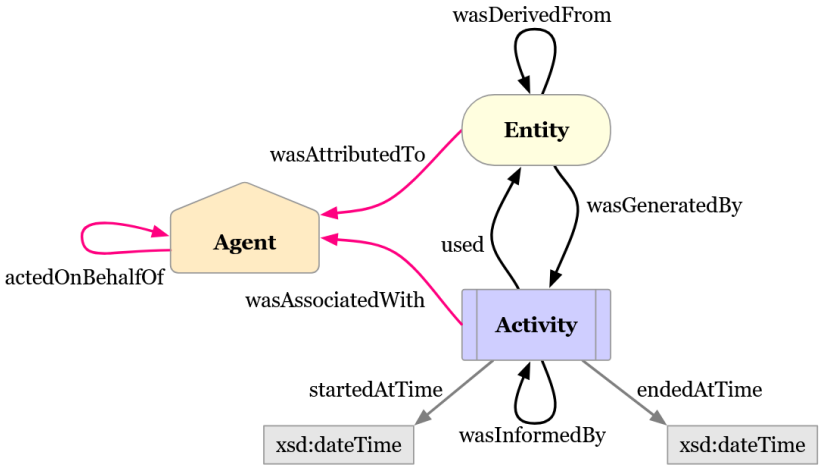
De forma semelhante, a PROV-O (W3C) permite representar a proveniência entre entidades, atividades e agentes, possibilitando que representações de proveniência específicas de domínio e aplicação sejam estabelecidas por meio deste modelo de dados e intercambiadas entre sistemas (Figura 6). No modelo PROV-DM (W3C), utilizado na Ontologia PROV, os dados são representados por um grafo direcionado para que a proveniência possa descrever o uso e a produção de entidades por atividades, as quais podem ser influenciadas pelos agentes (LEBO et al., 2013). Nesse modelo, um tipo pode se referir a uma classe ou subclasse, tendo como base as três classes a seguir:

- Entidade (*Entity*) - representa um objeto (físico/digital/conceitual) com alguns aspectos fixos;
- Atividade (*Activity*) - ocorre durante um período de tempo e atua sobre ou com

entidades; e

- Agente (*Agent*) - possui alguma forma de responsabilidade pela realização de uma atividade.

Figura 6 – Principais Conceitos da Ontologia PROV.




Fonte: W3C, 2013.

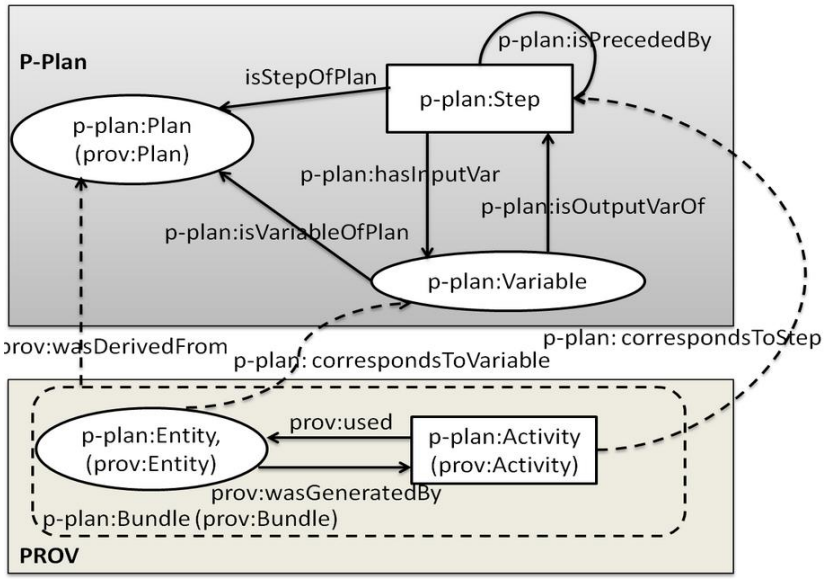
Já para os relacionamentos entre os tipos de elementos, são utilizadas propriedades, como as propriedades a seguir: geração de uma entidade por uma atividade (*WasGeneratedBy*), uso de uma entidade por uma entidade (*Used*), informação sobre uso de entidade (*WasInformedBy*), derivação de entidade (*WasDerivedFrom*), atribuição de uma entidade a um agente (*WasAttributedTo*), associação de uma atividade com um agente (*WasAssociatedWith*), delegação de responsabilidade entre agentes (*ActedOnBehalfOf*), derivação de entidade com objetivo de corrigir erros (*wasRevisionOf*), início de uma entidade já existente por uma atividade (*wasStartedBy*) ou finalização de uma entidade já existente por uma atividade (*wasEndedBy*).

Quando comparada com o OPM, a ontologia PROV possui maior destaque, pois é o modelo recomendado pelo W3C e permite armazenar a proveniência de forma mais detalhada. Essa ontologia contempla relacionamentos equivalentes a todos os relacionamentos apresentados no OPM, e inclui quatro relacionamentos adicionais: *wasInformedBy*, *wasEndedBy*, *actedOnBehalfOf* e *wasAttributedTo* (LEBO et al., 2013). Além disso, esses relacionamentos também permitem especificar responsabilidades e histórico de dados, enquanto o OPM está voltado apenas para o controle de fluxos de execução (BIVAR et al., 2013).

A PROV-O possui várias extensões que ampliam suas capacidades para atender a necessidades específicas. Uma dessas extensões é a ontologia P-PLAN (GARIJO & GIL,

2014), a qual expande a PROV-O para incluir a descrição detalhada dos planos de execução, oferecendo uma representação clara das etapas e ações em fluxos de trabalho complexos (Figura 7). No P-PLAN, um *Plan* é um conjunto de instruções ou ações planejadas, composto de *Steps*, que são as atividades específicas que compõem esse plano. Além disso, o P-PLAN define *Variables*, que são os dados utilizados ou gerados durante a execução de um plano. O P-PLAN utiliza os conceitos de Entidades, Atividades e Agentes da PROV-O para descrever os detalhes de como os processos são planejados e implementados. No P-PLAN, os Steps estão alinhados com as Atividades da PROV-O e *Variables* correspondem às Entidades do PROV-DM, o que permite descrever a sequência de atividades em projetos científicos e tecnológicos com um entendimento mais aprofundado e melhor rastreabilidade das atividades realizadas.

Figura 7 – Visão geral do P-PLAN e como se relaciona com os conceitos do PROV-DM.




Fonte: GARIJO e GIL, 2014.

No exemplo indicado na Figura 8, consideremos um *pipeline* de aprendizado de máquina, e as ontologias PROV-O e P-PLAN são utilizadas para descrever os dados de proveniência referentes ao plano de execução de uma atividade em um *pipeline* de AM. O *ex:TrainingData* corresponde a uma entidade da Ontologia PROV que representa o conjunto de dados utilizado durante o treinamento, enquanto o *ex:TrainedModel* é a entidade que representa o modelo final gerado. A atividade *ex:ModelTraining* descreve o processo de treinamento do modelo, vinculando os dados de entrada ao resultado final. Por outro lado, o ex:TrainingStep se refere a um passo da ontologia P-PLAN que está dentro do *pipeline*, que define o uso dos dados de treinamento como entrada *p-plan:hasInput* e o treinamento do

modelo treinado como saída *p-plan:hasOutput*. A utilização dessas ontologias permite rastrear de forma clara e detalhada as atividades e resultados do *pipeline*, além de estabelecer um vínculo entre o plano teórico do *workflow* e sua execução real, facilitando tarefas como a reprodutibilidade e a auditoria do processo.

Figura 8 – Exemplo de uso da Ontologia PROV em conjunto da ontologia P-Plan.

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix p-plan: <http://purl.org/net/p-plan#> .
@prefix ex: <http://example.org/> .

ex:TrainingData a prov:Entity .
ex:TrainedModel a prov:Entity ;
    prov:wasGeneratedBy ex:ModelTraining .

ex:ModelTraining a prov:Activity ;
    prov:used ex:TrainingData .

ex:TrainingStep a p-plan:Step ;
    p-plan:hasInput ex:TrainingData ;
    p-plan:hasOutput ex:TrainedModel .
```

Fonte: O Autor.

## 2.4 CONSIDERAÇÕES

Neste capítulo, foram apresentados os principais conceitos relacionados à proposta de uma abordagem que considera a estruturação semântica dos dados de proveniência da execução de *pipelines* de AM e à sua utilização. Também foram apresentados conceitos relacionados ao AM e ao modelo de processo para seu treinamento e atividades relacionadas, com destaque para a atividade de seleção de atributos. Além disso, foi discutido sobre dados de proveniência e suas classificações, explorando também a documentação a partir do uso de ontologias e os respectivos modelos para a representação desses dados.

No próximo capítulo, será apresentado um levantamento do estado da arte, considerando trabalhos que estão relacionados com as atividades que compõem a proposta desta pesquisa.

# 3 ESTADO DA ARTE

Este capítulo discute trabalhos existentes na literatura que estão relacionados com as questões de pesquisa. Neste cenário, são abordados estudos que se enquadram em duas principais categorias: (Seção 3.1) Extensão da Ontologia/Modelo PROV para Dados de Proveniência em Execução de *Pipelines* de Aprendizado de Máquina, e (Seção 3.2) Uso de Dados de Proveniência derivados da Execução de *Pipelines* de Aprendizado de Máquina. Na Seção 3.3, é realizada uma análise comparativa dos trabalhos relacionados, considerando algumas características importantes para esta tese, e na Seção 3.4, são tecidas algumas considerações sobre o capítulo.

## 3.1 EXTENSÃO DA ONTOLOGIA/MODELO PROV PARA DADOS DE PROVENIÊNCIA EM EXECUÇÃO DE PIPELINES DE APRENDIZADO DE MÁQUINA

Nesta seção, são discutidos trabalhos que estendem a Ontologia/Modelo PROV do W3C, considerando o contexto de execução de *pipelines* de aprendizado de máquina. Esses trabalhos foram analisados com o objetivo de entender como as ontologias/modelos foram utilizados para organizar e gerenciar as informações, além de explorar suas aplicações nos cenários apresentados. Consideram-se, nesta seção, as contribuições dessas pesquisas para o campo, bem como as principais questões e desafios enfrentados.

### 3.1.1 SCHLEGEL & SATTLER (2023)

No trabalho de SCHLEGEL & SATTLER (2023), foi apresentada uma abordagem que propõe um modelo de dados de proveniência que é uma extensão do modelo PROV-DM. Nesse modelo, são mapeadas as atividades de sistemas de versionamento de código (e.g., GitHub) e de sistemas de gerenciamento de experimentos de aprendizado de máquina (e.g., MLFlow). A proposta é implementada por meio da ferramenta MLflow2PROV, que realiza a extração de grafos de proveniência de acordo com o modelo proposto, baseado no PROV-DM, e possibilita a pesquisa, a análise e o processamento futuro das informações de proveniência obtidas.

No contexto do desenvolvimento de modelos de AM, devem ser considerados artefatos, como códigos de experimentos, modelos treinados e metadados associados. O gerenciamento desses elementos deve seguir uma abordagem estruturada e organizada, utilizando sistemas específicos de gerenciamento de experimentos, como o MLflow (Zaharia et al., 2018). No entanto, esses sistemas geralmente não fornecem a documentação completa sobre os dados de

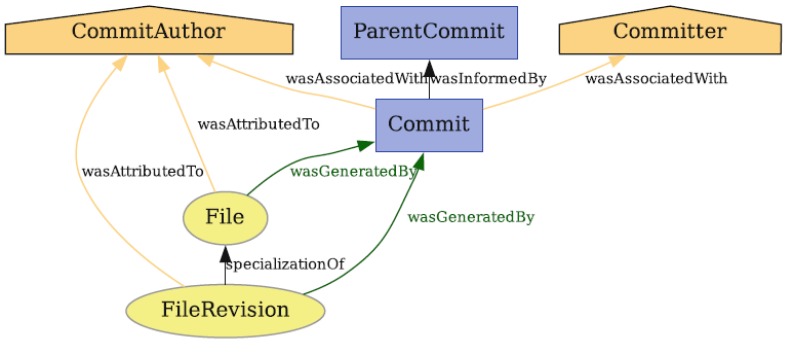
proveniência dos experimentos, incluindo todas as atividades relacionadas à execução, como entidades (arquivos de dados, modelos treinados), atividades (execuções de *pipelines*, treinamentos de modelos, tarefas de processamento de dados) e agentes (usuários responsáveis pelas mudanças, times de cientistas de dados). Além disso, nesses sistemas os dados de proveniência nem sempre são documentados em um formato interoperável, como o modelo PROV-DM (garantindo que dados e metadados sejam compreendidos e utilizados de forma consistente), o que limita o uso desses dados por outras ferramentas. A proposta do trabalho é extrair automaticamente a proveniência dos sistemas de gerenciamento de experimentos de AM e dos sistemas de gerenciamento de versões, criando grafos de proveniência a partir desses dados. As principais contribuições incluem um modelo de proveniência baseado nas atividades capturadas durante os experimentos e uma ferramenta que extrai e disponibiliza essa proveniência no modelo proposto.

Nesse contexto, são considerados como entidades os arquivos, experimentos, execuções e qualquer outro artefato e suas inter-relações. Isso resulta em um grafo direcionado acíclico, com seus nós e arestas definidos semanticamente, permitindo descrever o significado, a função e o papel de cada elemento no *pipeline* de AM. O modelo proposto utiliza a notação do PROV-DM e é composto por vários submodelos, considerando atividades relacionadas aos sistemas de gerenciamento e sistemas de versionamento. Seu objetivo é viabilizar a recuperação de informações relevantes, respondendo a questionamentos como: quais parâmetros foram utilizados no treinamento da versão mais recente do modelo X? Ou ainda, qual agente foi responsável pelo modelo com melhor avaliação e quem contribuiu mais para aquele modelo?

Os submodelos, que compõem a proposta, podem ser categorizados em dois grupos:

- Relacionados ao GitHub: Captura os efeitos dos C*ommits* realizados no Github sobre os estados dos arquivos e conteúdo, como: adição, alteração e exclusão de um arquivo. Como exemplo, tem-se a adição de um novo *File (*arquivo) (Figura 9), na qual o *File* representa uma entidade, o *Commit* corresponde à atividade, o *CommitAuthor e o Comitter* são os agentes envolvidos na realização deste C*ommit*; e

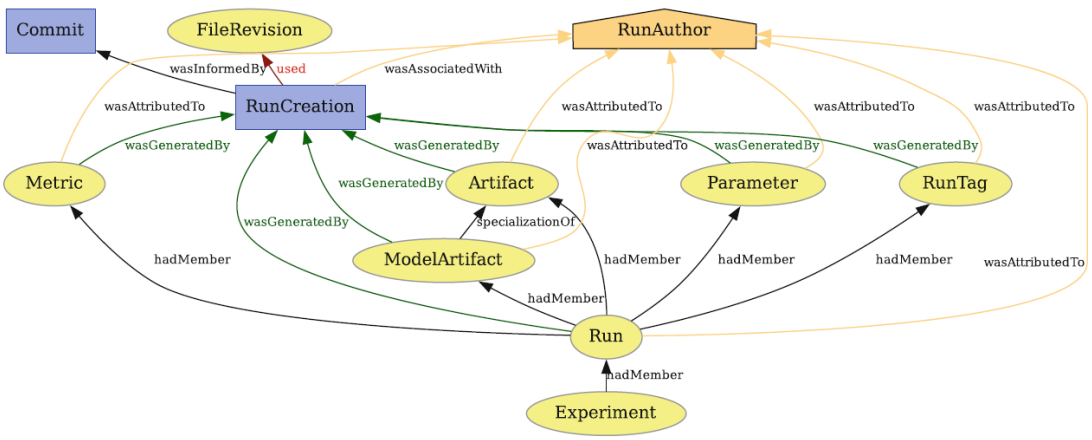
Figura 9 - Submodelo representando a criação de um arquivo.




Fonte: SCHLEGEL & SATTLER, 2023.

- Relacionados ao MLFlow: Realiza o rastreamento dos componentes que são organizados considerando os conceitos das execuções do *script* de AM, essas execuções podem ser organizadas em experimentos que agrupam execuções pertencentes à mesma tarefa. Neste submodelo, são capturados dados de proveniência de atividades como: adição de novo experimento e de nova execução; e a exclusão de execuções e experimentos. Essas atividades apresentam um conjunto de entidades, como, por exemplo, a atividade de criação de execução (*RunCreation)* (Figura 10), que ocorre com a execução de um *Experiment* considerando um modelo de treinamento, sendo possível identificar entidades importantes, como a própria execução (*Run*), métricas (*Metric*), parâmetros (*Parameter*), *tag* de execução (*RunTag)*, artefatos gerados (*Artifact*) e o modelo do artefato (*ModelArtifact).*

Figura 10 - Submodelo representando a criação de uma execução.

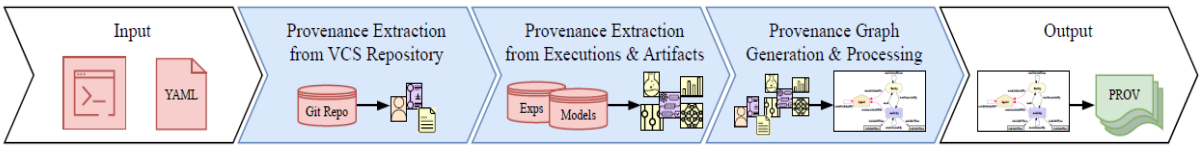



Fonte: SCHLEGEL & SATTLER, 2023.

A implementação da proposta foi realizada em Python, utilizando bibliotecas para acessar os sistemas do GitHub e o MLFlow. O processo de extração do grafo de proveniência (Figura 11) a partir do Github ou de uma instância do MLFlow segue as seguintes etapas (destacadas em azul na figura):

- Primeiramente, é obtida a proveniência do Github considerando elementos como: o *commit*, os arquivos e os usuários;
- Em seguida, é obtida a proveniência a partir do sistema de gerenciamento de experimento, de onde são obtidos metadados, como parâmetros, métricas e modelos; e
- Após isso, os dados são estruturados utilizando o modelo proposto, sendo gerados grafos de proveniência que podem ser processados em sistemas específicos de armazenamento de dados de proveniência, a exemplo do ProvStore (HUYNH & MOREAU, 2015), ou sistemas de banco de dados de grafos, como o Neo4J[6].

Figura 11 - Processo de extração de grafo de proveniência.




Fonte: SCHLEGEL & SATTLER, 2023.

A proposta foi avaliada por meio da implementação (MLflow2PROV), considerando sua capacidade de capturar a proveniência completa dos *pipelines* de experimentos. Todas as atividades de desenvolvimento e experimentos foram incluídas no repositório de código-fonte e na instância do sistema de gerenciamento de experimentos. A capacidade de captura foi medida por meio da realização de consultas a partir dos dados de proveniência obtidos. Isso permitiu responder a questionamentos, como determinar o modelo que apresenta melhor desempenho na métrica de acurácia, a partir dos dados de proveniência armazenados, que são filtrados e ordenados de acordo com a métrica de interesse. Além disso, a proposta foi avaliada quanto à sua capacidade de exportar a documentação de proveniência em um formato interoperável, como a PROV-O. Também foi possível verificar a geração de grafos de proveniência compatíveis com o padrão PROV do W3C, o que permite o uso de ferramentas compatíveis com esse padrão para processamento, análise e visualização adicionais.

[6] https://neo4j.com/

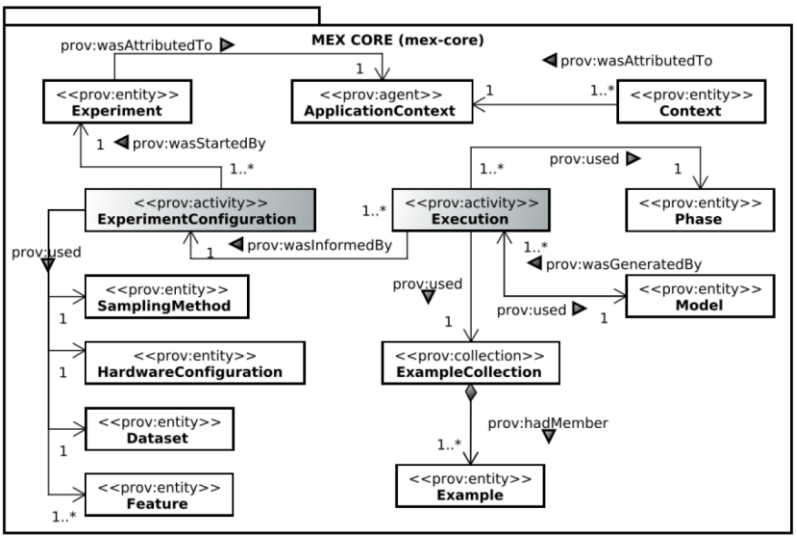
### 3.1.2 ESTEVES et al. (2015)

Nesse trabalho, é discutido um formato padrão para a disponibilização dos metadados provenientes de experimentos de aprendizado de máquina, sendo destacada a necessidade de disponibilização desses metadados em um formato legível por máquina e a grande quantidade de dados gerados durante esses experimentos, o que torna a sua análise desafiadora.

O vocabulário proposto por ESTEVES et al. (2015), chamado MEX, reutiliza as classes e propriedades do PROV-DM, porém apresenta uma estrutura voltada para a representação e compartilhamento de dados de proveniência de experimentos de AM. Sendo composto por três partes principais:

- MEX-Core (Figura 12): Estabelece as entidades essenciais para representar passos básicos em execuções de aprendizado de máquina, incluindo informações sobre o contexto do experimento, a configuração do hardware, e a atividade de execução propriamente dita, as quais são relacionadas utilizando propriedades do PROV-DM, como *prov:used* e *prov:wasAttributedTo*;

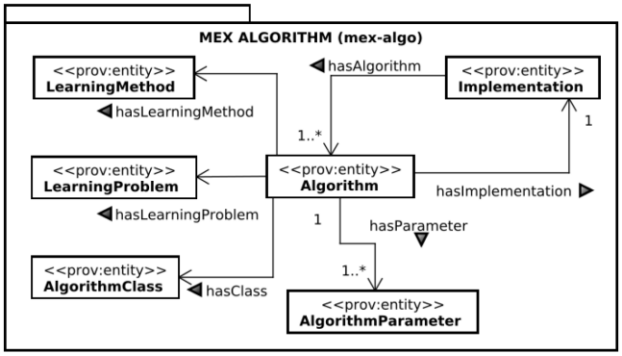
Figura 12 - MEX-Core




Fonte: ESTEVES et al., 2015.

- MEX-Algorithm (Figura 13): Representa os métodos de AM, considera o tipo de aprendizado (supervisionado, não supervisionado ou por reforço), o algoritmo e os parâmetros dos algoritmos. Estes componentes são associados por meio das propriedades *hasAlgorithm* e *hasParameter*, que relacionam os parâmetros às suas respectivas configurações e facilitam a troca de informações e a reprodutibilidade dos experimentos; e

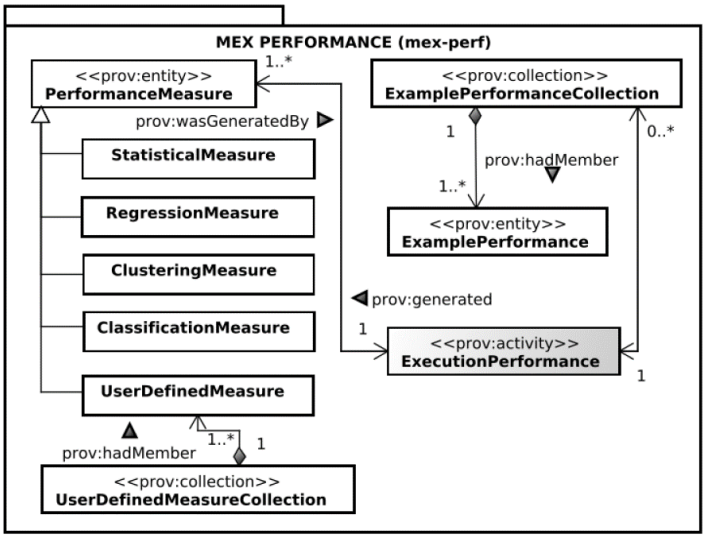
Figura 13 - MEX-Algorithm




Fonte: ESTEVES et al., 2015.

- MEX-Performance (Figura 14): Refere-se ao desempenho dos algoritmos em experimentos de AM. Considera as métricas configuradas para avaliar a eficácia dos modelos, como precisão, recall, *F1-score*. A associação entre a execução do algoritmo e as métricas utilizadas é realizada por meio de propriedades como *prov:wasGeneratedBy* e *prov:hadMember*, garantindo a padronização do registro dos dados de desempenho.

Figura 14 - MEX-Performance




Fonte: ESTEVES et al., 2015.

De forma geral, o objetivo é facilitar a descrição dos experimentos, considerando os dados de proveniência e a manutenção desses experimentos a longo prazo. O vocabulário foi avaliado em casos de uso práticos, considerando experimentos reais de aprendizado de máquina. Esses experimentos demonstraram sua facilidade para uso e implementação, permitindo que pesquisadores documentassem os experimentos de AM. Um desses casos de

uso envolveu a realização de várias atividades de classificação para avaliar ofertas em contratos de trabalho. Nesse contexto, o vocabulário permitiu organizar as execuções e avaliações dos modelos, estruturando as execuções em diferentes configurações e fases (treinamento, teste e validação), além de associar os modelos às suas métricas de desempenho de forma padronizada.

A partir da análise comparativa e do *feedback* dos usuários, foi indicado que o uso do vocabulário melhorou a interoperabilidade, no sentido de possibilitar a troca de informações entre diferentes plataformas e ferramentas de aprendizado de máquina. Além disso, o vocabulário contribuiu para a reprodutibilidade dos experimentos, ao registrar de forma precisa os parâmetros, dados, configurações e resultados. Permitindo que outros pesquisadores possam repetir os experimentos sob as mesmas condições.

### 3.1.3 BUTT & FITCH (2020)

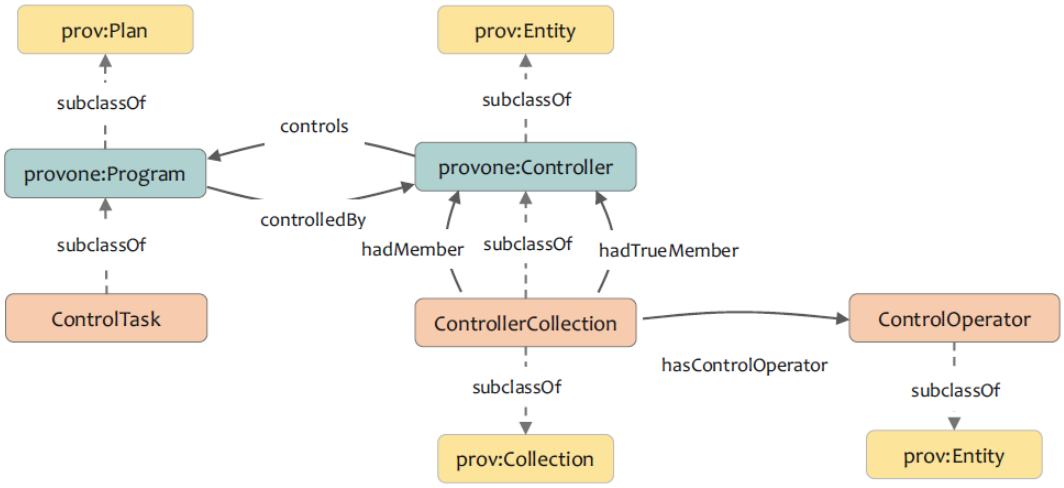
Os autores apresentam o ProvOne+, uma extensão do modelo ProvOne[7] e do modelo PROV-DM, sendo o mesmo modelo utilizado na Ontologia PROV. O objetivo dessa extensão é representar e gerenciar a proveniência em *workflows* científicos. Com isso, é possível especificar e rastrear o histórico de execução considerando detalhes como dados de entrada, resultados obtidos e o processo executado, promovendo a reusabilidade e reprodutibilidade dos resultados.

Nesse trabalho, são introduzidos conceitos, como *Workflow*, *Process* (processo), *Data* (dados) e *Execution* (execução), além de classes como *ControlTask*, *ControllerCollection* e *ControlOperator*, esses conceitos são utilizados para especificar comportamentos de controle de *workflows*, permitindo descrever de forma mais detalhada suas etapas e componentes. A partir do uso de classes do PROV-DM e ProvOne, as tarefas de *workflows* são representadas pela classe *Program* que pode ser simples ou composta, com entradas e saídas conectadas por *Channels*. A classe *Controller* gerencia a execução dos programas e as execuções são registradas pela classe *Execution*, rastreando entradas e saídas de dados, visualizações ou documentos, e obtendo a proveniência do *workflow*. Dados no *workflow* são representados pelas classes *Data*, *Visualization* e *Document*, e agrupados pela classe *Collection*, que organiza conjuntos de itens.

[7] http://jenkins-1.dataone.org/jenkins/view/Documentation%20Projects/job/ProvONE-Documentation-trunk/ws/provenance/ProvONE/v1/provone.html

As classes desta extensão se referem aos controladores, que permitem expressar o controle do fluxo do *workflow* de maneira mais detalhada. Esses controladores, indicados na Figura 15, são responsáveis por definir e gerenciar estruturas de condições, bifurcações e laços de repetição dentro do processo, facilitando a execução e o direcionamento das atividades de acordo com as regras estabelecidas. Essas classes se integram com outras do modelo PROV-DM, como *Entity*, *Plan* e *Collection*, e do ProvOne, como *Program* e *Controller*, ampliando a capacidade de rastrear a proveniência e a execução das atividades, garantindo interoperabilidade e transparência no *workflow*.

Figura 15 – ProvONE+: classes mais importantes e seus relacionamentos.




Fonte: BUTT & FITCH, 2020.

O modelo proposto (ProvONE+) foi avaliado por meio de uma análise comparativa com o modelo ProvONE, na qual foram identificadas limitações deste último, sobretudo no que se refere à representação da estrutura de *workflows*. Essa análise evidenciou a necessidade de um modelo mais expressivo, reforçando a validade da proposta apresentada. Com o detalhamento das interações e dependências entre tarefas promovido pelo ProvONE+, foi constatado que é possível melhorar a reprodutibilidade e a reusabilidade dos experimentos. Além disso, esse detalhamento facilita a auditoria em contextos regulamentados, promovendo também a transparência e eficiência nas tarefas realizadas.

### 3.1.4 SOUZA et al. (2021)

Nesse trabalho, são apresentadas estratégias relacionadas a dados de proveniência de *workflows* para melhorar a análise de dados em AM científicos, considerando os desafios de

heterogeneidade e complexidade desses dados. Também é proposto um modelo de ciclo de vida que integra essas análises de maneira abrangente, considerando as seguintes fases principais:

- Curadoria de Dados: realização da limpeza e preparação de dados brutos, onde são utilizadas ferramentas especializadas para transformá-los em dados úteis e compreensíveis;
- Preparação de Dados para Aprendizado: seleção e transformação dos dados curados em conjuntos de dados de aprendizado, considerando características específicas do domínio; e
- Aprendizado: realização das atividades de treinamento, validação e avaliação do modelo.

Em relação aos dados de proveniência obtidos, são consideradas informações sobre transformações, parâmetros e resultados de modelos. Para a representação desses dados, é utilizado o PROV-ML, que amplia o padrão PROV-DM (W3C) e incorpora elementos do ML *Schema* (W3C). O ML *Schema* é um modelo de dados que permite representar os algoritmos de aprendizado de máquina e as tarefas realizadas, considerando também implementações, execuções, dados de entrada e os modelos gerados.

No PROV-ML são considerados os seguintes conceitos principais:

- *Learning Data*: refere-se aos conjuntos de dados que são preparados e utilizados durante o processo de aprendizado de máquina, podendo ser divididos em: Dados Brutos, Dados Curados; Conjuntos de Dados de Treinamento, Conjuntos de dados de Validação e Teste;
- *Learning*: considera o treinamento, validação e avaliação do modelo de aprendizado de máquina; e
- *Model*: corresponde à representação matemática ou computacional que resulta do processo de aprendizado, podendo ser de diferentes tipos, como: Modelos de Regressão e Modelos de Classificação.

O PROV-ML é composto por várias classes que são fundamentais no ciclo de vida do aprendizado de máquina, entre essas classes que compõem o modelo temos:

- *Study*: Representa a investigação ou hipótese que orienta o *workflow* de aprendizado de máquina;
- *LearningExperiment*: Conjunto de análises que impulsionam o *workflow*;
- *LearningProcessExecution*: Execução de um *workflow* de aprendizado, incluindo

detalhes sobre como os dados foram processados;

- *LearningTask*: Define a tarefa do processo de aprendizado, como classificação ou regressão; e
- *BaseLearningStage*: Classes que representam as etapas do aprendizado (treinamento, validação e avaliação).

A abordagem foi avaliada considerando um estudo de caso na indústria de petróleo e gás. Os resultados indicaram que o sistema demonstrou ser eficiente na captura de dados de proveniência. O sistema possui capacidade de registrar detalhadamente as transformações e interações nos *workflows* ao longo do ciclo de vida de AM, facilitando análises que conectam dados brutos a modelos aprendidos.

## 3.2 USO DE DADOS DE PROVENIÊNCIA DERIVADOS DA EXECUÇÃO DE PIPELINES DE APRENDIZADO DE MÁQUINA

Nesta seção, são descritos trabalhos que exploram a execução de *pipelines* de aprendizado de máquina, com ênfase especial no uso de dados de proveniência. Esses estudos investigam como a rastreabilidade e a documentação das etapas do processo de desenvolvimento de modelos preditivos podem ser aprimoradas para promover um processo transparente e compreensível. Esse aprimoramento reflete na melhoria da análise dos resultados e na geração de conhecimento a partir desses processos.

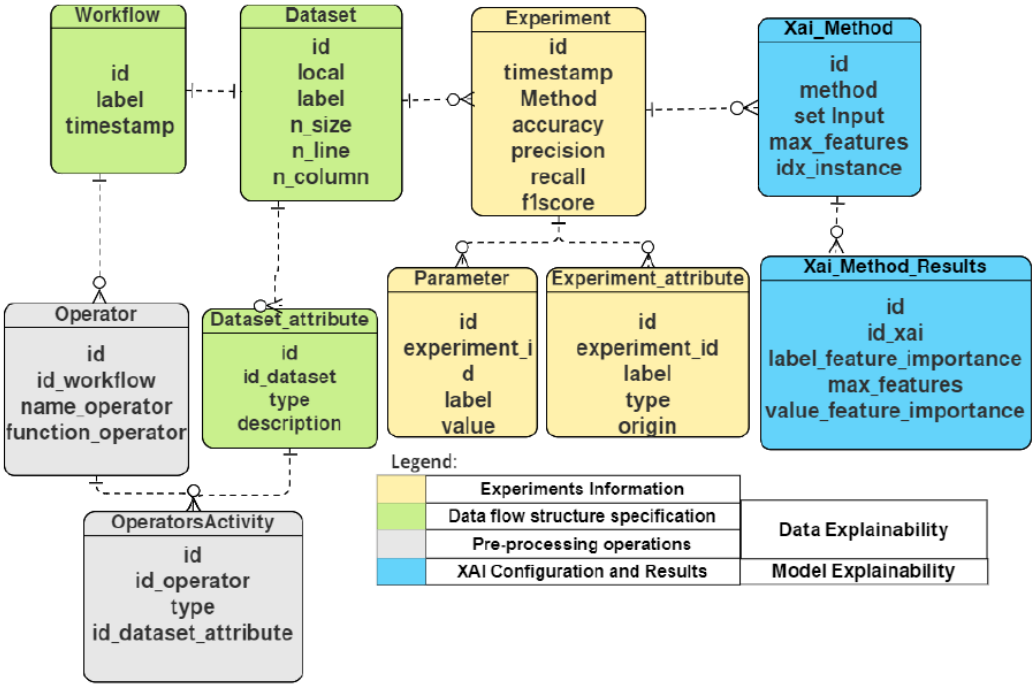
### 3.2.1 OLIVEIRA et al. (2024)

O trabalho de OLIVEIRA et al. (2024) promove a explicabilidade de modelos por meio de dados de proveniência obtidos a partir de transformações dos dados. O trabalho utiliza técnicas de Inteligência Artificial Explicável (XAI, do inglês *Explainable Artificial Intelligence*) para encontrar atributos relevantes para o modelo e indicar as transformações realizadas sobre esses dados por meio dos dados de proveniência, melhorando a explicabilidade dos modelos de AM.

Os dados de proveniência são obtidos a partir da execução de *pipelines* de AM, considerando especialmente a fase de pré-processamento dos dados. Os dados de proveniência, referentes às transformações realizadas sobre os atributos utilizados no modelo de AM, são utilizados para explicar melhor os resultados obtidos pelo modelo de AM. O objetivo é entender o tratamento dos dados, como as atividades de preparação de dados, e sua influência nos resultados dos modelos.

Para a organização dos dados de proveniência, foi especificado um modelo lógico de dados baseado no paradigma relacional, utilizando a notação *Cross Foot* (Figura 16). Cada cor utilizada na especificação das entidades representa uma categoria de dados relacionada ao *workflow*, sendo: verde para informações básicas do *workflow* e do conjunto de dados, cinza para atividades de pré-processamento, amarelo para informações dos experimentos realizados, e azul para a configuração e resultados do XAI.

Figura 16 - Modelo de Dados especificado em OLIVEIRA et al. (2024).




Fonte: OLIVEIRA et al., 2024.

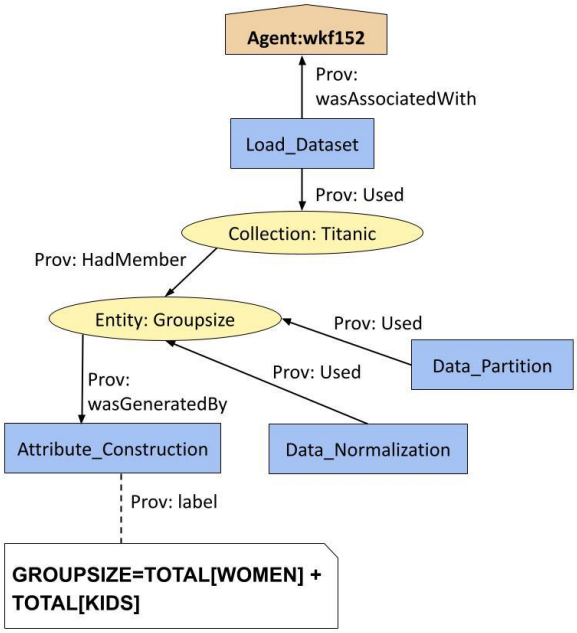
Além da especificação das entidades, as operações realizadas sobre os atributos também ficam disponíveis graficamente. Conforme observado na Figura 17, a representação dos dados de proveniência utiliza como base o modelo PROV-DM do W3C, o qual indica as atividades relacionadas aos atributos, no qual cada atributo é representado como uma entidade. Nesse exemplo da Figura 17, o atributo *Groupsize*, da coleção de dados *Titanic*, é associado a um agente e foi gerado a partir da soma do total de mulheres e crianças, sendo utilizado nas atividades de divisão de dados e normalização.

A proposta para a obtenção e representação das transformações aplicadas sobre os atributos mais influentes de modelos supervisionados foi avaliada por meio de experimentos, nos quais os dados de proveniência foram relacionados às explicações dos resultados dos

modelos, sendo demonstrado que um maior entendimento sobre a derivação dos atributos melhorava a confiabilidade e a clareza das previsões.

Como principal limitação do trabalho, destaca-se a complexidade das operações de pré-processamento, que envolvem técnicas variadas, etapas encadeadas e decisões contextuais nem sempre registradas de forma explícita. Essa complexidade dificulta a captura completa dessas etapas e a interpretação de seu impacto nos atributos do modelo. Como contribuições do trabalho é possível considerar a integração dos dados de proveniência com técnicas de XAI, o que fornece mais detalhes acerca dos atributos importantes para o modelo gerado, promovendo o aumento da confiança nos resultados.

Figura 17 - Visualização das operações realizadas sobre o atributo *Groupsize*.




Fonte: Adaptado de OLIVEIRA et al., 2024.

### 3.2.2 KERZEL et al. (2021)

No artigo de KERZEL et al. (2021), é considerado o ambiente do JupyterLab[8], que permite o desenvolvimento interativo e flexível para execução de códigos e a visualização de dados. Nesse contexto, foi criada uma extensão (MLProvLab), que visa o rastreamento e a gestão de dados de proveniência de *pipelines* de aprendizado de máquina. O trabalho também aborda o aspecto da reprodutibilidade de experimentos de AM. Isso é importante porque, à medida que o uso de algoritmos de AM se expande, a dificuldade em reproduzir resultados se torna um

[8] https://jupyterlab.readthedocs.io/en/latest/

desafio significativo, devido à falta de documentação adequada sobre os dados utilizados, transformações aplicadas e parâmetros dos modelos.

Os dados de proveniência capturados pelo MLProvLab abrangem uma variedade de informações relacionadas à execução dos *pipelines* de aprendizado de máquina, como parâmetros utilizados em cada etapa e os resultados obtidos. Também são registrados detalhes sobre fontes de dados, transformações aplicadas, configurações do ambiente de execução e o contexto geral do experimento. Isso permite, entre outras funcionalidades, a comparação entre execuções, aspecto fundamental para a reprodutibilidade, pois viabiliza a restauração e revisão de alterações feitas em um projeto, incluindo código, dados e configurações. Conforme observado na Figura 18, é possível identificar modificações no código, com destaque para os elementos alterados em relação à versão anterior.

A abordagem proposta foi aplicada a partir do desenvolvimento do MLProvLab, para a captura e gerenciamento dos dados de proveniência de experimentos de AM em ambientes de *notebook*, como o JupyterLab. Entretanto, é uma ferramenta ainda em desenvolvimento, com funcionalidades parcialmente implementadas, sendo indicada a futura realização de testes e levantamentos de *feedback* de usuários. Como contribuições do trabalho, destaca-se a extensão proposta, que facilita a reprodução dos experimentos por meio do rastreamento dos dados de proveniência. Além disso, promove um ambiente colaborativo a partir do compartilhamento de *notebooks* com dados de proveniência, viabilizando também o melhor entendimento acerca das decisões tomadas durante o desenvolvimento do modelo de AM.

Figura 18 - Comparação entre execuções realizadas pelo MLProvLAb.


Fonte: KERZEL et al., 2021.

### 3.2.3 SCHELTER et al. (2023)

O trabalho apresentado por SCHELTER et al. (2023) permite a captura de dados de proveniência gerados durante a execução dos *pipelines* de aprendizado de máquina. A partir desses dados, é possível detectar antecipadamente problemas no *pipeline*, como, por exemplo, o vazamento de dados entre conjuntos de dados de treinamento e teste, que ocorre quando dados do conjunto de teste são utilizados para treinar o modelo. Além disso, podem ser identificados erros de rotulagem e violações de equidade, que podem gerar desigualdades em seus resultados com base em características como raça, gênero e idade. Essa detecção antecipada busca melhorar a qualidade e a confiabilidade dos modelos de AM, promovendo uma abordagem mais responsável e eficiente.

Os dados de proveniência são utilizados para realizar uma análise detalhada das interações entre entradas e saídas nos *pipelines* de AM, considerando um contexto de ajustes contínuos. Esse tipo de atividade permite que os cientistas de dados revisem cada etapa e identifiquem onde erros ou inconsistências no modelo podem ter ocorrido, como manipulações inadequadas sobre os dados que afetam a sua qualidade.

A abordagem foi avaliada considerando os seguintes cenários: (i) cenário relacionado a um *pipeline* de visão computacional, em que erros de rotulagem foram identificados e corrigidos, nos quais imagens de botas foram rotuladas como imagens de tênis e vice-versa; (ii) cenário que considera um *pipeline* de previsão de preços utilizando modelos de regressão, no qual foram detectados vazamentos de dados, em que várias amostras de treinamento também foram acidentalmente incluídas no conjunto de teste; e (iii) cenário aplicado em um *pipeline* de pontuação de crédito, em que foi analisada a equidade das previsões e foram identificadas disparidades entre grupos demográficos. O sucesso nas tarefas desempenhadas consistiu na detecção e correção proativa desses problemas (erros de rotulagem, vazamento de dados e violações de justiça) antes da implantação dos modelos. Esse resultado demonstrou a eficácia da abordagem na antecipação de falhas críticas nos *pipelines*, permitindo intervenções mais rápidas e assertivas ao longo do ciclo de desenvolvimento.

### 3.2.4 NAMAKI et al. (2020)

No trabalho de NAMAKI et *al.* (2020), é apresentado o sistema Vamsa, que permite rastrear os dados de proveniência de atributos de modelos de aprendizado de máquina. O sistema tem como objetivo principal rastrear automaticamente quais atributos de um conjunto de dados

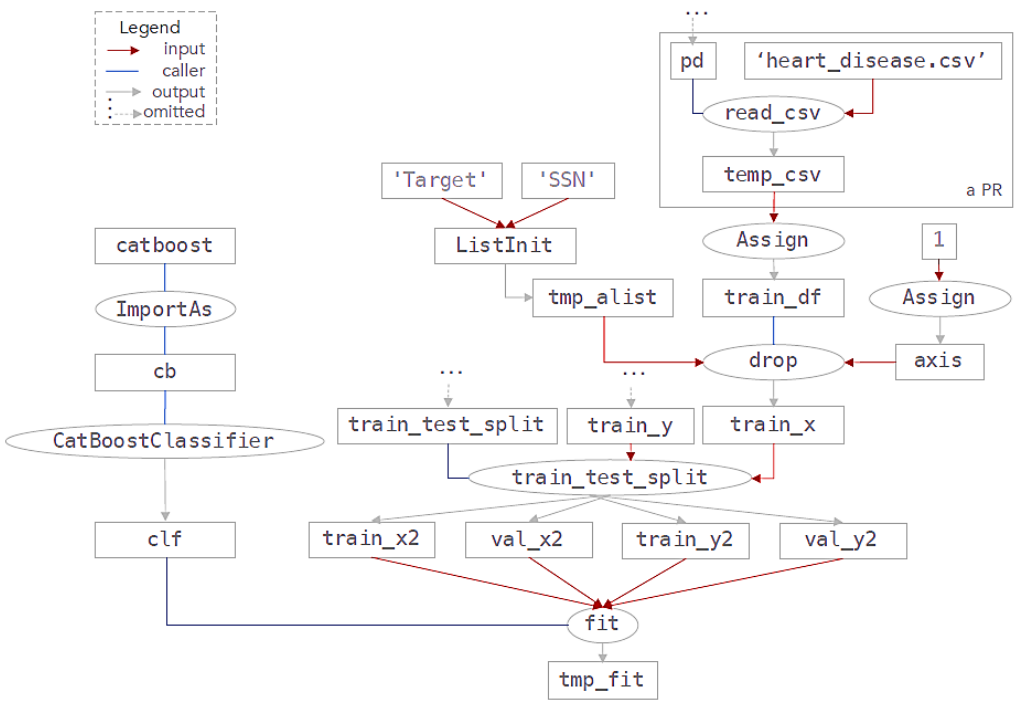
foram utilizados para derivar os recursos de um modelo de AM. O Vamsa é composto por três módulos principais:

- Extrator de derivação dos dados: gera uma representação intermediária do *workflow* do *script* (*Workflow Intermediate Representation* - WIR). É utilizada para formar um grafo direcionado a partir dos principais elementos do *workflow*, incluindo bibliotecas importadas, variáveis, funções e suas dependências. Conforme exemplificado na Figura 19, o WIR indica todo o fluxo de atividades do *script* de AM, o qual considera desde o conjunto de dados utilizado (*heart_disease.csv*), as bibliotecas (*Pandas*[9] *e CatBoost*[10]) até as atividades de treinamento e teste que são realizadas utilizando o algoritmo *CatBoostClassifier*;
- Anotador de variáveis utilizando base de conhecimento: utiliza um algoritmo de anotação genérica e uma base de conhecimento que contém informações sobre as várias APIs de diferentes bibliotecas de AM. Esse módulo permite a anotação de variáveis do fluxo de trabalho intermediário com base em suas funções no *script* (por exemplo, recursos, rótulos e modelos); e
- Rastreador de dados de proveniência: infere um conjunto de colunas que foram explicitamente incluídas ou excluídas para uso no modelo de AM, o que é obtido por meio do fluxo de trabalho intermediário anotado (uma representação estruturada que captura todas as operações e transformações realizadas sobre os dados no script) e consultando a base de conhecimento. Essa inferência é realizada considerando regras definidas na base de conhecimento, que orientam o sistema a percorrer o grafo do *script* para localizar constantes, nomes de colunas ou intervalos de índices utilizados nas operações de seleção ou remoção de colunas.

Os testes foram realizados considerando scripts Python (provenientes do Kaggle[11] e do GitHub[12]) para a validação da aplicabilidade do Vamsa em diferentes contextos. Os experimentos avaliaram a eficácia da proposta na atividade de identificação de colunas dos conjuntos de dados utilizados para derivar os atributos utilizados em modelos de aprendizado de máquina. Os resultados demonstraram que o Vamsa atingiu uma precisão que varia entre 90,4% e 99,1% para rastrear as colunas relevantes. Além disso, o sistema se mostrou capaz de operar de forma autônoma sobre *scripts* não modificados, adaptando-se a diferentes bibliotecas e frameworks de AM, o que reforça sua aplicabilidade prática em cenários reais.

[9] https://pandas.pydata.org/
[10] https://catboost.ai/docs/en/
[11] https://www.kaggle.com/
[12] https://github.com

Figura 19 – Exemplo de WIR.




Fonte: NAMAKI et al., 2020.

### 3.2.5 CHAPMAN et al. (2020)

Nesse trabalho é realizada a análise das operações realizadas durante o pré-processamento de dados dentro do contexto de AM. Os autores consideram as operações de pré-processamento dos conjuntos de dados em três classes principais, a saber:

(i) Redução de dados: operações que reduzem o tamanho do conjunto de dados eliminando linhas (*selection)* ou colunas (*projection)*;

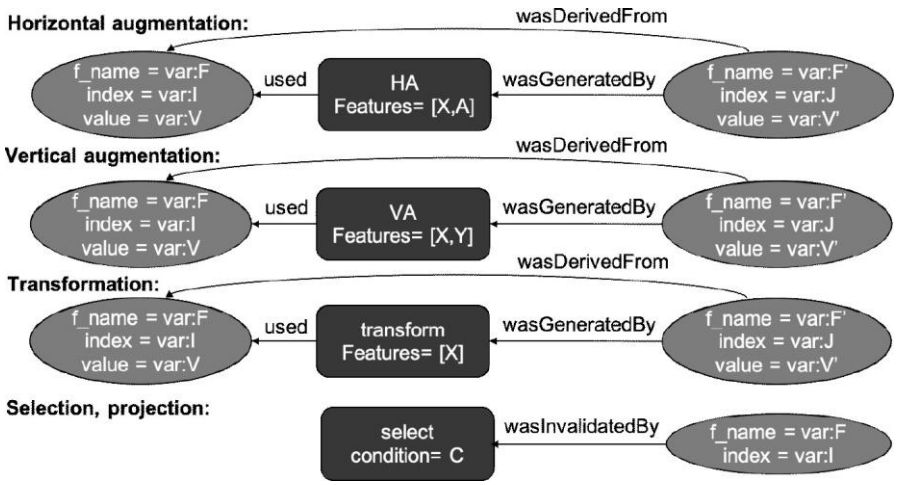
(ii) Aumento de dados: operações que aumentam o tamanho do conjunto de dados adicionando linhas (*vertical augmentation)* ou colunas (*horizontal augmentation*); e

(iii) Transformação de dados (*data transformation*): operações que, aplicando funções adequadas, transformam elementos do conjunto de dados sem alterar seu tamanho ou seu esquema. Essas funções modificam apenas os valores internos dos dados, mantendo o mesmo número de linhas e colunas. Exemplos dessas funções incluem imputação de valores faltantes, normalização, binarização, discretização e correção de valores inconsistentes, como a substituição de valores nulos e a binarização de atributos.

Para cada operação de pré-processamento, é associado um modelo (Figura 20) de função de geração de dados de proveniência (p-gen), que registra os dados de proveniência

das operações realizadas em detalhes. Nesse modelo formal, foram desenvolvidas técnicas práticas para registrar as derivações de dados no nível dos elementos atômicos no conjunto de dados, para uma classe geral de operadores de transformação de dados. No trabalho, essas derivações foram registradas utilizando o modelo de dados PROV (W3C). Os dados de proveniência associados às derivações de dados formam um conjunto de metadados estruturados em forma de grafo, que podem ser consultados como uma etapa preliminar para dar suporte às perguntas do usuário sobre as propriedades do modelo de AM.

Figura 20 - Modelos utilizados pela função p-gen.




Fonte: CHAPMAN et *al.*, 2020.

No trabalho, foram consideradas transformações que se aplicam em conjuntos de dados tabulares ou relacionais, como, por exemplo, detecção e remoção de outliers e eliminação de duplicatas. No modelo de proveniência, uma entidade representa um elemento de um conjunto de dados e é identificada exclusivamente pelo conjunto de dados e por seu índice de linha + atributo. Já uma atividade representa qualquer manipulação de dados de pré-processamento que opera sobre um conjunto de dados. Os dados de proveniência gerados são armazenados no MongoDB utilizando coleções em formato JSON.

Conforme demonstrado na Figura 21, para cada registro de dados em uma tabela, é realizada uma derivação do atributo *Age* com a criação do atributo *AgeRange*. No exemplo também ocorre a seleção de registros que não tenham o valor do atributo *AgeRange* igual à *Young*. Isso resulta em um novo conjunto (à direita) de modelos instanciados que são produzidos por uma função de geração de proveniência (*p-gen*), tendo o primeiro registro sendo considerado inválido.

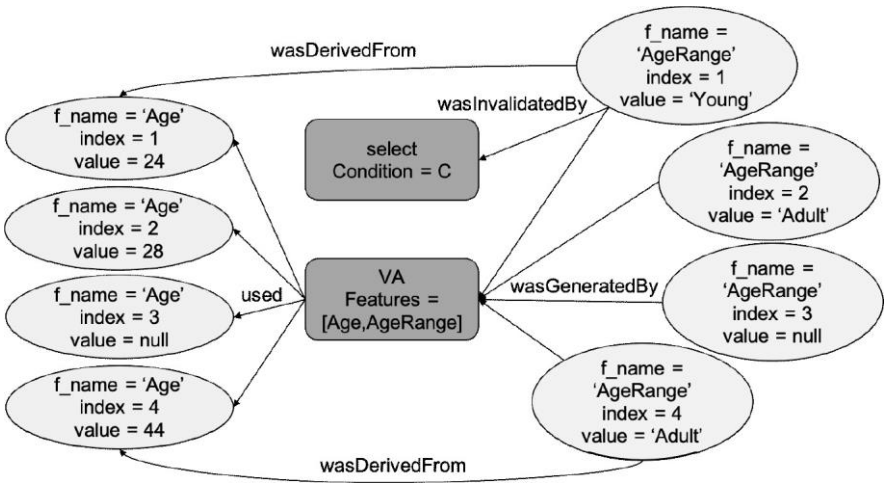
Os autores ressaltam que um esforço substancial é dedicado à preparação dos dados para uso na modelagem de AM e que as alterações feitas durante a preparação podem afetar o

modelo final. Por isso, torna-se importante ser capaz de rastrear o que está acontecendo com os dados em um detalhado nível de granularidade, como o realizado por eles. A abordagem foi avaliada por meio de experimentos realizados em *pipelines* do mundo real na linguagem Python, focando em operações realizadas utilizando a biblioteca Pandas, para limpeza e pré-processamento de dados.

Durante as análises, também foi avaliada a capacidade de responder a consultas que consideram os dados de proveniência. Exemplos dessas consultas incluem identificar todas as operações aplicadas aos dados, rastrear quais dados de entrada influenciaram determinado valor, verificar como valores foram gerados e identificar registros ou atributos que foram invalidados ao longo do *pipeline*. Além disso, foi avaliada a capacidade de analisar mudanças na distribuição dos dados, demonstrando a eficácia da captura de proveniência em cenários práticos. Por exemplo, no caso em que um modelo apresentou uma avaliação de precisão incorreta devido a um conjunto de dados desbalanceados, e a análise da proveniência permitiu identificar como as transformações impactaram a distribuição dos atributos.

A partir da avaliação da proposta, os autores observaram que é possível coletar proveniência das operações de forma útil, oferecendo informações relevantes para auditoria e depuração de *pipelines*, e eficiente, com baixa sobrecarga de tempo e espaço, através da geração incremental de fragmentos de proveniência armazenados de forma escalável em MongoDB.

Figura 21 - Exemplo de modelos instanciados produzidos por uma função *p-gen.*




Fonte: CHAPMAN et *al.,* 2020.

## 3.3 ANÁLISE COMPARATIVA

Em relação aos trabalhos que realizam a extensão da PROV no contexto de *pipelines* de AM, SCHLEGEL & SATTLER (2023) focam no versionamento e documentação de execuções, mas não oferecem suporte ao detalhamento da proveniência prospectiva. ESTEVES et al. (2015) priorizam a interoperabilidade para o vocabulário MEX, e focam nas execuções e algoritmos de AM, sem o detalhamento de outras atividades, como atividades de pré-processamento.

BUTT & FITCH (2020), apesar de considerar o uso da extensão no contexto de AM, propõem uma ontologia um pouco mais genérica e voltada a *workflows* científicos. Já SOUZA et al. (2021), no PROV-ML, não consideram o detalhamento da origem de todos os artefatos envolvidos na execução. Além disso, é considerado apenas o agente diretamente relacionado à execução, sem a definição de outros agentes que possam estar envolvidos em elementos ou artefatos que também fazem parte do processo.

A extensão PROVX se destaca por integrar, de forma estruturada, o detalhamento das execuções de *pipelines* de AM com aspectos específicos da proveniência em um contexto iterativo. A extensão proposta inclui classes como: *Run*, composta pelos metadados relacionados às execuções; *BasicProvenance*, descreve a origem do *pipeline* e do conjunto de dados; e *ExtendedProvenance*, combina a perspectiva de planejamento (prospectiva) e o histórico de execução (retrospectiva). Além disso, possui a propriedade *hasPrevious*, a qual permite identificar execuções anteriores relacionadas, o que também viabiliza a reutilização dos dados.

Embora alguns dos trabalhos analisados abordem o uso de dados de proveniência em *pipelines* de AM, não são considerados aspectos para viabilizar a reexecução consistente de atividades específicas e nem a sua reutilização automatizada em novas execuções do *pipeline*. Tais aspectos envolvem: a obtenção detalhada dos dados de proveniência dos artefatos e atividades relacionadas às execuções; a estruturação semântica dos dados de proveniência, com suporte à consulta e interpretação automática; a identificação de execuções anteriores relacionadas; e a utilização automatizada desses dados para ajustes em atividades.

Em OLIVEIRA et al. (2024), os dados de proveniência são relacionados com os atributos importantes para o treinamento do modelo gerado, o que permite indicar o histórico de transformações realizadas sobre esses atributos, promovendo um maior detalhamento. Esse detalhamento realizado é utilizado para o entendimento do usuário sobre o processo pelo qual os dados passaram. No trabalho desenvolvido, esses dados não são aplicados para outra finalidade. Também não há a possibilidade de as informações obtidas influenciarem diretamente nos resultados dos modelos, sendo necessário que o usuário aplique esse conhecimento adquirido a partir do que foi observado.

A abordagem proposta por KERZEL et al. (2021) se concentra mais na captura e visualização de dados de proveniência. Embora o MLProvLab ofereça funcionalidades para o rastreamento e a visualização de dados de proveniência em *pipelines* de aprendizado de máquina, não são utilizadas ontologias para estruturar semanticamente os dados de proveniência. No trabalho também não é implementado um repositório dedicado para o gerenciamento desses dados, o que dificulta o seu uso posterior e a análise desses dados de proveniência.

O trabalho realizado por SCHELTER et al. (2023) se concentra na identificação de problemas como vazamentos e erros de rotulagem. A abordagem enfatiza a importância da estruturação semântica dos dados de proveniência e promove melhorias contínuas nos modelos de AM por meio da análise semântica, que permite a identificação desses problemas.

CHAPMAN et *al*. (2020) realizam a captura de dados de proveniência considerando apenas as atividades de preparação dos dados no contexto de AM, sendo essa proveniência capturada a partir da execução das atividades (*workflows*). Por outro lado, em NAMAKI et *al.* (2020) não são consideradas atividades de preparação de dados, o estudo se concentra em atividades que correspondem à modelagem de AM. Além disso, os dados de proveniência obtidos não são estruturados de forma semântica.

Trabalhos como os de SCHELTER et al. (2024), CHAPMAN et al. (2020) e OLIVEIRA et al. (2024) apresentam contribuições relevantes no que se refere à documentação e análise de execuções de *pipelines* de aprendizado de máquina. No entanto, tais abordagens não propõem mecanismos integrados que viabilizem a reexecução e o ajuste direto de atividades com base em execuções anteriores.

O protótipo nFlowX oferece uma solução integrada e estruturada, permitindo desde a documentação, até o suporte à reexecução e o ajuste de atividades, visando a melhoria dos modelos de AM. O Quadro 1 compara os trabalhos sobre dados de proveniência em pipelines de AM, apresentados neste capítulo, com esta tese. Nesse quadro, são destacados aspectos como o tipo de proveniência abordado (prospectiva, retrospectiva ou ambos), o modelo utilizado para representação/persistência dos dados, além de funcionalidades como gerenciamento do histórico de execuções, reexecução de atividades de preparação de dados, uso de dados de proveniência para ajustes automatizados no *pipeline* e realização de análises sobre as execuções.

Nesse sentido, esta tese propõe uma solução que, por meio do nFlowX, permite a realização de ajustes automatizados e a reexecução das atividades do *pipeline*, com base em dados de proveniência estruturados de execuções anteriores. Embora existam trabalhos que

utilizam dados de proveniência de forma ativa para guiar modificações no *pipeline* (como em OLIVEIRA et al. (2024) e em SCHELTER et al. (2023)), as alterações ainda são conduzidas manualmente. Nesses trabalhos, o usuário é responsável, ao menos, por realizar os ajustes e, em alguns casos, também pela análise detalhada das execuções anteriores para decidir quais ajustes devem ser realizados. Essa abordagem manual demanda esforço interpretativo, limitando o uso mais dinâmico e automatizado desses dados na melhoria do *pipeline*.

Em relação aos trabalhos que estendem os elementos da PROV, todos buscam se adequar a aspectos mais específicos dentro do contexto de AM ou não contemplam todo o detalhamento da execução do *pipeline*. Entre esses trabalhos, não foi identificada extensão da PROV que represente a precedência entre diferentes execuções e que, simultaneamente, considere um conjunto de metadados de proveniência de todos os artefatos envolvidos na execução. Essa representação também deveria incluir a proveniência prospectiva e retrospectiva, dados relacionados ao modelo de AM, assim como dados gerais e detalhes sobre a execução do pipeline.

Quadro 1 - Comparação entre os trabalhos relacionados.

| **Referência** | **Tipo de Proveniência** | **Modelo para Representação/Persistência dos dados de proveniência** | **Gerenciamento do histórico de execuções de atividades do *Pipeline* de AM** | **Aplica a reexecução de Atividades de Preparação de Dados** | **Uso de dados de Proveniência para ajustar de forma automatizada atividades do *Pipeline* de AM** | **Análises sobre as execuções (consultas)** |
|---|---|---|---|---|---|---|
| ESTEVES et al. (2015) | Prospectiva e Retrospectiva | MEX (Extensão/PROV) | Sim | Não | Não | Não |
| NAMAKI et al. (2020) | Prospectiva e Retrospectiva | WIR | Não | Não | Não | Não |
| CHAPMAN et al. (2020) | Prospectiva e Retrospectiva | PROV | Não | Não | Não | Sim |
| KERZEL et al. (2021) | Prospectiva e Retrospectiva | Não | Sim | Não | Não | Sim |
| SCHLEGEL & SATTLER (2023) | Retrospectiva | MLflow2PROV (Extensão/PROV) | Sim | Não | Não | Sim |
| SCHELTER et al. (2023) | Retrospectiva | Não | Sim | Não | Não | Sim |
| OLIVEIRA et al. (2024) | Prospectiva e Retrospectiva | PROV | Sim | Não | Não | Sim |
| BUTT & FITCH (2020) | Prospectiva e Retrospectiva | ProvONE+ (Extensão/PROV) | Não | Não | Não | Não |
| SOUZA et al. (2021) | Prospectiva e Retrospectiva | Prov-ML (Extensão/PROV) | Sim | Não | Não | Sim |
| **Esta Tese** | **Prospectiva e Retrospectiva** | **PROVX (Extensão/PROV)** | **Sim** | **Sim** | **Sim** | **Sim** |

Fonte: O Autor.

## 3.4 CONSIDERAÇÕES

Neste capítulo, foram apresentados os trabalhos relacionados ao tema desta tese, com uma análise comparativa entre as abordagens, considerando suas principais características em relação à proposta deste trabalho. A partir dessa análise, embora existam trabalhos relevantes para a captura, representação e utilização de dados de proveniência em pipelines de aprendizado de máquina, destaca-se a ausência de soluções focadas no processo iterativo de execuções do *pipeline* de AM que utilizem esses dados de proveniência para a realização de ajustes automáticos no *pipeline*. No próximo capítulo, é apresentada a proposta deste trabalho, que utiliza dados de proveniência para otimizar o desenvolvimento do *pipeline* de AM, com foco na atividade de seleção de atributos de forma automatizada.

# 4 DADOS DE PROVENIÊNCIA EM CONTEXTOS DE EXECUÇÕES DE PIPELINES

Este capítulo inicia com a Seção 4.1, a qual detalha o problema e o cenário em que a proposta se enquadra. Em seguida, a Seção 4.2 aborda as definições preliminares necessárias para o entendimento deste trabalho. A Seção 4.3 apresenta a abordagem proposta, por meio da exploração das fases de ajustes de atividades do *Pipeline*, Execução do *Pipeline*, Estruturação da Proveniência (Extensão PROVX) e Realização de Consultas. Por fim, é apresentado o caso de uso, além de serem tecidas algumas considerações do capítulo.

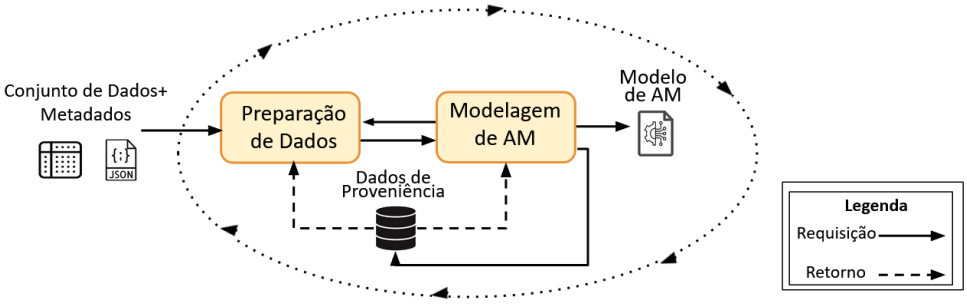
## 4.1 DEFINIÇÃO DO PROBLEMA E CENÁRIO

Neste trabalho, é considerado o contexto de *pipelines* de treinamento de modelos preditivos. Sendo analisado de que forma os dados de proveniência capturados nas fases de preparação de dados e modelagem do aprendizado podem contribuir para a reexecução e o ajuste de atividades do *pipeline* de AM, especialmente em processos iterativos.

A partir do acesso a dados de proveniência obtidos durante execuções anteriores do *pipeline*, é possível auxiliar no desenvolvimento de modelos de aprendizado. Como ilustração, o uso de dados de proveniência prospectiva/retrospectiva, considerando o plano de execução do *pipeline* ou os resultados obtidos, podem ser aplicados em atividades do *pipeline*, como na escolha dos parâmetros do modelo de aprendizado e em atividades de seleção de atributos durante a fase de preparação de dados.

Para isso, consideramos um processo de desenvolvimento do *pipeline* de AM (Figura 22), focado nas fases de preparação de dados e modelagem de AM. Os dados de proveniência capturados durante a execução dessas fases podem ser utilizados em iterações futuras, auxiliando o desenvolvimento das atividades do *pipeline*.

Figura 22 – Escopo da Abordagem Proposta para o *Pipeline* de AM.




Fonte: O Autor.

Conforme mostrado a Figura 22, para o desenvolvimento do *pipeline* de AM, consideramos um processo composto por etapas, as quais possuem suas respectivas atividades, e um repositório de dados de proveniência. Esses dados, coletados durante a execução dessas etapas, podem ser utilizados para ajustar atividades das etapas do pipeline, como a seleção de atributos.

A primeira execução de um *pipeline* de treinamento de um modelo preditivo é considerada como a execução base, pois não há execução anterior do *pipeline* que permita utilizar os dados de proveniência. Mas, a partir dessa primeira execução do *pipeline*, ocorre o armazenamento de dados de proveniência da execução no repositório de dados utilizando o formato RDF, o que considera desde os dados sobre a origem do conjunto de dados e do *pipeline* envolvido na execução, até a proveniência prospectiva e retrospectiva, definidas na seção a seguir. Quando houver uma nova execução, é indicada a existência de execução anterior relacionada e será possível fazer uso desses dados.

Após a primeira execução do *pipeline*, em que os dados de proveniência são armazenados no repositório, novas execuções podem se beneficiar dos dados armazenados. Essa identificação das execuções relacionadas é estabelecida quando existem execuções anteriores que utilizam o *pipeline P* e o conjunto de dados *D*. O objetivo é permitir o reaproveitamento dos dados de proveniência para ajustes e otimizações nas novas execuções, evitando retrabalho e aprimorando o processo de desenvolvimento de modelos de AM.

Para formalizar essa relação, consideramos o seguinte cenário:

- Um *pipeline* $P_i$ é executado utilizando um conjunto de dados $D_j$, que possui seus respectivos metadados $D_{jmeta}$ ;
- Cada execução é representada como $Exec_n(P_i, D_j, D_{jmeta})$; e
- Durante a execução, são gerados dados de proveniência $Prov_{Exec_n}$, os quais são armazenados em um repositório de dados.

Se for identificada no repositório de dados uma execução ($Exec_{n-m}(P_i , D_j, D_{jmeta})$), isto é, uma execução anterior do *pipeline* $P_i$ com o conjunto de dados $D_j$, os dados de proveniência de uma execução anterior ($Prov_{Exec_{n-m}}$) podem ser utilizados para auxiliar no ajuste da nova execução $Exec_n(P_i , D_j, D_{jmeta})$.

Formalmente, essa relação pode ser descrita como:

> Se $\exists \left(Exec_{n-m}(P_i , D_j, D_{jmeta})\right)$, então utilizar $Prov_{Exec_{n-m}}$ para ajustar a execução $Exec_n(P_i , D_j, D_{jmeta})$.

Por exemplo, os dados de proveniência armazenados podem ser utilizados para otimizar a etapa de seleção de atributos, permitindo a aplicação de técnicas como a RFE, para priorizar os atributos mais relevantes do modelo da execução relacionada. Além disso, os dados de

proveniência podem ser utilizados em outras alterações no *pipeline*, como a escolha de hiperparâmetros, algoritmos e estratégias de pré-processamento, além de evitar falhas recorrentes.

## 4.2 DEFINIÇÕES PRELIMINARES

Nesta seção, são apresentados conceitos específicos adotados neste trabalho, que refletem as particularidades do estudo e estão alinhados com a literatura existente (TEUBL et al., 2023; FREIRE et al., 2008; LÓSCIO et al., 2017; LEBO et al., 2013; GARIJO et al., 2014). As definições são usadas para fundamentar a proposta da abordagem e algumas discussões apresentadas nas seções seguintes.

**Definição 01 - Proveniência básica do conjunto de dados – $\boldsymbol{PBD(D_{jmeta})}$:** Tendo em consideração os metadados $D_{jmeta}$ de um conjunto de dados $D_j$ , a proveniência básica de $D_j$ se refere aos metadados relacionados à sua origem, expressa da seguinte forma:

$$PBD(D_{jmeta}) = (dt_c,\ dt_u, v, cr, pb, f)$$

Em que:

$dt_c$*:* Data de criação, representada como *timestamp*;

$dt_u$*:* Data da última atualização, representada como *timestamp*;

$v$*:* Versão do conjunto de dados;

$cr$*:* Criador do conjunto de dados;

$pb$*:* Publicador do conjunto de dados; e

$f$*:* Fonte do conjunto de dados.

**Definição 02 - Proveniência Básica do *Pipeline* - $\boldsymbol{PBP(P_i)}$:** Considerando um *pipeline* $P_i$ , a proveniência básica de $P_i$ se refere aos metadados relacionados à sua origem, definida como:

$$PBP(P_i) = (dt_c,\ dt_u, v, r)$$

Em que:

$dt_c$*:* Data de criação, representada como *timestamp*;

$dt_u$*:* Data da última atualização, representada como *timestamp*;

$v$: Versão do *pipeline*; e

$r$: Responsável pelo desenvolvimento do *pipeline*.

**Definição 03 - Proveniência Prospectiva – $\boldsymbol{\pi PP(P_i,\ D_j)}$:** A proveniência prospectiva de um *pipeline* $P_i$ , que utiliza o conjunto de dados $D_j$ , é composta por um conjunto de etapas planejadas a serem realizadas, definido por:

$$\pi PP(P_i,\ D_j) = E$$

Em que:

$E$ representa o conjunto de etapas do *pipeline*, dado por: $E = \{e_1, e_2, \ldots e_k\}$.

Cada $e_x$ contém um conjunto de atividades: $A_x = \{a_{x1}, a_{x2}, \ldots a_{xp}\}$, em que cada atividade $a_{xy}$ representa uma função ou operação a ser executada.

**Definição 04 - Proveniência Retrospectiva – $\pi PR(P_i, D_j)$:** A proveniência retrospectiva de um *pipeline* $P_i$, que utiliza o conjunto de dados $D_j$, corresponde a um conjunto de etapas que foram executadas $\pi PR(P_i, D_j) = Res(E) = \{r_1, r_2, \ldots r_K\}$, em que:

Cada $r_x$ representa o resultado da execução da etapa planejada $e_x$, que corresponde aos dados reais da execução de $A_x$ associados à etapa $e_x$, incluindo suas entradas e saídas. Considera $\pi PP(P_i, D_j) \rightarrow \pi PR(P_i, D_j)$, ou seja, as etapas executadas $Res(E)$ são dependentes das etapas planejadas $E$.

**Definição 05 - Proveniência Estendida – $PE(P_i, D_j)$:** se refere ao planejamento e execução de um *pipeline* $P_i$ que utiliza um conjunto de dados $D_j$, sendo:

$$PE(P_i, D_j) = \left(\pi PP(P_i, D_j), \pi PR(P_i, D_j)\right)$$

Em que:

$\pi PP(P_i, D_j)$*:* Proveniência Prospectiva de uma execução; e

$\pi PR(P_i, D_j)$*:* Proveniência Retrospectiva de uma execução.

**Definição 06 – Dados do Modelo de AM – $MM(P_i, D_j)$:** Considerando que um *pipeline* $P_i$ é executado utilizando um conjunto de dados $D_j$, os dados do modelo de AM correspondem às informações capturadas durante essa execução. Esses dados descrevem aspectos do algoritmo de AM utilizado no pipeline, definidos da seguinte forma:

$$MM(P_i, D_j) = (alg, param_{alg}, sets, scr)$$

Em que:

$alg$: o algoritmo de classificação, por exemplo, *Random Forest* ou *SVM* (Máquinas de Vetores de Suporte);

$param_{alg}$: conjunto de parâmetros especificados para o algoritmo, como por exemplo, para *Random Forest*: *n_estimators*=100, *max_depth*=10;

$sets$: especificação da partição dos conjuntos de treinamento, teste e validação, por exemplo: divisão de 60% para treinamento, 20% para teste e 20% para validação; e

$scr$: conjunto de pontuações das métricas de avaliação, como acurácia, precisão, cobertura e *F1-score*.

**Definição 07 - Execução do *pipeline* de AM – $\boldsymbol{Exec_n(P_i,\ D_j,\ D_{jmeta})}$:** Considera a execução de um *pipeline* $P_i$ que utiliza um conjunto de dados $D_j$ (e apresenta seus metadados $D_{jmeta}$), referindo-se a um processo para executar uma sequência de atividades estabelecidas nas etapas que compõe $P_i$ e utilizam $D_j$, a fim de treinar um modelo de AM. O resultado do modelo e as informações relacionadas com a execução são obtidos da seguinte forma:

$$Exec_n(P_i,\ D_j,\ D_{jmeta}) = (Prov_{Exec_n}, res_p)$$

Em que:

$Prov_{Exec_n}$ : é o conjunto de informações de proveniência associadas à execução, incluindo:

$PBD(D_{jmeta})$ : é a proveniência básica do conjunto de dados;

$PBP(P_i)$ : é a proveniência básica do *pipeline*;

$PE(P_i,\ D_j)$ : é a proveniência estendida da execução*; e*

$MM(P_i,\ D_j)$ : são os dados referentes ao modelo gerado pelo *pipeline*.

$res_p$ são as saídas do *pipeline* executado.

**Definição 8 - Reexecução de Atividade por Dados de Proveniência – $\boldsymbol{AtvR(D_j, A)}$:** Antes da execução $Exec_n(P_i,\ D_j,\ D_{jmeta})$, um conjunto de Atividades ($A$) pertencentes a uma ou mais Etapas ($E$) de um *pipeline* $P_q$, pode ser reexecutado considerando o plano apresentado em $\pi PP(P_q,\ D_r)$, estabelecido em uma $PE_{n-m}(P_q,\ D_r) \in Exec_{n-m}(P_q,\ D_r,\ D_{rmeta})$.

Essa reexecução é possível quando as seguintes condições são atendidas:

$Title(P_i) = Title(P_q)$ && $(dt_c, r) \in PBP(P_i) = (dt_c, r) \in PBP(P_q)$, ou seja, o título, a data de criação e o responsável pelo $P_i$ são equivalentes ao do *pipeline* $P_q$; e

$Title(D_j) = Title(D_r)$ && $(dt_c, cr) \in PBD(D_j) = (dt_c, c\,r) \in PBD(D_r)$, indicando que o título, a data de criação e o criador do conjunto de dados $D_j$ e $D_r$ também são equivalentes.

Isso possibilita a realização de novas análises que podem contribuir para o ajuste do *pipeline* $P_i$ na nova execução $Exec_n(P_i,\ D_j,\ D_{jmeta})$.

**Definição 9 - Ajuste de *Pipeline* Utilizando Dados de Proveniência - $\boldsymbol{AjProv(P_i,\ A)}$ :** Antes de uma execução $Exec_n(P_i,\ D_j,\ D_{jmeta})$, uma ou mais atividades do *pipeline* $P_i$ podem ser ajustadas com base em dados derivados das análises obtidas em $Exec_{n-m}(P_q,\ D_r,\ D_{rmeta})$ ou em $AtvR(D_j, A)$, desde que sejam atendidas as seguintes condições:

$Title(P_i) = Title(P_q)$ && $(dt_c, r) \in PBP(P_i) = (dt_c, r) \in PBP(P_q)$, ou seja, o título, a data de criação e o responsável pelo $P_i$ são equivalentes ao do *pipeline* $P_q$; e $Title(D_j) = Title(D_r)$ && $(dt_c, cr) \in PBD(D_j) = (dt_c, c\,r) \in PBD(D_r)$, indicando que o título, a data de criação e o criador do conjunto de dados $D_j$ e $D_r$ também são equivalentes.

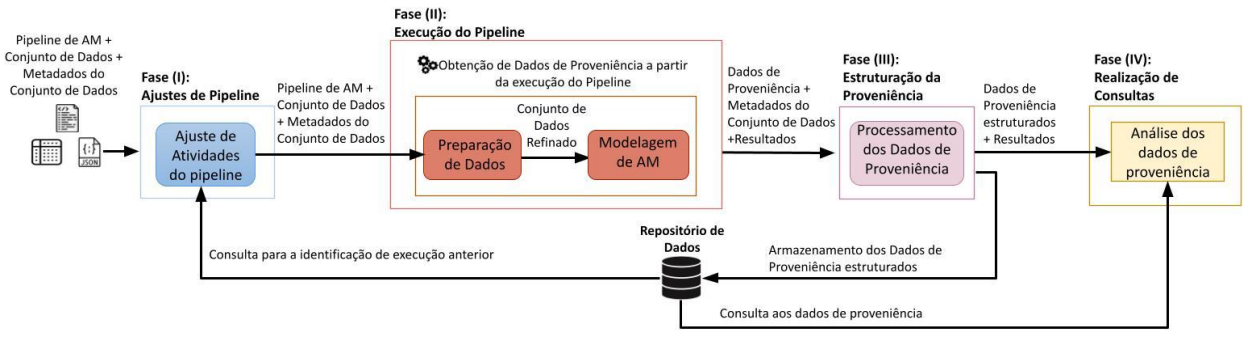
Como, por exemplo, com a identificação de um conjunto de atributos que melhora a avaliação do modelo de AM, é possível ajustar o *pipeline*, resultando em um *pipeline* $P'_i$ e em uma execução $Exec_n(P'_i, D_j, D_{jmeta})$, refletindo as modificações realizadas.

## 4.3 DADOS DE PROVENIÊNCIA APLICADOS AO DESENVOLVIMENTO DO PIPELINE DE AM

A abordagem proposta considera um processo que inclui a estruturação semântica dos dados de proveniência relacionados à execução de *pipelines* de AM, até a sua utilização para apoiar a seleção mais adequada de atributos. A seleção de atributos exerce influência direta sobre os resultados dos modelos gerados, podendo contribuir significativamente para sua melhoria. A proposta deste trabalho destaca o uso dos dados de proveniência como elemento central e diferencial para a realização da seleção de atributos, viabilizando a aplicação desse processo de forma mais automatizada e orientada, com base nas informações registradas durante as execuções anteriores dos *pipelines*.

A abordagem proposta nesta tese (Figura 23) apresenta quatro fases principais: Ajuste de Atividades, Execução do *Pipeline* (composta, ao menos, por atividades relacionadas à preparação dos dados e ao treinamento do modelo de AM), Estruturação da Proveniência e Realização de Consultas.

Figura 23 – Visão Geral da abordagem proposta.



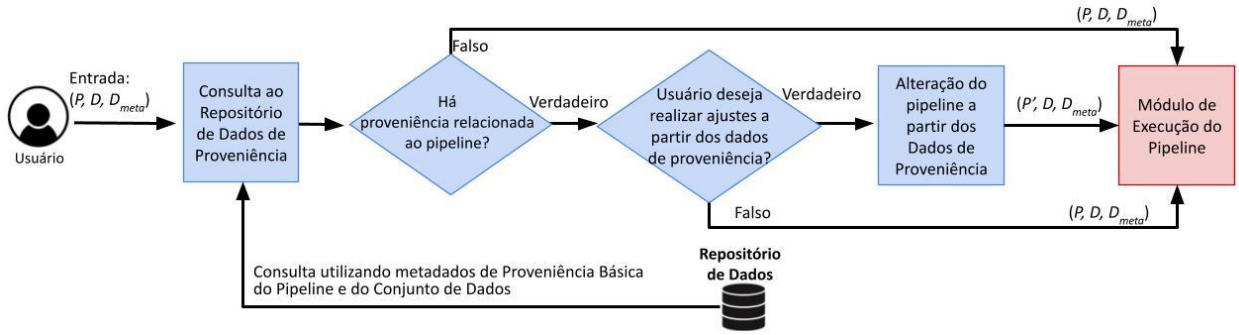

Fonte: O Autor.

### 4.3.1 Fase I: Ajustes de atividades do *Pipeline*

Nessa fase, é possível realizar ajustes antes da execução de um dado *pipeline*, sendo aplicados a partir de dados de proveniência de execuções anteriores que estão armazenados no repositório de dados (Figura 24). Para essa fase, são necessários os artefatos para a execução da abordagem (*pipeline* de AM, conjunto de dados e metadados do conjunto de dados). A partir dos dados de Proveniência Básica, é identificado se no repositório de dados existem execuções anteriores que utilizaram o *pipeline P* e conjunto de dados *D* da execução atual (Consulta SPARQL (I), detalhada no Apêndice C).

Para este cenário, consideramos um *pipeline* (*P)*, que pode apresentar um conjunto de execuções ($Exec_1$ até $Exec_n$), utilizando um conjunto de dados (*D)* e seus respectivos metadados ($D_{meta}$). Uma execução atual pode ter suas atividades ajustadas considerando os dados de proveniência obtidos a partir de execuções anteriores, de modo que uma determinada execução *n* utilize as informações obtidas a partir de uma execução *n-i*.

Figura 24 – Etapas da fase de ajustes de atividades do *pipeline*.




Fonte: O Autor.

Conforme observado na Figura 24, a partir da verificação da existência de uma execução anterior relacionada à atual, torna-se viável o ajuste automatizado do *pipeline*. Com a identificação de execução relacionada, é possível ajustar atividades já desempenhadas no *pipeline* a partir dos dados de proveniência da execução relacionada, como a atividade de seleção de atributos, utilizada neste trabalho para explicar seu uso. Essa atividade de seleção de atributos será realizada utilizando RFE do método *Wrapper* para a recomendação dos melhores atributos para o modelo.

Neste contexto, a seleção ocorre a partir da reexecução de atividades do *pipeline* da execução relacionada, considerando os dados obtidos por meio da proveniência prospectiva e retrospectiva. Com a identificação desses atributos, o *pipeline P* é ajustado para que o modelo considere apenas os atributos recomendados, resultando em um *pipeline P',* que será utilizado na execução.

Para a etapa de reexecução de atividades (que inclui atividades de preparação dos dados e do modelo de AM) e realização da RFE, foi especificado o Algoritmo I. Esse algoritmo inicia com a obtenção da execução melhor avaliada do *pipeline P* associada ao conjunto de dados *D*, por meio

da função c*onsultar_execucoes(*$P_{título,dt\_criação,resp}$*,*$D_{título,dt\_criação,resp}$*).* Caso existam execuções anteriores, é priorizada a execução que apresente melhor avaliação, e então a proveniência prospectiva *πPP(P,D)* é obtida (linha 4), permitindo a identificação das atividades da fase de preparação dos dados (linha 5) e as atividades relacionadas ao modelo de AM utilizado (linha 6), considerando os mesmos parâmetros e algoritmo. A seleção de atributos (linha 7) é realizada mediante a função *exec_selecao(AtvR([*$atv_{preparacao}$*,* $atv_{modelo}$*], D))*, sendo reexecutadas as atividades de preparação de dados e do modelo de AM, utilizando a função *AtvR*. Em seguida, é aplicada a seleção de atributos considerando o modelo obtido em *AtvR*, retornando o conjunto de atributos recomendados pela atividade de seleção ($atr_{sel}$) (linha 9).

Para a seleção dos atributos, foi utilizada a técnica RFE, que foi automatizada e integrada ao processo de reexecução de atividades, sendo realizada após a reexecução das atividades de preparação de dados e do modelo de AM. Com a avaliação iterativa do impacto da exclusão de cada atributo com base no desempenho do modelo, como na métrica de acurácia, obtido na execução do *pipeline* de melhor desempenho, é possível comparar os desempenhos e identificar se existe um conjunto de atributos que melhora a avaliação do modelo. Dessa forma, a RFE atua de forma mais direcionada e automatizada, permitindo a seleção de atributos com base no modelo da execução de melhor desempenho do *pipeline***,** buscando aprimorar os resultados**.**

Com a identificação desse conjunto de atributos, o *pipeline (P)* passa pelo ajuste da atividade de seleção de atributos utilizando os dados de proveniência (processo *AjProv(P,A)),* resultando no *pipeline atualizado (P'),* o qual será executado. Caso não seja identificado um conjunto de atributos que resulte em uma melhor avaliação do modelo, o *pipeline (P)* é executado sem alterações.

**Algoritmo I:** Seleção de Atributos com Reexecução de Atividades

**Entrada:** Conjunto de dados*(D)*, *Pipeline* de AM(*P)*

**Saída:** atributos recomendados ($atr_{sel}$)

**Início**

1: $atr_{sel} \leftarrow \{\ \}$

2: $melhor_{exec} \leftarrow$ *consultar_execucoes(*$P_{título,dt\_criação,resp}$*,*$D_{título,dt\_criação,resp}$*)*

3: **Se** $melhor_{exec} \neq \emptyset$ **Então**

4: *πPP(P,D)* ← $melhor_{exec}$.prospectiva()

5: $atvs_{preparacao}$ ← preparação_dados(*πPP(P,D))*

6: $atvs_{modelo}$ ← modelo_AM(*πPP(P,D))*

7: $atr_{sel}$ ← *exec_selecao(AtvR([*$atv_{preparacao}$*,* $atv_{modelo}$*], D))*

8: **fim Se**

9: **retorna** $atr_{sel}$

**Fim**

### 4.3.2 Fase II: Execução do *Pipeline*

Nesta fase, é realizada a execução do *pipeline (P),* ou do *pipeline atualizado (P'),* utilizando o conjunto de dados *(D).* São destacadas, nesse contexto, as etapas de preparação dos dados e de modelagem de AM, detalhadas nas seções a seguir, e considerando o pressuposto de que o conjunto de dados a ser utilizado foi previamente coletado e disponibilizado. Além disso, os dados de proveniência da execução do *pipeline* são obtidos a partir da execução do *pipeline* em conjunto com uma ferramenta que realiza a captura da proveniência, como a noWorkflow (Seção 2.2).

Como resultado dessa execução, é obtido o modelo gerado, juntamente com os dados de proveniência básica e estendida, além dos dados do modelo de AM. Em seguida, esses dados são estruturados e armazenados no repositório de dados (processo detalhado na Seção 4.3.3), para posterior utilização na fase de realização de consultas (Seção 4.3.4).

#### *4.3.2.1 Preparação de Dados*

Esta etapa recebe como entrada o conjunto de dados para a realização de atividades de pré-processamento que permitam transformar os dados a fim de atender às necessidades para o desenvolvimento do modelo de AM. Entre as atividades a serem realizadas, podemos considerar aquelas estabelecidas na fase de preparação de dados da metodologia CRISP-DM (conforme Seção 2.1.1), as quais geralmente também são executadas em *pipelines* de AM, conforme descrito em SINGH et al. (2022) e PURBASARI et al. (2021), destacando-se, neste trabalho, as seguintes atividades:

- Seleção de Dados - Seleção dos dados a serem utilizados no treinamento do modelo, considerando a tarefa de predição ou tipo de dados;
- Limpeza de Dados - Utiliza técnicas para correção de registros do conjunto de dados, por exemplo, deletar ou tratar erros e dados ausentes; e
- Enriquecimento de Dados - Geração de novas informações a partir de dados existentes, como produzir novos recursos a partir da derivação de recursos existentes.

A realização das atividades de preparação de dados depende das necessidades de cada conjunto de dados e do objetivo do modelo a ser gerado. Nem todas as atividades precisam ser necessariamente realizadas durante a execução desta fase, mas, para aquelas que forem realizadas, será feita a documentação do fluxo e das ações executadas por meio de uma ferramenta para captura de dados de proveniência. Durante a execução das atividades desta fase, os dados de proveniência prospectiva e retrospectiva também são capturados e, com a conclusão desta fase, o conjunto de dados refinado segue para a etapa de treinamento e avaliação do modelo.

#### *4.3.2.2 Modelagem de AM*

Esta é a última etapa da fase do *pipeline* e corresponde às atividades de modelagem de AM e avaliação. Com base na literatura (TEUBL et al., 2023; CHERAGHI et al., 2021), neste trabalho, são consideradas as seguintes atividades:

- Divisão do Conjunto de Dados: separação dos dados para as atividades voltadas à obtenção do modelo de AM, como treinamento, teste e validação. Todas as partes do conjunto de dados devem ser preparadas de forma que possam ser selecionadas aleatoriamente, para evitar qualquer viés ou influência intencional no processo de seleção.
- Treinamento do Modelo: etapa em que é necessário realizar ajustes nos parâmetros específicos do algoritmo escolhido; e
- Avaliação: avaliar o modelo, utilizando o conjunto de dados de teste para verificar o desempenho do modelo, podendo utilizar algumas métricas, como cobertura, precisão, *F1-score* (definidas na Seção 2.1).

Assim como na etapa anterior, os dados de proveniência prospectiva e retrospectiva também são obtidos a partir da execução das atividades realizadas. Os metadados do conjunto de dados, os dados sobre a origem do *pipeline* (por exemplo, data de atualização e criador) e os dados de proveniência obtidos a partir do *pipeline* (proveniência prospectiva e retrospectiva) são encaminhados para a fase de estruturação dos dados de proveniência.

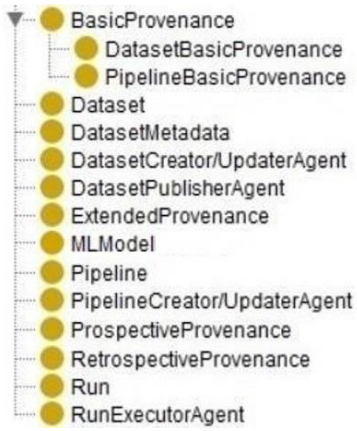
### 4.3.3 Fase III: Estruturação dos Dados de Proveniência – Extensão PROVX

Nesta fase, os dados de proveniência obtidos na fase anterior são estruturados semanticamente utilizando a extensão proposta (PROVX) da Ontologia PROV (W3C). As informações estruturadas geradas nesta fase podem ser agrupadas em três categorias: (I) Proveniência Básica do Conjunto de Dados e Proveniência Básica do *Pipeline*: consistem em informações descritivas sobre a origem dos artefatos, como data de criação, atualização, e agentes responsáveis (por exemplo, autores e publicadores); (II) Dados do Modelo de AM: obtidos por meio do *pipeline* executado, fornecem detalhes essenciais sobre o modelo de AM e promovem confiança nos modelos (LI et al., 2023); e (III) Proveniência da Execução: considerando a proveniência estendida (proveniência prospectiva e retrospectiva), detalhes da execução e versões.

A PROVX, detalhada no Apêndice A, compreende 15 novas classes (Figura 25), sendo 13 entidades primárias e 2 subclasses. Algumas dessas classes estão vinculadas à ontologia P-Plan, que descreve planos e seus componentes, e à MLOnto[13], especializada em representar conceitos e práticas em aprendizado de máquina.

[13] https://github.com/MLOntology/MLO

Figura 25 - Lista de Classes definidas.




Fonte: O Autor.

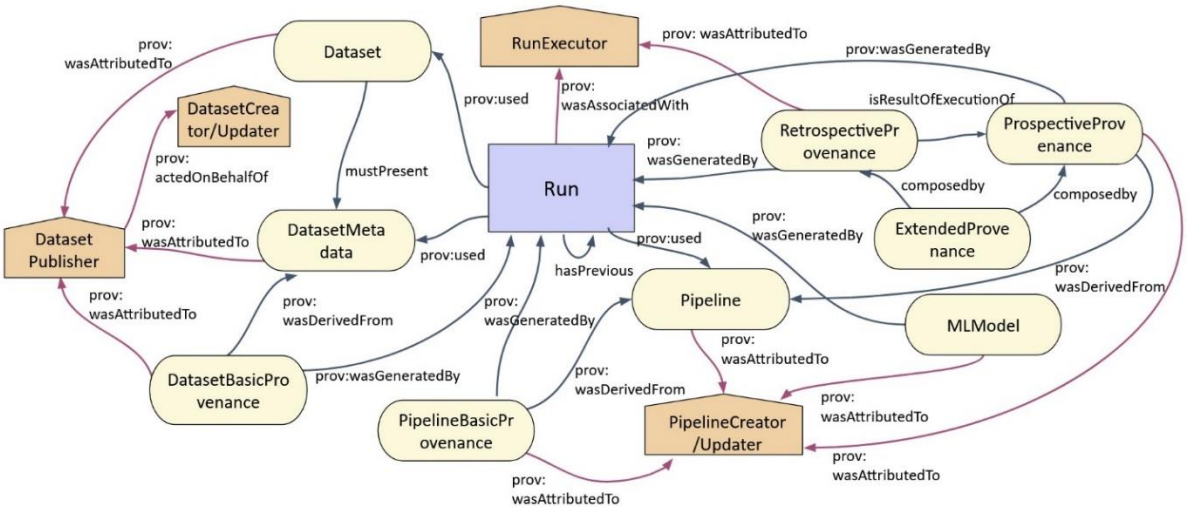
Essa integração facilitou a construção de um modelo capaz de descrever tanto o planejamento e a execução de processos quanto o desenvolvimento das atividades ao longo do tempo, tendo seus conceitos e definições estabelecidos a seguir:

- *BasicProvenance*: Esta classe representa a entidade generalizada de um *DatasetBasicProvenance* ou um *PipenilineBasicProvenance*. Ela é gerada pela atividade *Run* e consiste de propriedades que descrevem a origem do arquivo, incluindo sua data de criação/atualização e o *prov:Agent* responsável por ele;
- *Dataset*: Esta classe representa uma coleção organizada de dados disponíveis na Web, a qual é usada na atividade *Run*;
- *DatasetMetadata*: Esta classe representa uma coleção organizada de metadados que descreve o conjunto de dados, incluindo informações sobre sua origem e estrutura, sendo usada na atividade *Run*;
- *DatasetCreator*/*UpdaterAgent*: Esta classe representa o *prov:Agent* responsável por criar/atualizar o conjunto de dados;
- *DatasetPublisherAgent*: Esta classe representa o *prov:Agent* responsável por publicar o conjunto de dados na Web;
- *ExtendedProvenance*: Esta classe representa os detalhes da execução do *pipeline*, incluindo a indicação da *ProspectiveProvenance* e da *RetrospectiveProvenance*;
- *MLModel*: Esta classe se refere aos dados que descrevem o modelo de AM. Está relacionada a *mlo:Algorithms* (indica o(s) algoritmo(s) usado(s) no modelo) e é gerada pela atividade *Run*;
- *Pipeline*: Esta classe representa uma sequência organizada de instruções bem definidas projetadas para treinar e testar um ou mais algoritmos de AM, usando dados para fazer previsões ou classificações. Essas instruções são executadas na atividade *Run*;

- *PipelineCreator*/*UpdaterAgent*: Esta classe representa o *prov:Agent* responsável por criar e atualizar o *pipeline*;
- *ProspectiveProvenance*: Esta classe representa a estrutura do *Pipeline*, suas etapas formam um *p-plan:Plan.* Cada etapa corresponde a um *p-plan:Step*.
- RetrospectiveProvenance: Esta classe representa a proveniência retrospectiva obtida a partir de *provx:Run*, descrevendo a execução das etapas previstas em *ProspectiveProvenance*.
- *Run*: Esta classe representa a atividade que abrange todo o processo de execução de um *pipeline* de AM, gera metadados que consideram aspectos relacionados à proveniência e ao modelo de AM; e
- *RunExecutorAgent*: Esta classe representa o *prov:Agent* responsável por executar a atividade *Run*.

Conforme observado na Figura 26, o modelo associado à ontologia proposta apresenta a atividade central *Run* que considera as fases de execução do *pipeline* e a geração dos metadados. Essa atividade representa a execução de uma entidade *pipeline*, sobre uma determinada entidade de conjunto de dados e sua respectiva entidade de metadados do conjunto de dados (todas essas entidades também possuem a indicação dos agentes relacionados), sendo a atividade *Run* de responsabilidade do agente *RunExecutorAgent*. Como resultado, são geradas as seguintes entidades: (i) *DatasetBasicProvenance* e *PipelineBasicProvenance*, responsáveis pelas informações relacionadas à origem dos artefatos utilizados; (ii) *ExtendedProvenance*, composta pela *ProspectiveProvenance* e *RetrospectiveProvenance*, que descrevem o plano e a execução efetiva do pipeline, respectivamente; e (iii) *MLModel*, que armazena informações relacionadas ao modelo de AM gerado. Além disso, a identificação de execuções relacionadas é realizada quando existem execuções anteriores que utilizam o *pipeline P* e o conjunto de dados *D*, identificados pela relação *hasPrevious*, possibilitando a análise entre as iterações e a reutilização de execuções anteriores.

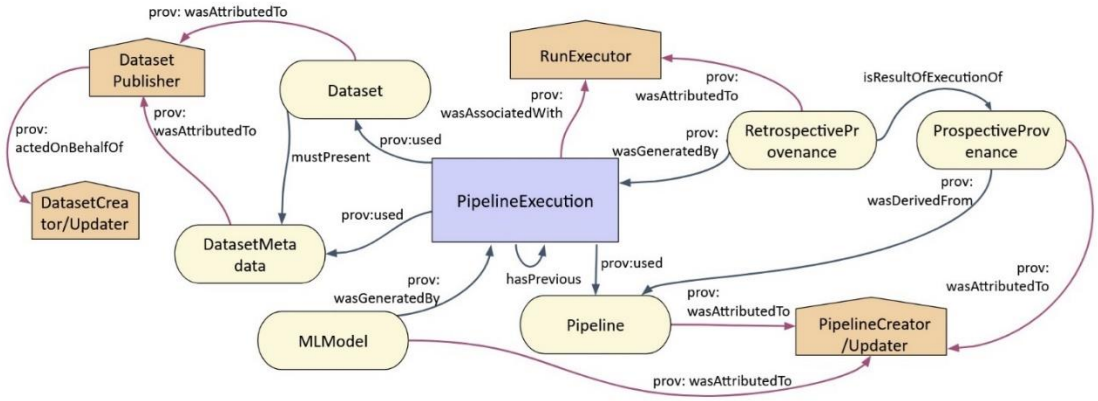
Figura 26 – PROVX: Modelo baseado no PROV(W3C).




Fonte: O Autor.

Considerando as subatividades realizadas na Atividade *Run*, a Figura 27 ilustra a fase de execução do *pipeline*. Essa subatividade é responsável por descrever as operações realizadas sobre os dados, desde o pré-processamento até a geração do modelo preditivo. A representação semântica contempla os artefatos utilizados (como o *pipeline* e os dados de entrada), os parâmetros aplicados, bem como os produtos gerados (modelo treinado e proveniência retrospectiva), que são relacionados aos agentes responsáveis. Esse detalhamento permite registrar o histórico das ações executadas e fornecer detalhes que possibilitam a comparação entre diferentes execuções.

Figura 27 – PROVX - Subatividade referente à fase de execução do *Pipeline*.

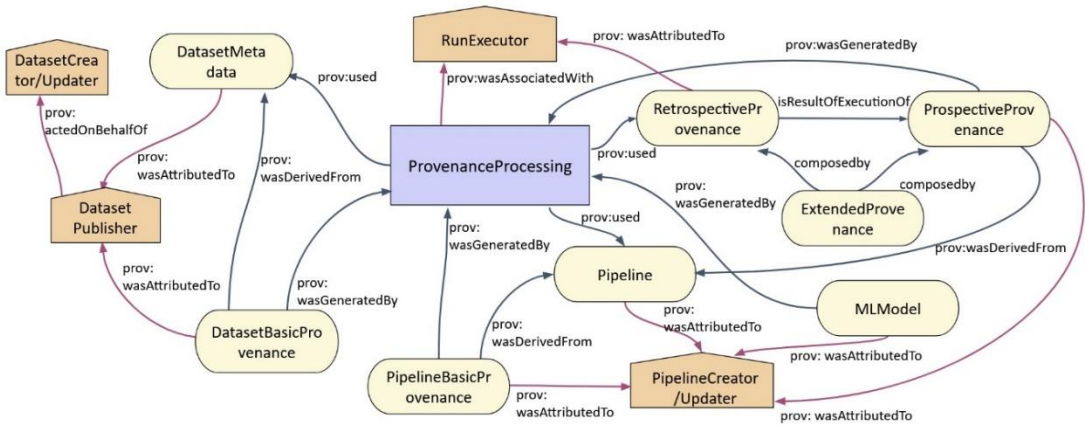



Fonte: O Autor.

A Figura 28 apresenta a subatividade responsável pela estruturação dos dados de proveniência, ocorrida após a execução do *pipeline*. Esta subatividade é voltada para a geração dos metadados de proveniência, abrangendo tanto a *Proveniência Prospectiva*, que documenta o plano de execução (com base na ontologia P-Plan), quanto a *Proveniência Retrospectiva*, que descreve as ações efetivamente realizadas. A modelagem semântica dessas informações permite capturar o

plano de execução e os resultados observados durante essa execução, sendo essencial para a verificação da conformidade do processo. A disponibilização desses metadados no formato RDF viabiliza a realização de consultas semânticas via SPARQL e a interoperabilidade entre diferentes sistemas, contribuindo para a transparência e o aprimoramento contínuo de *pipelines* de AM.

Figura 28 – PROVX - Subatividade referente à fase de Proveniência.




Fonte: O Autor.

### 4.3.4 Fase IV: Realização de Consultas

A fase de consultas aos dados de proveniência tem o objetivo de facilitar a análise detalhada desses dados de proveniência gerados durante a execução de *pipelines* de AM. Ela possibilita a extração de informações que podem apoiar tanto o desenvolvimento do modelo de AM quanto a definição do melhor modelo entre várias iterações de execuções do *pipeline* de AM.

Para a realização de consultas ao repositório de dados, é possível utilizar consultas predefinidas ou escrever novas consultas SPARQL, que permitem analisar os dados de proveniência e responder a perguntas sobre o desempenho e as configurações dos modelos treinados. Por exemplo, é possível identificar, em um conjunto de iterações de um *pipeline*, qual delas apresentou o melhor valor de acurácia e qual foi esse valor. Além disso, nesta fase, também é possível verificar os valores dos parâmetros e métricas de avaliação associados a um *pipeline* específico, sendo possível comparar diferentes execuções de um *pipeline*, permitindo rastrear as alterações realizadas ao longo da execução do *pipeline*, e proporcionar uma visão completa e detalhada do processo de modelagem.

## 4.4 EXEMPLO DE USO

Esta seção apresenta uma ilustração do passo a passo definido na abordagem proposta para a execução de *pipelines* de AM, nos quais dados de proveniência são utilizados para reexecutar e/ou ajustar atividades para novas execuções. O objetivo deste exemplo é demonstrar como os dados

de execuções anteriores podem ser utilizados para orientar ajustes para a seleção de atributos e como os dados estruturados são organizados e disponibilizados ao final da execução. Para isso, consideramos o *pipeline* ($P_1$) *Alzheimer's Disease Prediction on Patient*[14] para o problema de detecção da doença de Alzheimer em pacientes com base no histórico médico, utilizando algoritmos como *Random Forest* e *Decision Tree*, e apresentando as fases correspondentes à preparação de dados e à modelagem de AM. Nesse exemplo, os dados de proveniência de execuções anteriores do *pipeline* contribuem para novas execuções por meio da identificação e seleção dos atributos mais relevantes com base nas execuções passadas.

O conjunto de dados ($D_1$) utilizado neste *pipeline* de AM, Alzheimer's Disease Dataset[15] (Quadro 2), é composto por 35 atributos e 2149 instâncias, com informações sobre a saúde de pacientes, abrangendo dados demográficos, fatores de estilo de vida, histórico médico, medições clínicas, avaliações cognitivas e funcionais, sintomas e diagnósticos de Alzheimer, o que permite que cientistas e analistas de dados explorem os fatores associados à doença de Alzheimer, desenvolvam modelos preditivos e realizem análises estatísticas. Junto a este conjunto de dados também são disponibilizados seus metadados em formato JSON-LD. Nesse arquivo[12], são disponibilizados metadados que permitem descrever o conjunto de dados, incluindo metadados que descrevem a sua estrutura e metadados de proveniência básica do conjunto de dados.

Quadro 2 - Fragmento do conjunto de dados "*Alzheimer's Disease Dataset*".

| PatientID<br>patient_id | # Age<br>age | # Gender<br>gender | # BMI<br>bmi | ... | # Forgetfulness<br>forgetfulness | # Diagnosis<br>diagnosis | DoctorInCharge<br>Doctor_In_Charge |
|---|---|---|---|---|---|---|---|
| 4751 | 73 | 0 | 22.927749230993864 | ... | 0 | 0 | XXXConfid |
| 4752 | 89 | 0 | 26.82768119159602 | ... | 1 | 0 | XXXConfid |
| 4753 | 73 | 0 | 17.795882442817113 | ... | 0 | 0 | XXXConfid |
| 4754 | 74 | 1 | 33.80081704413547 | ... | 0 | 0 | XXXConfid |
| 4755 | 89 | 0 | 20.716973826446807 | ... | 0 | 0 | XXXConfid |
| 4756 | 86 | 1 | 30.626885546270938 | ... | 0 | 0 | XXXConfid |
| 4757 | 68 | 0 | 38.387621858169126 | ... | 1 | 0 | XXXConfid |
| 4758 | 75 | 0 | 18.776009409162835 | ... | 1 | 1 | XXXConfid |
| 4759 | 72 | 1 | 27.833188380332352 | ... | 0 | 0 | XXXConfid |

Fonte: ELKHAROUA (2023).

A abordagem proposta neste trabalho recebe como entrada o *pipeline* com o desenvolvimento do modelo preditivo (sem alterações a partir dos dados de proveniência, sendo considerado, assim, o *baseline*), o conjunto de dados utilizado no *pipeline* e o conjunto de

[14] https://www.kaggle.com/code/pilarkukuh/alzheimer-s-disease-prediction-on-patient
[15] https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset

metadados que descreve o conjunto de dados. A primeira execução do *pipeline* é considerada como *baseline*, pois não há execuções anteriores que permitam utilizar os dados de proveniência, e o *pipeline* não foi ajustado a partir desses dados.

Nesta execução, focando no modelo gerado pelo algoritmo de *Random Forest*, foram utilizados os parâmetros base do algoritmo. Em relação ao conjunto de dados, foi utilizada a variável *target Diagnosis* e os seguintes atributos para os dados de treinamento e teste: Age, Gender, Ethnicity, EducationLevel, BMI, Smoking, AlcoholConsumption, PhysicalActivity, DietQuality, SleepQuality, FamilyHistoryAlzheimers, CardiovascularDisease, Diabetes, Depression, HeadInjury, Hypertension, SystolicBP, DiastolicBP, CholesterolTotal, CholesterolLDL, CholesterolHDL, CholesterolTriglycerides, MMSE, FunctionalAssessment, MemoryComplaints, BehavioralProblems, ADL, Confusion, Disorientation, PersonalityChanges, DifficultyCompletingTasks e Forgetfulness.

Na execução *baseline*, os resultados obtidos para as métricas de acurácia, cobertura, precisão e *F1-score* variaram entre 0.82 e 0.96 (conforme observado na Tabela 2). Após a conclusão da execução do *pipeline*, ocorre a estruturação semântica dos metadados dessa execução, que são armazenados em um banco de dados NoSQL (GraphDB), o qual contém os metadados de proveniência básica, proveniência estendida e os dados do modelo gerado, como o algoritmo e seus parâmetros.

Na segunda execução, são utilizados os dados de proveniência obtidos a partir da identificação de execução com a melhor avaliação do modelo de AM. Neste cenário, o *pipeline* foi ajustado com os atributos indicados pela técnica RFE, que foi aplicada em conjunto com a reexecução das atividades, considerando as atividades da preparação dos dados e o algoritmo de classificação utilizado (com os mesmos parâmetros). A RFE foi aplicada de forma que os atributos selecionados sejam determinados de maneira automatizada e com base no desempenho do modelo, o que permite otimizar o conjunto de atributos para melhorar os resultados do modelo.

Com a aplicação da RFE identifica-se o conjunto de atributos que permite otimizar a acurácia do modelo de classificação, o valor dessa melhor acurácia identificado pela RFE também é comparado com o resultado do modelo da execução relacionada (obtida a partir da proveniência retrospectiva) para verificar se é apresentado algum ganho com a realização da atividade. A partir disso, foram recomendados os seguintes atributos para otimizar o desempenho do modelo: PhysicalActivity, DietQuality, CholesterolTotal, CholesterolHDL, CholesterolTriglycerides, MMSE, FunctionalAssessment, MemoryComplaints, BehavioralProblems, ADL, Diagnosis.

O Quadro 3 se refere a uma amostra dos dados de proveniência prospectiva das duas execuções do *pipeline* de *Alzheimer's Disease*. Cada linha do quadro descreve uma etapa do *pipeline*, identificando a execução específica, a proveniência básica do *pipeline* correspondente, a

origem da proveniência prospectiva, além do identificador e o algoritmo utilizado em cada etapa. Esses dados estruturados ficam persistidos no repositório de dados, sendo possível a realização de consultas e utilização desses dados para o ajuste de futuras execuções (conforme indicado na Seção 4.3.1 e na Seção 4.3.2).

Quadro 3 – Amostra dos Dados Referentes à Proveniência Prospectiva para as duas Execuções do *Pipeline* sobre *Alzheimer's Disease*.

| | execucao | pipelineBasicProvenance | wasDerivedFrom | id | step |
|---|---|---|---|---|---|
| 1 | _:5.1.1 | provx:PipelineBasicProvenance/5.1.1 | provx:Pipeline/5.1.1 | "1" | provx:PreparationAlgorithm/5.1.1 |
| 2 | _:5.1.1 | provx:PipelineBasicProvenance/5.1.1 | provx:Pipeline/5.1.1 | "2" | provx:ModelingAlgorithm/5.1.1 |
| 3 | _:6.1.1 | provx:PipelineBasicProvenance/6.1.1 | provx:Pipeline/6.1.1 | "1" | provx:PreparationAlgorithm/6.1.1 |
| 4 | _:6.1.1 | provx:PipelineBasicProvenance/6.1.1 | provx:Pipeline/6.1.1 | "2" | provx:ModelingAlgorithm/6.1.1 |

Fonte: O Autor.

Conforme observado na Tabela 2, ao comparar os resultados obtidos nas duas execuções realizadas, foi apresentado um aumento de até cerca de +8% nas métricas que avaliam o desempenho do modelo de classificação. Na segunda execução, a inclusão automatizada da etapa de seleção de atributos, por meio da técnica RFE e com base nos dados provenientes de execuções anteriores, possibilitou a geração de um modelo de aprendizado de máquina mais eficiente, o que se refletiu positivamente nas métricas avaliadas. Além disso, os benefícios promovidos pela abordagem proposta não se limitam apenas aos resultados apresentados no desempenho dos modelos gerados, sendo possível também realizar análises sobre os dados de proveniência obtidos durante as execuções, o que permite responder a questionamentos. Entre as possibilidades de análise, destacam-se questões como qual execução de um *pipeline* apresentou o melhor desempenho, quais parâmetros e atributos foram utilizados nessa execução, quais *pipelines* utilizaram determinado conjunto de dados e qual o agente responsável por esse conjunto de dados. O cenário experimental, conjuntos de dados utilizados, configuração de ambiente e resultados obtidos são detalhados no capítulo a seguir.

Tabela 2 – Métricas de desempenho avaliadas durante as execuções do *pipeline* sobre *Alzheimer's Disease*.

| **Execução** | **Acurácia** | **Precisão** | **Cobertura** | ***F1-score*** |
|---|---|---|---|---|
| Primeira Execução (*baseline)* | 0,9256 | 0,9618 | 0,8235 | 0,8873 |
| Segunda Execução (utilizando dados de proveniência) | 0,9535 | 0,9650 | 0,9020 | 0,9324 |

Fonte: O Autor.

## 4.5 CONSIDERAÇÕES

Ao comparar essa proposta com os trabalhos apresentados no Quadro 1, observa-se que, embora as propostas de SCHLEGEL & SATTLER (2023), OLIVEIRA et al. (2024) e os demais trabalhos relacionados forneçam contribuições importantes para o uso de dados de proveniência em *pipelines* de AM, essas abordagens não abordam de forma integrada e sistemática a reexecução de atividades e a utilização dos dados de proveniência para ajustar o *pipeline* de AM em execuções futuras.

O trabalho de OLIVEIRA et al. (2024), que foca na relação entre os dados de proveniência com os atributos importantes para o treinamento do modelo, não realiza a aplicação direta desses dados para influenciar em avaliações futuras do modelo.

A abordagem de KERZEL et al. (2021) que permite a captura e visualização de dados de proveniência, e a de SCHELTER et al. (2023) que trata de problemas nos modelos gerados, também não exploram a reexecução de atividades, a estruturação semântica completa dos dados de proveniência, nem seu uso otimizado para ajustes nos modelos.

Em relação aos trabalhos que estendem a PROV-O, as abordagens buscam adaptar seus elementos a contextos mais específicos de AM. No entanto, não foi observada uma extensão dessa ontologia que integre os dados de proveniência de todos os artefatos envolvidos na execução, como o *pipeline* e o conjunto de dados, com dados de proveniência prospectiva e retrospectiva, como propõe a abordagem deste trabalho, além de permitir relacionar as diferentes execuções.

A proposta apresentada neste trabalho se destaca por promover a reexecução e o ajuste de atividades, como a seleção de atributos, por meio de dados de proveniência. Para isso, utiliza a ontologia PROVX de modo integrado à reexecução de atividades em execuções posteriores de *pipelines*. O capítulo a seguir descreve os experimentos realizados, bem como os resultados obtidos a partir de sua execução.

# 5 EXPERIMENTOS E RESULTADOS

Neste capítulo, são apresentados os resultados obtidos com a implementação do protótipo desenvolvido para apoiar a abordagem proposta, bem como por meio dos experimentos conduzidos para sua avaliação. A Seção 5.1 descreve o protótipo nFlowX, desenvolvido para a abordagem proposta, incluindo a arquitetura que detalha os módulos e as funcionalidades do protótipo. Na Seção 5.2, é descrito o cenário utilizado para a realização dos experimentos. Em seguida, a Seção 5.3 discorre sobre a avaliação experimental, incluindo a metodologia adotada e as métricas de avaliação empregadas, e, na Seção 5.4, é realizada a análise dos resultados obtidos. Por fim, na Seção 5.5 são apresentadas as considerações finais deste capítulo.

## 5.1 PROTÓTIPO nFlowX

Nesta seção, é apresentado o protótipo desenvolvido (nFlowX) para apoiar a abordagem proposta. Com o uso do nFlowX, usuários, como cientistas de dados e engenheiros de AM, podem desenvolver *pipelines* de AM utilizando os dados de proveniência capturados durante execuções anteriores do *pipeline*, visando melhorar o desempenho do modelo de AM em questão. A implementação do protótipo viabiliza a reexecução de atividades realizadas em iterações anteriores, a análise dos dados de proveniência para responder a questões relacionadas ao desenvolvimento do modelo de AM, além de utilizar essas informações para ajustar atividades desempenhadas no *pipeline* de AM.

### 5.1.1 Arquitetura

O nFlowX foi estruturado considerando uma arquitetura que apresenta três camadas (Figura 29), contendo a camada do usuário, a camada lógica e a camada de dados, definidas a seguir:

- Camada do Usuário: consiste na interface da aplicação onde o usuário interage com o protótipo, fazendo uso das funcionalidades;
- Camada Lógica: composta pelos seguintes módulos disponibilizados pela aplicação:
    - Módulo de Execução do *Pipeline*: essa é a funcionalidade principal do protótipo, onde o usuário pode ajustar, executar e obter dados de proveniência de *pipelines* de AM. Para utilizar essa funcionalidade, é necessário fornecer o *pipeline*, o conjunto de dados que será utilizado pelo *pipeline* e os metadados do conjunto de dados. Sendo composto por três submódulos: (i) ajuste do *pipeline*: onde o usuário poderá ajustar o *pipeline* a partir dos dados de proveniência; (ii) execução do *pipeline*: permite executar o *pipeline*, o que é realizado utilizando a ferramenta noWorkflow para que sejam capturados os dados de proveniência durante a execução, devido às

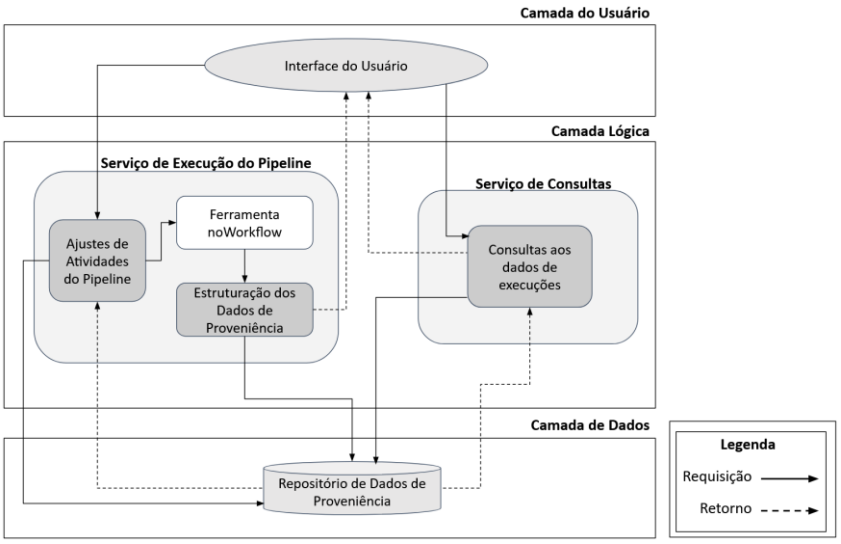
funcionalidades oferecidas, conforme explicado na Seção 2.2. Para executar o *pipeline* com esta ferramenta, é utilizado o comando `now run -v [pipeline] [conjunto de dados]`, especificando o *pipeline* e o conjunto de dados - esse comando permite a execução do *pipeline* e fornece um feedback geral sobre o processo. Durante a execução, as atividades relacionadas ao *pipeline*, como o uso de variáveis, funções e acesso a arquivos, também são registradas. Com a conclusão da execução, os dados de proveniência são obtidos por meio de comandos como `now prov [id execução]`, que exporta a proveniência da execução, e `now show -a [id execução]`, que retorna o detalhamento da execução e de chamadas de funções, incluindo status e duração da execução, e a data e horário de ativação de cada função; e (iii) estruturação dos dados de proveniência: nesse submódulo, é utilizada a extensão PROVX para estruturar semanticamente os dados obtidos a partir da execução do *pipeline* e os metadados referentes à proveniência básica do conjunto de dados e do *pipeline*, sendo armazenados no repositório de dados.

    - Módulo de Consultas: permite consultar os dados de proveniência armazenados no repositório de dados para a obtenção de informações sobre as execuções dos *pipelines* de AM, podendo responder a questionamentos acerca dos resultados obtidos sobre os modelos e da composição desses *pipelines*.

- Camada de Dados: os dados de proveniência gerados durante cada execução são armazenados em um repositório de dados orientado a grafos e disponibilizados para acesso pelo módulo de consulta. Os dados de proveniência de cada execução são compostos por metadados de proveniência básica, metadados de proveniência estendida e os metadados referentes ao modelo de aprendizado de máquina. O sistema utilizado foi o GraphDB[16], uma plataforma NoSQL, que permite o armazenamento de grafos e as consultas sobre esses dados utilizando a linguagem SPARQL, além de ser acessível por meio de biblioteca para conexão com a aplicação (que foi desenvolvida em Python).

[16] https://graphdb.ontotext.com/

Figura 29- Visão geral da arquitetura do nFlowX.




Fonte: O Autor.

### 5.1.2 Implementação do nFlowX

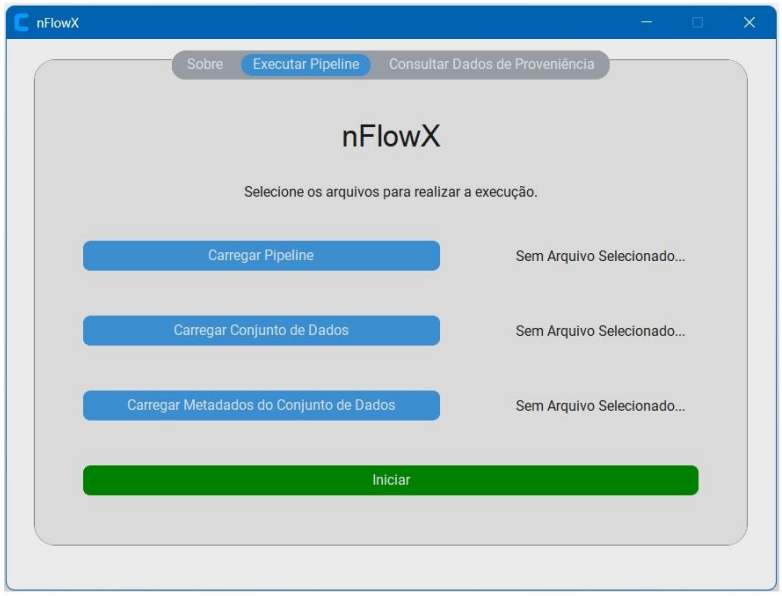
O protótipo foi desenvolvido utilizando a linguagem de programação Python, tendo a interface desenvolvida utilizando o CustomTkinter[17], que é uma extensão personalizada da biblioteca TKinter[18], sendo a biblioteca padrão de interface gráfica do usuário (GUI) para Python. Para o acesso ao GraphDB, foi utilizada a biblioteca SPARQLWrapper[19], que permite acesso ao *endpoint* SPARQL disponibilizado por meio do repositório de dados.

#### *5.1.2.1 Interface do nFlowX*

O protótipo nFlowX apresenta uma interface amigável que permite que os usuários acessem os módulos disponibilizados pela aplicação. Como mostrado na Figura 30, o usuário tem acesso ao menu com os módulos disponíveis, os quais correspondem às opções para execução do *pipeline* e para realizar consultas ao repositório de dados. No módulo de execução do *pipeline*, é requisitado que o usuário realize o *upload* do *pipeline* e dos arquivos referentes ao *pipeline* de AM, a saber: o conjunto de dados e o arquivo de metadados do conjunto de dados (Seção 5.1.1).

[17] https://customtkinter.tomschimansky.com/
[18] https://docs.python.org/3/library/tkinter.html
[19] https://sparqlwrapper.readthedocs.io/en/latest/

Figura 30 - nFlowX - Módulo de Execução do *pipeline*.




Fonte: O Autor.

No módulo de consultas, o usuário pode utilizar consultas predefinidas ou desenvolver novas consultas SPARQL. Entre as consultas predefinidas (detalhadas no Apêndice C), podemos destacar: a partir da identificação do *pipeline* (por meio dos seus metadados de proveniência), podemos consultar o desempenho do modelo de predição gerado pelo *pipeline*, considerando o desempenho de todas as iterações executadas, e consultar a iteração do *pipeline* que apresentou o melhor desempenho; a partir da identificação do conjunto de dados (por meio dos seus metadados de proveniência), é possível consultar as execuções relacionadas a um determinado conjunto de dados; e consultar os dados de proveniência de uma iteração específica.

## 5.2 CENÁRIO DOS EXPERIMENTOS

Nesta seção, são apresentados os elementos essenciais para a realização dos experimentos, considerando os *pipelines*, incluindo seus respectivos problemas de classificação e os conjuntos de dados utilizados, além da configuração do ambiente para execução dos experimentos.

### 5.2.1 Problemas de Classificação e *Pipelines* de AM

Para a realização dos experimentos, foram obtidos, a partir da plataforma Kaggle[20], cinco *pipelines* de AM, sendo cada um associado a um conjunto de dados diferente. Para os *pipelines*, foram considerados aqueles que aplicam algoritmos de classificação e apresentam pelo menos as etapas de preparação de dados e modelagem de AM. Além disso, foram utilizados *pipelines* que

[20] https://www.kaggle.com/

apresentam variações nas atividades de preparação de dados realizadas e que apresentam variação na quantidade de atributos disponibilizados pelos conjuntos de dados.

Para os algoritmos de classificação, foram considerados modelos amplamente utilizados na literatura, como *Decision Tree*[21] e *Random Forest*[22], devido à sua interpretabilidade e capacidade de lidar com diferentes tipos de dados. Para a identificação das fases no *pipeline*, foi utilizada uma notação em forma de comentário na linguagem Python, por exemplo: fase de preparação de dados: `#fase:preparação_dados`; fase de modelagem: `#fase:modelagem_AM`. Por fim, para cada conjunto de dados, também foram obtidos seus respectivos arquivos de metadados (disponibilizados na plataforma).

A seguir, são listados os problemas de classificação dos *pipelines* e os conjuntos de dados selecionados:

- *Pipeline* - *Breast Cancer*: tem o objetivo de classificar se o tumor da paciente é benigno ou maligno, a partir de atributos como as medidas relacionadas à forma, tamanho e textura das células do tumor.
    - Conjunto de Dados - *Breast Cancer*[23]: conjunto de dados que apresenta informações detalhadas sobre pacientes com câncer de mama, sendo composto por 32 atributos e 569 instâncias. Esse conjunto de dados viabiliza a análise dos dados e o desenvolvimento de modelos preditivos que auxiliam na detecção e classificação do câncer de mama.
- *Pipeline* - *NASA: Asteroids Classification*: permite classificar os asteroides em potencialmente perigosos ou não perigosos.
    - Conjunto de Dados - *NASA: Asteroids Classification*[24]: é um conjunto de dados que disponibiliza a Classificação de Asteroides, sua fonte é o *Near-Earth Object* (NEO)[25] da NASA, um serviço web RESTful para informações de Asteroides próximos à Terra. O conjunto de dados consiste em 40 atributos e 4.687 instâncias, fornecendo informações como nome, magnitude, data de aproximação, velocidades relativas e órbita.
- *Pipeline* - *Alzheimer's Disease*: realiza a detecção da doença de Alzheimer em pacientes com base em seu histórico médico.

[21] https://scikit-learn.org/stable/modules/tree.html
[22] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[23] https://www.kaggle.com/datasets/rahmasleam/breast-cancer
[24] https://www.kaggle.com/datasets/lovishbansal123/nasa-asteroids-classification
[25] https://cneos.jpl.nasa.gov/

- Conjunto de Dados - *Alzheimer's Disease Dataset*[26]: conjunto de dados que permite explorar fatores associados à doença de Alzheimer, sendo composto por dados de 2.149 pacientes e 35 atributos, apresentando detalhes como: dados demográficos, fatores de estilo de vida, histórico médico, medidas clínicas, avaliações cognitivas e funcionais, sintomas e diagnóstico da doença de Alzheimer.

- *Pipeline* - *Comprehensive Diabetes Clinical*: a partir de dados demográficos e de saúde, permite classificar se cada instância representa um caso de diabetes ou não.
  - Conjunto de Dados - *Comprehensive Diabetes Clinical Dataset*[27]: conjunto de dados que viabiliza análises e modelagem sobre a diabetes, apresenta 100.000 instâncias e 16 atributos, que incluem informações sobre gênero, idade, localização, raça, hipertensão, doença cardíaca, histórico de tabagismo, IMC, nível de HbA1c, nível de glicose no sangue e status de diabetes.
- *Pipeline* - *Airlines Delay*: realiza a classificação para a previsão de atrasos em voos.
  - Conjunto de Dados - *Airlines Delay*[28]: conjunto de dados que fornece dados sobre voos e atrasos, sendo composto por 53.9382 instâncias e 8 atributos, como: aeroporto de origem e de destino, horário programado de partida e de chegada, horário real de partida e de chegada, condições climáticas e dia da semana.

Para a seleção desses conjuntos de dados foi considerada a variação de cenários e a complexidade dos problemas abordados, incluindo desde a detecção de doenças até a previsão de eventos no setor de transporte. Além disso, a escolha dos *pipelines* relacionados aos conjuntos de dados, considerando algoritmos de aprendizado supervisionado semelhantes, visa obter resultados mais consistentes e comparáveis, facilitando a análise de como as transformações dos dados impactam a avaliação e a comparação entre os modelos.

### 5.2.2 Configuração do Ambiente para Execução dos Experimentos

Todos os experimentos foram conduzidos em uma máquina com processador Intel core i7 de 1,99 GHz com 16GB de RAM, Sistema Operacional Windows 10. Foi utilizado o interpretador Python versão 3.7. Para a captura dos dados de proveniência, foi utilizada a versão 2.0 da ferramenta noWorkflow e, durante a execução dos *pipelines*, foram utilizadas bibliotecas que apoiam o desenvolvimento de modelos de AM, como: NumPy[29], Pandas e Sklearn[30].

[26] https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset
[27] https://www.kaggle.com/datasets/priyamchoksi/100000-diabetes-clinical-dataset
[28] https://www.kaggle.com/datasets/ulrikthygepedersen/airlines-delay/
[29] https://numpy.org/
[30] https://scikit-learn.org/stable/

## 5.3 AVALIAÇÃO EXPERIMENTAL

Nestes experimentos, foi considerada a hipótese de que a utilização de dados de proveniência de execuções anteriores melhora a avaliação do modelo preditivo, sendo:

- $H_0$: Quando a avaliação de modelos de AM que passaram por ajustes bem-sucedidos com base nos dados de proveniência é comparada à avaliação de modelos sem esses ajustes, não se observa melhoria nos resultados do modelo; e
- $H_1$: Quando a avaliação de modelos de AM que passaram por ajustes bem-sucedidos com base nos dados de proveniência é comparada à avaliação de modelos sem esses ajustes, observa-se melhoria nos resultados do modelo.

Para a realização dos experimentos, os *pipelines* foram submetidos ao seguinte cenário (Cenário 01): Execuções que utilizam dados de proveniência considerando o *pipeline P* de AM e o conjunto de dados *D* da execução *baseline* (indicada na Seção 4.4), para ajustar atividades por meio dos dados de proveniência. Para efeito de controle experimental, a primeira execução (*baseline*) dos *pipelines* é realizada sem a garantia de que tenha ocorrido a seleção ideal de atributos. Essa decisão visou possibilitar a observação do impacto da aplicação orientada da técnica RFE com base em dados de proveniência. Contudo, em cenários práticos, essa simplificação nem sempre se aplica, sendo possível que o *pipeline* já utilize técnicas de seleção com ajustes ótimos. Mesmo assim, a proposta apresentada ainda pode ser utilizada para comparar e apoiar execuções futuras com base no histórico de execuções.

No Cenário 01, a atividade de seleção de atributos foi ajustada com base nos dados de proveniência. Para isso, foram seguidas as instruções estabelecidas na Seção 4.3.2 e no Algoritmo I, implementadas no protótipo nFlowX. Para cada *pipeline*, foi verificada, no repositório de dados, a existência de execuções relacionadas. Em seguida, os dados de proveniência foram utilizados para aplicar a RFE, utilizando o mesmo modelo e os atributos da execução relacionada com melhor avaliação, com o objetivo de obter um subconjunto de atributos que melhorasse a avaliação desse modelo. Após essa etapa, o *pipeline* foi ajustado com a seleção dos atributos recomendados pela RFE, finalizado com a execução do *pipeline* atualizado e o armazenamento dos dados de proveniência da nova execução.

### 5.3.1 Comparação avaliativa de modelos de Aprendizado de Máquina

Para comparar os resultados dos modelos de AM obtidos, considerando a execução do *baseline* e do Cenário 1 de cada *pipeline*, foi aplicada, inicialmente, a técnica de *k-fold cross-validation* (KOHAVI, 1995), com $K = 10$, devido ao equilíbrio entre viés e variância na estimativa do desempenho de modelos, e apresentar custo computacional viável com resultados estatisticamente

confiáveis. Onde, a cada iteração, k−1 *folds* são usados para treino e um *fold* é usado para teste, repetindo esse processo k vezes. Resultando em um conjunto de 10 valores para as métricas de acurácia, precisão, cobertura e *F1-score*.

Em seguida, foi realizada avaliação estatística para verificar a significância das diferenças observadas entre o *baseline* e o Cenário 1 dos *pipelines*. Considerando o contexto do estudo e a natureza das amostras pareadas entre o *baseline* e o Cenário 1, a aplicação do teste de hipótese de Wilcoxon (WILCOXON, 1945) é apropriada para avaliar a significância da diferença entre os resultados obtidos por ambos os modelos. A partir das métricas que avaliam os modelos, o teste permite verificar se as diferenças entre os resultados dos modelos são significativas do ponto de vista estatístico, considerando as hipóteses definidas na seção anterior.

## 5.4 RESULTADOS E DISCUSSÕES

Nesta seção são avaliados os resultados obtidos durante a execução da abordagem, considerando os *pipelines* indicados na Seção 5.2.1 e as especificações da Seção 5.3. Para cada um dos cinco *pipelines*, a partir da execução *baseline* e do Cenário 1, foi aplicada a abordagem *k-fold*, na qual cada modelo obteve 10 avaliações para as métricas de acurácia, cobertura, precisão e *F1-score*, indicados nas Tabelas 3 a 7.

Tabela 3 – *Pipeline* - *Breast Cancer*: Resultados das execuções aplicando a estratégia *k-fold*.

| | **Acurácia** | | **Precisão** | | **Cobertura** | | ***F1-score*** | |
|---|---|---|---|---|---|---|---|---|
| ***k*** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** |
| **1** | 0.9122 | 0.9298 | 0.9047 | 0.9090 | 0.8636 | 0.9090 | 0.8837 | 0.9090 |
| **2** | 0.8596 | 0.9298 | 0.85 | 1.0 | 0.7727 | 0.8181 | 0.8095 | 0.9 |
| **3** | 0.9298 | 0.92982 | 0.9474 | 0.9474 | 0.8571 | 0.8571 | 0.9 | 0.9 |
| **4** | 0.8421 | 0.9298 | 0.7727 | 0.9474 | 0.8095 | 0.8571 | 0.7906 | 0.9 |
| **5** | 0.9824 | 1.0 | 0.9545 | 1.0 | 1.0 | 1.0 | 0.9767 | 1.0 |
| **6** | 0.9123 | 0.9474 | 0.9 | 0.9090 | 0.8571 | 0.9523 | 0.8780 | 0.9302 |
| **7** | 0.8596 | 0.9473 | 0.8095 | 0.95 | 0.8095 | 0.9047 | 0.8095 | 0.9268 |
| **8** | 0.9473 | 1.0 | 0.9090 | 1.0 | 0.9523 | 1.0 | 0.9302 | 1.0 |
| **9** | 0.9123 | 0.9649 | 0.8077 | 0.9130 | 1.0 | 1.0 | 0.8936 | 0.9545 |
| **10** | 0.9464 | 0.9286 | 0.95 | 0.869 | 0.9047 | 0.9523 | 0.9268 | 0.9090 |

Fonte: O Autor.

Tabela 4 – *Pipeline* - NASA: *Asteroids Classification*: Resultados das execuções aplicando a estratégia *k-fold*.

| | **Acurácia** | | **Precisão** | | **Cobertura** | | ***F1-score*** | |
|---|---|---|---|---|---|---|---|---|
| ***k*** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** |
| **1** | 0.9829 | 0.9850 | 0.9722 | 1.0 | 0.9210 | 0.9078 | 0.9459 | 0.9517 |
| **2** | 0.9893 | 0.9978 | 1.0 | 1.0 | 0.9342 | 0.9868 | 0.9659 | 0.9933 |
| **3** | 0.9808 | 0.9936 | 1.0 | 1.0 | 0.8815 | 0.9605 | 0.9370 | 0.9798 |
| **4** | 0.9829 | 0.9893 | 1.0 | 0.9863 | 0.8947 | 0.9473 | 0.9444 | 0.9664 |
| **5** | 0.9914 | 0.9936 | 1.0 | 1.0 | 0.9473 | 0.9605 | 0.9729 | 0.9798 |
| **6** | 0.9893 | 0.9957 | 1.0 | 1.0 | 0.9333 | 0.9733 | 0.9655 | 0.9864 |
| **7** | 0.9872 | 0.9914 | 1.0 | 1.0 | 0.92 | 0.9466 | 0.9583 | 0.9726 |
| **8** | 0.9871 | 0.9935 | 0.9859 | 0.9864 | 0.9333 | 0.9733 | 0.9589 | 0.9798 |
| **9** | 0.9914 | 0.9957 | 1.0 | 1.0 | 0.9466 | 0.9733 | 0.9726 | 0.9864 |
| **10** | 0.9957 | 0.9957 | 1.0 | 1.0 | 0.9733 | 0.9733 | 0.9864 | 0.9864 |

Fonte: O Autor.

Tabela 5 – *Pipeline - Alzheimer's Disease*: Resultados das execuções aplicando a estratégia *k-fold*.

| | **Acurácia** | | **Precisão** | | **Cobertura** | | ***F1-score*** | |
|---|---|---|---|---|---|---|---|---|
| ***k*** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** |
| **1** | 0.9534 | 0.9860 | 0.9852 | 0.9866 | 0.8815 | 0.9736 | 0.9305 | 0.9801 |
| **2** | 0.9720 | 0.9906 | 0.9861 | 0.9868 | 0.9342 | 0.9868 | 0.9594 | 0.9868 |
| **3** | 0.9813 | 0.9767 | 0.9864 | 0.9610 | 0.9605 | 0.9736 | 0.9733 | 0.9673 |
| **4** | 0.9720 | 0.9906 | 0.9729 | 0.9743 | 0.9473 | 1.0 | 0.9599 | 0.9870 |
| **5** | 0.9813 | 0.9813 | 0.9864 | 0.9864 | 0.9605 | 0.9605 | 0.9733 | 0.9733 |
| **6** | 0.9581 | 0.9813 | 0.9718 | 0.9736 | 0.9078 | 0.9736 | 0.9387 | 0.9736 |
| **7** | 0.9953 | 1.0 | 1.0 | 1.0 | 0.9868 | 1.0 | 0.9933 | 1.0 |
| **8** | 0.9581 | 0.9906 | 0.9718 | 0.9743 | 0.9078 | 1.0 | 0.9387 | 0.9870 |
| **9** | 0.9348 | 0.9534 | 0.9696 | 0.9714 | 0.8421 | 0.8947 | 0.9014 | 0.9315 |
| **10** | 0.6775 | 0.6775 | 0.5686 | 0.5660 | 0.3815 | 0.3947 | 0.4566 | 0.4651 |

Fonte: O Autor.

Tabela 6 – *Pipeline - Comprehensive Diabetes Clinical*: Resultados das execuções aplicando a estratégia *k-fold*.

| | **Acurácia** | | **Precisão** | | **Cobertura** | | ***F1-score*** | |
|---|---|---|---|---|---|---|---|---|
| ***k*** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** | **Baseline** | **Cenário 1** |
| **1** | 0.9522 | 0.9728 | 0.7074 | 1.0 | 0.7482 | 0.6811 | 0.7272 | 0.8103 |
| **2** | 0.9530 | 0.9719 | 0.7128 | 1.0 | 0.7505 | 0.6705 | 0.7312 | 0.8028 |
| **3** | 0.9533 | 0.9728 | 0.7142 | 1.0 | 0.7529 | 0.6811 | 0.7331 | 0.8103 |
| **4** | 0.95 | 0.9691 | 0.6934 | 1.0 | 0.74 | 0.6376 | 0.7159 | 0.7787 |
| **5** | 0.955 | 0.9722 | 0.7323 | 1.0 | 0.7435 | 0.6741 | 0.7378 | 0.8053 |
| **6** | 0.951 | 0.9725 | 0.6981 | 1.0 | 0.7482 | 0.6776 | 0.7223 | 0.8078 |
| **7** | 0.9502 | 0.9712 | 0.6928 | 1.0 | 0.7458 | 0.6623 | 0.7184 | 0.7968 |
| **8** | 0.9522 | 0.9716 | 0.7065 | 1.0 | 0.7505 | 0.6670 | 0.7278 | 0.8002 |
| **9** | 0.9478 | 0.9704 | 0.6833 | 1.0 | 0.7211 | 0.6529 | 0.7017 | 0.7900 |
| **10** | 0.9522 | 0.9732 | 0.7011 | 1.0 | 0.7647 | 0.6858 | 0.7315 | 0.8136 |

Fonte: O Autor.

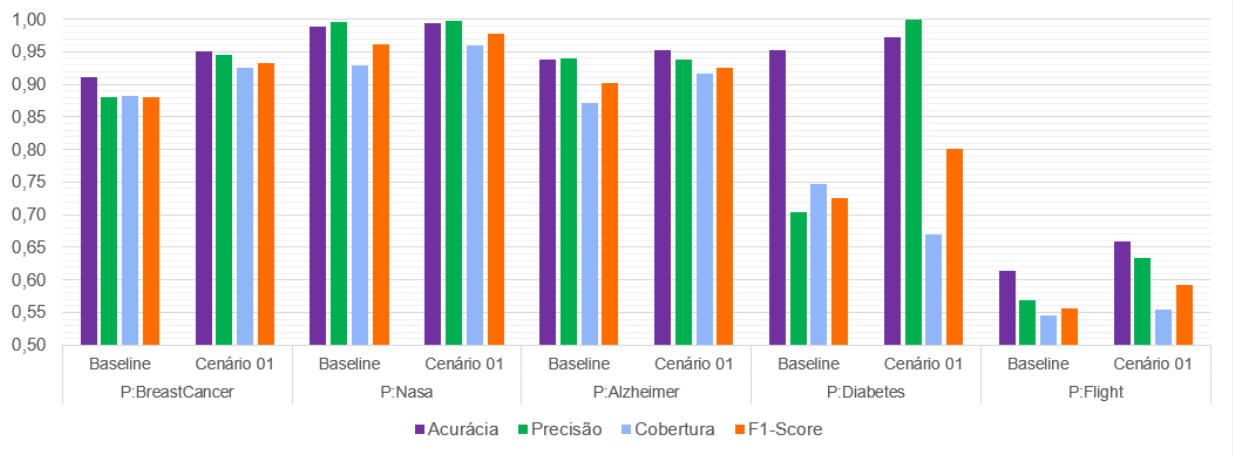
Tabela 7 – *Pipeline - Airlines Delay*: Resultados das execuções aplicando a estratégia *k-fold*.

| *k* | Acurácia | | Precisão | | Cobertura | | *F1-score* | |
|---|---|---|---|---|---|---|---|---|
| | Baseline | Cenário 1 | Baseline | Cenário 1 | Baseline | Cenário 1 | Baseline | Cenário 1 |
| **1** | 0.6108 | 0.6598 | 0.5662 | 0.6352 | 0.5406 | 0.5550 | 0.5531 | 0.5924 |
| **2** | 0.6131 | 0.6591 | 0.5687 | 0.6346 | 0.5439 | 0.5535 | 0.5560 | 0.5912 |
| **3** | 0.614 | 0.6611 | 0.5698 | 0.6368 | 0.5475 | 0.5566 | 0.5584 | 0.5940 |
| **4** | 0.6098 | 0.6592 | 0.5647 | 0.6338 | 0.5414 | 0.5563 | 0.5528 | 0.5925 |
| **5** | 0.6121 | 0.6575 | 0.5671 | 0.6319 | 0.5457 | 0.5533 | 0.5562 | 0.5900 |
| **6** | 0.6171 | 0.6607 | 0.5734 | 0.6361 | 0.5488 | 0.5568 | 0.5608 | 0.5939 |
| **7** | 0.6134 | 0.6586 | 0.5689 | 0.6331 | 0.5459 | 0.5557 | 0.5571 | 0.5919 |
| **8** | 0.6139 | 0.6578 | 0.5695 | 0.6319 | 0.5461 | 0.5552 | 0.5575 | 0.5911 |
| **9** | 0.6150 | 0.6605 | 0.5710 | 0.6376 | 0.5451 | 0.5510 | 0.5578 | 0.5912 |
| **10** | 0.6139 | 0.6584 | 0.5693 | 0.6326 | 0.5472 | 0.5563 | 0.5580 | 0.5920 |

Fonte: O Autor.

Conforme observado no gráfico da Figura 31, a partir dos valores obtidos nas tabelas apresentadas, foram calculadas as médias de cada uma das métricas para cada *pipeline*. Esses valores médios permitiram uma análise mais abrangente do impacto da abordagem proposta, facilitando a comparação entre o *baseline* e o Cenário 1.

Figura 31 - Comparativo dos resultados obtidos a partir das duas execuções do *pipeline* de AM – *Baseline* e Cenário 01.




Fonte: O Autor.

Ao comparar os resultados, foi observado que, em todos os *pipelines*, as métricas de acurácia e *F1-score* apresentaram melhorias. Nos *pipelines* que registravam para essas métricas valores de até aproximadamente 94%, houve um aumento mais significativo nos resultados. De forma geral, para essas duas métricas, os *pipelines* mostraram uma melhora de até cerca de +8%. Por outro lado, no *pipeline* de Diabetes, quando comparado com o baseline, a métrica de cobertura diminuiu cerca de -7,5%, mas houve um aumento nas métricas de acurácia, precisão e *F1-score*, o que indica que o modelo cometeu menos erros ao prever a classe positiva, mas pode não ter conseguido

detectar todas as instâncias positivas. Além disso, nesse *pipeline*, a precisão alcançou 100%, o que sugere um ajuste excessivo do modelo aos dados de treinamento, caracterizando um *overfitting*.

No *pipeline* de *Alzheimer*, a métrica de precisão apresentou valor um pouco menor (-0,0019), mas houve uma melhora de +0,0447 na métrica de cobertura, o que indica que, apesar da leve diminuição na precisão, o modelo conseguiu identificar melhor as instâncias positivas, o que é relevante nesse contexto.

Ainda que as melhorias observadas nos modelos de AM possam ser associadas à aplicação da técnica de RFE, o diferencial desta proposta está centrado na utilização dos dados de proveniência como mecanismo de apoio à execução dessa técnica. Neste trabalho, a aplicação da RFE ocorre de forma orientada a partir dos dados de execuções anteriores do *pipeline*, permitindo o direcionamento, a automatização do processo e a reexecução das atividades realizadas anteriormente. Dessa forma, os resultados apresentados refletem a eficácia da técnica de seleção de atributos e, sobretudo, evidenciam a relevância da contribuição dos dados de proveniência para a realização dos ajustes nos *pipelines* de AM.

### 5.4.1 Análises acerca do uso da proveniência das execuções dos *pipelines* de AM

Nesta seção, são realizadas avaliações sobre os dados de proveniência das execuções dos *pipelines* de AM. A abordagem proposta facilita a análise do processo de desenvolvimento do *pipeline*, além de possibilitar a reexecução das atividades realizadas, trazendo mais transparência e confiabilidade aos experimentos. Com a captura e uso de dados de proveniência, torna-se viável avaliar e comparar diferentes versões do modelo, ajudando a identificar a melhor configuração para o problema de modelagem de aprendizado em questão. Isso também viabiliza o gerenciamento do desenvolvimento do *pipeline*, incluindo o rastreamento de mudanças e a avaliação de impacto entre versões de forma estruturada e eficiente.

Para isso, é considerado o módulo de consultas disponibilizado no protótipo (indicado na Seção 4.3.4), o qual acessa o repositório de dados onde são armazenados os dados de proveniência gerados durante as execuções do *pipeline*. Essas consultas são escritas em SPARQL e executadas no repositório de dados. No entanto, são disponibilizadas algumas consultas predefinidas, permitindo que usuários sem conhecimento prévio da linguagem realizem análises. Se o usuário tiver domínio sobre a linguagem SPARQL, o módulo também oferece a possibilidade de escrever consultas personalizadas, o que traz maior flexibilidade e aprofundamento nas análises.

Com a estruturação dos dados de proveniência, obtêm-se informações organizadas com base nos metadados definidos no capítulo anterior. Estes são exemplificados a seguir:

- Dados de proveniência básica do *pipeline*: incluem as informações referentes à criação e à atualização do *pipeline*, o responsável pelo desenvolvimento do *pipeline*, e o título do *pipeline* (obtido a partir da especificação do nome do arquivo);
- Dados de proveniência básica do conjunto de dados: incluem informações como datas de criação e modificação, título, publicador, versão, URL de acesso e agente responsável. Alguns deles estão representados no Quadro 4;

Quadro 4 – Exemplo de Dados de Proveniência Básica do Conjunto de Dados (*Pipeline Alzheimer's Disease*)

| | execution | created | modified | publisher | accessURL |
|---|---|---|---|---|---|
| 1 | _:5.1.1 | "2024-06-14T23:27:03.5636133" | "2024-06-11T20:24:30.613" | provx:DatasetPublisher/5.1.1 | "https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset/versions/1" |
| 2 | _:6.1.1 | "2024-06-14T23:27:03.5636133" | "2024-06-11T20:24:30.613" | provx:DatasetPublisher/6.1.1 | "https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset/versions/1" |

Fonte: O Autor

- Dados do Modelo de AM: Esses dados são compostos por detalhes sobre os modelos gerados, considerando o algoritmo utilizado, os parâmetros especificados, a especificação da partição dos conjuntos de treinamento, teste e validação, bem como os valores obtidos pelas métricas de avaliação. Conforme observado no Quadro 5, a consulta exemplifica o retorno de alguns desses dados de AM referentes às execuções do *pipeline Alzheimer's Disease*. Nessa consulta, os pares rdfs:label e provx:value, correspondentes a cada parâmetro do modelo e a cada métrica de avaliação, são agrupados no resultado com o objetivo de facilitar a visualização; e

Quadro 5 – Exemplo de Dados do Modelo de AM (*Pipeline Alzheimer's Disease*)

| | execution | algorithm | allParams | allScores |
|---|---|---|---|---|
| 1 | _:5.1.1 | "RandomForestClassifier" | "n_estimators=100, criterion=gini, max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=sqrt, max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None, monotonic_cst=None" | "accuracy=0.9302325581395349, recall=0.8366013071895425, precision=0.9624060150375939, f1=0.895104895104895" |
| 2 | _:6.1.1 | "RandomForestClassifier" | "n_estimators=100, criterion=gini, max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=sqrt, max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None, monotonic_cst=None" | "accuracy=0.9511627906976744, recall=0.8954248366013072, precision=0.9647887323943662, f1=0.9288135593220339" |

Fonte: O Autor

- Dados da execução: exemplificados no Quadro 6, incluem detalhes como a duração da execução, os momentos de início e término e o status de conclusão. Entre esses dados, também estão incluídos os dados de proveniência estendida, que estabelecem os pares que identificam o plano definido para a execução e os resultados gerados. Esses elementos correspondem, respectivamente, à proveniência prospectiva (instruções e planos) e à proveniência retrospectiva (resultados observados da execução).

Quadro 6 – Exemplo de Dados de Proveniência da Execução – Dados da Execução (*Pipeline Alzheimer's Disease*)

| | execution | duration | endedAtTime | startedAtTime | wasAssociatedWith |
|---|---|---|---|---|---|
| 1 | _:5.1.1 | "0:00:56.558931" | "2025-02-10 04:51:54.506302" | "2025-02-10 04:50:57.947371" | "provx:RunExecuter/5.1.1" |
| 2 | _:6.1.1 | "0:00:59.876984" | "2025-02-10 04:59:00.888532" | "2025-02-10 04:58:01.011548" | "provx:RunExecuter/6.1.1" |

Fonte: O Autor

Conforme observado no Quadro 7, temos uma amostra dos dados de proveniência retrospectiva, indicando algumas das entidades relacionadas à execução do *pipeline*. De acordo com o detalhado na Seção 5.1.1, essa proveniência retrospectiva é disponibilizada pela ferramenta noWorkflow e estruturada utilizando o modelo PROVX. Nesta abordagem, essa proveniência retrospectiva é associada à sua classe correspondente (provx:RetrospectiveProvenance), a qual compõe a proveniência estendida juntamente com a proveniência prospectiva. Isso possibilita a realização de consultas e o uso desses dados para ajustar futuras execuções (conforme indicado na Seção 4.3.1 e na Seção 4.3.2).

Quadro 7 – Amostra dos Dados Referentes à Proveniência Retrospectiva para uma Execução do *Pipeline Alzheimer's Disease.*

| | execu... | pipeline | retrospective |
|---|---|---|---|
| 1 | _:5.1.1 | "alzheimer_s_disease_prediction_on_patient" | "[ 'prov: entity(attribute24, [value="<function read_csv at 0x000002DEE662CE18>", type="script:attribute", label="pd.read_csv"]) ', 'prov: entity(10_\'alzheimers_disease_data.csv\', [value="\'alzheimers_disease_data.csv\'", type="script:literal"]) ', 'prov:entity(10_ \'PatientID\', [value="\'PatientID\'", type="script:literal"]) ', 'prov:entity(call20, [value=" Age Gender Ethnicity EducationLevel ... DifficultyCompletingTasks Forgetfulness Diagnosis DoctorInCharge ', 'PatientID ..." |

Fonte: O Autor.

O usuário pode acessar todas essas informações referentes às execuções específicas e relacionar os dados de diferentes execuções, sendo possível analisá-los para realizar ajustes no *pipeline* como, por exemplo, a otimização da escolha dos parâmetros do algoritmo de AM, o que pode auxiliar nas execuções futuras do *pipeline*. Para as consultas predefinidas, é necessário apenas que o usuário especifique o alvo de cada consulta (por exemplo: indicar qual *pipeline* que deseja verificar o melhor desempenho). Entre essas consultas, destacam-se:

- Quais foram os resultados obtidos para as métricas avaliadas em todas as execuções de um mesmo *pipeline*?
- Qual iteração apresentou o melhor resultado para a métrica de acurácia, considerando todas as execuções de um mesmo *pipeline*?

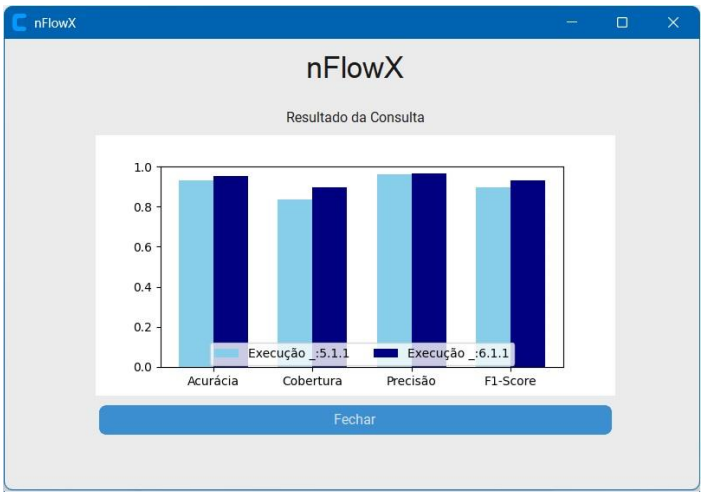
- Quais execuções de *pipelines* de AM foram realizadas utilizando um mesmo conjunto de dados específico?
- Quais foram os dados obtidos para uma execução específica?

A seguir, a primeira consulta é detalhada, mostrando seu funcionamento e os resultados obtidos nos cenários avaliados. O detalhamento das demais consultas utilizadas neste trabalho, incluindo exemplos de uso e execuções, estão disponíveis no Apêndice B, já as consultas em linguagem SPARQL estão disponíveis no Apêndice C.

- Consulta predefinida 01: Quais foram os resultados obtidos para as métricas avaliadas em todas as execuções de um mesmo *pipeline*?

  Para esta consulta, é necessário que o usuário indique dados de proveniência básica do *pipeline* alvo (o título do *pipeline*, responsável e a data de criação). Isso permite identificar quais execuções correspondem ao *pipeline* indicado, sendo retornado ao usuário os resultados obtidos a partir do repositório de dados. Na Figura 32, é apresentado o resultado obtido para a execução do *pipeline Alzheimer's Disease*, o qual retorna ao usuário o identificador da execução e os resultados obtidos para as métricas avaliadas. A partir desses resultados, é possível comparar os resultados das execuções de forma direta, onde o usuário também pode explorar os resultados de uma execução específica a partir da utilização da Consulta 04, a qual retorna o seu respectivo detalhamento.

Figura 32 – Resultado obtido para a Consulta 01




Fonte: O Autor.

### 5.4.2 Análises acerca dos modelos gerados no Cenário 01

Para avaliar se os modelos criados no Cenário 1 apresentam melhorias significativas em comparação aos modelos gerados no *baseline*, a partir das métricas avaliativas dos modelos, conforme detalhado na Seção 5.3.1, foi aplicado o teste de Wilcoxon (Tabela 8), considerando as

hipóteses definidas na seção 5.3. Caso o p-valor obtido no teste seja menor do que o nível de significância estabelecido (por padrão, $\alpha = 0,05$), pode-se afirmar que há uma diferença estatisticamente significativa entre os dois modelos.

Tabela 8 – Resultados da Avaliação de Modelos - Valores do Teste de Wilcoxon.

| | **Wilcoxon: p-valor** | | | |
|---|---|---|---|---|
| ***Pipeline*** | **Acurácia** | **Precisão** | **Cobertura** | ***F1-score*** |
| Pip:Breast Cancer | 0.0207 | 0.0284 | 0.0176 | 0.0109 |
| Pip:Nasa | 0.0076 | **0.5930** | 0.0107 | 0.0077 |
| Pip:Alzhiemier | 0.0199 | **0.6744** | 0.0076 | 0.0109 |
| Pip:Diabetes | 0.0020 | 0.0020 | **0.0020** | 0.0020 |
| Pip:Flight | 0.0020 | 0.0020 | 0.0020 | 0.0020 |

Fonte: O Autor.

Com a aplicação do p-valor dos testes de Wilcoxon para os diferentes *pipelines* de modelos, avaliando quatro métricas (Acurácia, Precisão, Cobertura e *F1-score*), os resultados demonstraram que, para o *pipeline* Pip:Breast Cancer e Pip:Flight, os p-valores são estatisticamente significativos em todas as métricas. No caso do Pip:Nasa, apesar dos resultados do p-valor para acurácia, *F1-score* e cobertura serem significativos, a precisão não indicou melhorias. Para o Pip:Alzheimer, já havia sido observado nos resultados da Seção 5.4 que a precisão seria dificilmente beneficiada pelas alterações que foram realizadas no *pipeline*. No caso do *pipeline* Pip:Diabetes, embora os modelos indiquem melhorias em geral, as diferenças observadas na comparação da métrica de cobertura entre o baseline e o Cenário 1 foram predominantemente negativas. Isso significa que, apesar da existência de uma diferença entre os resultados do baseline e do Cenário 1, o modelo do *baseline* apresentou melhores resultados. Portanto, essa métrica também não demonstrou um desempenho satisfatório para o Cenário 1. Esse comportamento está relacionado ao fato de que o processo aplicado, de forma geral, promoveu um equilíbrio entre precisão e cobertura, o que refletiu diretamente na melhora do *F1-score*, mas não necessariamente em ganhos expressivos em cada uma dessas métricas de forma individual.

Dessa forma, ao considerar as hipóteses formuladas na seção 5.3:

- $H_0$: Quando a avaliação de modelos de AM que passaram por ajustes bem-sucedidos com base nos dados de proveniência é comparada à avaliação de modelos sem esses ajustes, não se observa melhoria nos resultados do modelo; e
- $H_1$: Quando a avaliação de modelos de AM que passaram por ajustes bem-sucedidos com base nos dados de proveniência é comparada à avaliação de modelos sem esses ajustes, observa-se melhoria nos resultados do modelo.

Os resultados obtidos indicam que a hipótese H1 é confirmada para as métricas de acurácia e *F1-score*, uma vez que foi observada uma melhoria estatisticamente significativa nessas

métricas. No entanto, a hipótese H1 foi refutada para as métricas de precisão e cobertura, já que as alterações nos modelos não demonstraram ganhos significativos nessas métricas. É possível afirmar que a abordagem utilizada é eficaz no aprimoramento de modelos quando o foco está na melhoria da acurácia e do *F1-score*, atendendo, assim, às necessidades dos modelos voltados para a otimização dessas métricas específicas.

Com os resultados obtidos, também é relevante considerar os aspectos relacionados ao custo computacional envolvido na aplicação da abordagem, bem como seus benefícios práticos. Embora a captura e estruturação dos dados de proveniência, assim como a realização de consultas ao repositório de dados, introduzam um custo computacional adicional ao processo, esse custo pode ser compensado pelos benefícios decorrentes de sua aplicação. A abordagem proposta contribui para reduzir o retrabalho, otimizar o desenvolvimento do *pipeline* e promover a transparência e reexecução das atividades, o que é importante em contextos iterativos, nos quais a melhoria dos modelos de AM e a rastreabilidade das alterações assumem um papel importante na qualidade e confiabilidade dos resultados obtidos.

## 5.5 CONSIDERAÇÕES

A partir dos experimentos realizados, considerando o cenário especificado, foi possível observar que o uso de dados de proveniência para ajustar *pipelines* de AM resultou em uma melhoria na avaliação dos modelos ao considerar as métricas de acurácia e *F1-score*. Em resumo, os resultados dos experimentos indicam que a captura, documentação e acesso aos dados de proveniência, considerando a proveniência básica e estendida, viabilizam a reexecução de atividades originárias de execuções prévias, o ajuste de atividades do *pipeline* e a avaliação dos resultados, considerando as iterações, de forma que é possível refletir positivamente nos resultados dessas métricas.

# 6 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, foi apresentada uma abordagem que explora o uso de dados de proveniência capturados nas fases de preparação de dados e modelagem de AM em *pipelines*. O foco da abordagem é demonstrar como dados de proveniência podem ser utilizados para a reexecução de atividades entre execuções de *pipelines* e no ajuste de atividades associadas. Para isso, um caso de uso voltado à seleção de atributos foi realizado. A partir da implementação da abordagem, por meio de um protótipo denominado de nFlowX, é possível utilizar os dados de proveniência para otimizar as atividades nas execuções subsequentes de *pipelines*.

Para a realização dos experimentos, foram conduzidas duas execuções para cada *pipeline*: uma execução *baseline* e outra correspondente ao Cenário 01. Nessas execuções, foram empregados os mesmos *pipelines* e conjuntos de dados. No entanto, no Cenário 01, os dados de proveniência foram utilizados com o propósito de otimizar e beneficiar novas execuções.

Os resultados indicaram uma melhoria na avaliação da acurácia e do *F1-score* dos modelos de aprendizado de máquina analisados que realizaram os ajustes a partir dos dados de proveniência, quando comparados aos resultados obtidos sem o uso dos dados de proveniência. Dessa forma, é possível concluir que o presente trabalho atingiu seus objetivos, evidenciando o potencial do uso de dados de proveniência como uma estratégia para aprimorar o ajuste contínuo de modelos de AM. Nas seções 6.1, 6.2 e 6.3, respectivamente, são apresentadas as contribuições deste trabalho, seguidas das limitações e sugestões para trabalhos futuros.

## 6.1 CONTRIBUIÇÕES

As principais contribuições deste trabalho são listadas a seguir:

- **A definição de uma extensão de ontologia baseada na PROV-O (W3C):** a extensão PROVX considera especificamente o contexto da modelagem e o gerenciamento dos dados de proveniência em *pipelines* de AM. A PROVX inclui conceitos e relacionamentos associados à proveniência básica dos artefatos envolvidos em um *pipeline* de AM (conjunto de dados e atividades) e à proveniência estendida referente à execução do *pipeline*, o qual considera a proveniência prospectiva e retrospectiva, além de permitir relacionar uma execução com execuções prévias. Desta forma, é possível estabelecer uma estrutura bem definida que auxilia na compreensão e análise detalhada das atividades realizadas durante o desenvolvimento dos modelos de aprendizado;
- **Uma estratégia para uso dos dados de proveniência de forma que auxilie no desenvolvimento e aprimoramento de *pipelines* de AM:** a estratégia proposta permite melhorar os resultados obtidos pelos modelos de AM, mas também viabiliza a realização de análises utilizando os dados de proveniência, os quais podem fornecer informações

valiosas a serem utilizadas para alterar o *pipeline*, permitindo a realização de ajustes personalizados para os *pipelines* com base em execuções anteriores. A estratégia permite identificar execuções anteriores que utilizaram o *pipeline P* e o conjunto de dados *D* e, com base nos dados de proveniência estruturados, realizar intervenções automatizadas, como o uso da técnica RFE para refinar a seleção de atributos. No cenário *baseline*, o *pipeline* é executado sem modificações, enquanto no Cenário 1 os ajustes baseados em dados de proveniência são aplicados. Essa configuração resultou em melhorias nas métricas de avaliação, como acurácia e *F1-score*, demonstrando que a intervenção com a escolha otimizada de atributos trouxe um ganho significativo na avaliação dos modelos preditivos; e

- **O protótipo nFlowX, que implementa a abordagem proposta:** para avaliar a abordagem proposta, foi implementado o protótipo nFlowX, que automatiza todo o processo e inclui a captura e a estruturação dos dados de proveniência, a utilização desses dados obtidos em execuções anteriores para ajustes nos *pipelines*, o acesso e a análise das informações geradas durante as execuções dos *pipelines* de AM. Os experimentos realizados com o protótipo comprovaram a relevância dos resultados obtidos por meio da implementação da abordagem.

As seguintes publicações foram produzidas, até o momento, com base nos resultados alcançados durante o desenvolvimento desta tese:

- N. Targino, D. Souza, A. C. Salgado. An Approach to Associate Provenance From ML Pipelines With Transparency and Accountability. 2024. 15th International Conference on Information, Intelligence, Systems & Applications (IISA), Chania Crete, Greece, 2024, pp. 1-8, Disponível em: < https://doi.org/10.1109/IISA62523.2024.10786622>.
- N. Targino, D. Souza, A. C. Salgado. Using Provenance Data For Enhancing Model Training Pipelines: A Use Case with Feature Selection. 2025. Journal of Intelligent Information Systems. (Submetido).

## 6.2 LIMITAÇÕES

A seguir, estão descritas algumas limitações identificadas na abordagem proposta.

- **Ajuste de atividades utilizando dados de proveniência apenas para a seleção de atributos:** nesta versão, os dados de proveniência são utilizados para ajustar a atividade de seleção de atributos. No entanto, há potencial para expandir seu uso para considerar outras atividades no *pipeline*. Por exemplo, a seleção de parâmetros para o algoritmo de AM também pode ser aprimorada com base no histórico de execuções anteriores. O

uso desses dados de proveniência pode facilitar ajustes mais eficientes, contribuindo para um desempenho geral melhor do modelo;

- **Anotação para as fases do *pipeline*:** para a execução do *pipeline,* é necessário que as suas fases estejam explicitamente bem definidas, o que requer a inclusão de anotações que indiquem as fases principais gerenciadas na proposta. Atualmente, esse processo envolve alterações manuais no *pipeline* por parte do usuário. No entanto, seria vantajoso desenvolver uma abordagem que automatize a identificação e a marcação das fases, reduzindo a necessidade de intervenção direta e facilitando a integração e o gerenciamento do *pipeline*; e
- **Experimentos:** os experimentos realizados apresentaram resultados satisfatórios. Entretanto, foi observada a necessidade de realizar novos experimentos com usuários especialistas, considerando o desenvolvimento do *pipeline* desde suas etapas iniciais. Isso permitirá um acompanhamento mais detalhado e completo da evolução do *pipeline* ao longo do processo.

## 6.3 TRABALHOS FUTUROS

A seguir, são delineados alguns tópicos que podem ser investigados em pesquisas futuras relacionadas à abordagem desenvolvida:

- **Além da Seleção de Atributos:** A abordagem apresentada utiliza os dados de proveniência para atualizar a seleção de atributos, permitindo a seleção a partir do histórico das execuções. No entanto, é interessante expandir o uso da proveniência para outras atividades do *pipeline* de aprendizado, conforme indicado anteriormente em outros trabalhos, como para a otimização de hiperparâmetros e a detecção de falhas;
- **Exploração além da atividade de classificação**: A abordagem desenvolvida foi avaliada em problemas de classificação, como detecção de doenças e categorização de asteroides. No entanto, o uso de dados de proveniência estruturados semanticamente pode ser explorado em outras tarefas de AM, como problemas de regressão, o que demanda diferentes ajustes na estrutura dos *pipelines* e na forma de avaliação dos modelos;
- **Priorização de métricas avaliativas**: A abordagem proposta demonstrou melhorias nas métricas de acurácia e *F1-score* dos modelos de AM, ao utilizar dados de proveniência para ajustes nos *pipelines*. No entanto, métricas como precisão e cobertura não apresentaram avanços significativos separadamente, indicando oportunidades para aperfeiçoamentos mais direcionados. Neste sentido, podem ser investigadas técnicas de ajuste do *pipeline* voltadas à melhoria de métricas específicas, como a melhora direta da avaliação da precisão ou da cobertura, além da exploração de métodos que permitam

balancear de forma mais controlada as métricas avaliativas mais relevantes ao contexto de aplicação do *pipeline*; e

- **Fases do *pipeline*:** Neste trabalho, a abordagem proposta é voltada para as fases de preparação de dados e modelagem de AM, mas existem outras fases que compõem o ciclo de atividades que um *pipeline* pode contemplar, como a fase de entendimento dos dados e a fase de implantação, as quais envolvem a integração do modelo treinado em sistemas de produção para gerar previsões em tempo real. Ao considerar mais fases do *pipeline*, será possível fornecer uma visão mais abrangente do processo de desenvolvimento e manutenção de modelos de aprendizado de máquina.

# REFERÊNCIAS

BERNERS-LEE, T.; HENDLER, J.; LASSILIA, O. The Semantic Web. Scientific American, v. 284, n. 5, p. 34–44, 2001. Disponível em: <http://dx.doi.org/10.1038/scientificamerican0501-34>.

BIVAR, B.; SANTOS, L.; KOHWALTER, T.; MARINHO, A.; MATTOSO, M.; BRAGANHOLO, V. Uma Comparação entre os Modelos de Proveniência OPM e PROV. In: Proceedings of VII Brazilian e-Science Workshop (BreSci), p. 1787-1794, 2013. Disponível em: <https://sol.sbc.org.br/index.php/bresci/issue/view/796>.

BORST, P.; AKKERMANS, H.; TOP, J. Engineering ontologies. International Journal of Human-Computer Studies, v. 46, n. 2-3, p. 365-406, 1997. Disponível em: <https://doi.org/10.1006/ijhc.1996.0096>.

BUNEMAN, P.; KHANNA, S.; TAN, W. Data provenance- Some basic issues. Springer, p. 87–93, 2000. Disponível em: <https://link.springer.com/chapter/10.1007/3-540-44450-5_6>.

BURKART, N.; HUBER, M. F. A Survey on the Explainability of Supervised Machine Learning. J. Artif. Int. Res., v. 70, p. 245–317, May 2021. Disponível em: <https://doi.org/10.1613/jair.1.12228>.

BUTT, A. S.; FITCH, P. "ProvONE+: A Provenance Model for Scientific Workflows," in Web Information Systems Engineering – WISE 2020, vol. 12221, Lecture Notes in Computer Science, 2020, pp. 459-474, Disponível em: <https://doi.org/10.1007/978-3-030-62008-0_30>.

CELDRAN, A. H.; KREISCHER, J.; DEMIRCI, M.; LEUPP, J.; SÁNCHEZ SÁNCHEZ, P. M.; FIGUEREDO FRANCO, M.; BOVET, G.; MARTÍNEZ PÉREZ, G.; STILLER, B. A Framework Quantifying Trustworthiness of Supervised Machine and Deep Learning Models. In: The AAAI Workshop on Artificial Intelligence Safety 2023 (SafeAI 2023), Washington DC, USA, 13-14 Feb. 2023. CEUR-WS, pp. 1-14, 2023. Disponível em: <https://ceur-ws.org/Vol-3381/1.pdf>.

CHAPMAN, P.; CLINTON, J.; KERBER, R.; KHABAZA, T.; REINARTZ, T.; SHEARER, C.; WIRTH, R. CRISP-DM 1.0, CRISP-DM Consortium, 2000.

CHAPMAN, A.; MISSIER, P.; SIMONELLI, G.; TORLONE, R. "Capturing and querying fine-grained provenance of preprocessing pipelines in data science." Proceedings of the VLDB Endowment, v. 14, n. 4, p. 507-520, 2020. Disponível em: <https://doi.org/10.14778/3436905.3436911>.

CHERAGHI, Y.; KORD, S.; MASHAYEKHIZADEH, V. "Application of machine learning techniques for selecting the most suitable enhanced oil recovery method; challenges and opportunities." Journal of Petroleum Science and Engineering, v. 205, 2021, p. 108761. ISSN 0920-4105. Disponível em: <https://doi.org/10.1016/j.petrol.2021.108761>.

ELKHAROUA, R. *Alzheimer's Disease Dataset*. Kaggle, 2023. Disponível em: <https://doi.org/10.34740/kaggle/dsv/8668279>. Acesso em: 21 jan. 2025.

ELMRABIT, N.; ZHOU, F.; LI, F.; ZHOU, H. "Evaluation of Machine Learning Algorithms for Anomaly Detection." 2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), Dublin, Irlanda, 2020, p. 1-8. Disponível em: <https://doi.org/10.1109/CyberSecurity49315.2020.9138871>.

ESTEVES, D.; MOUSSALLEM, D.; NETO, C. B.; SORU, T.; USBECK, R.; ACKERMANN, M.; LEHMANN, J. "MEX vocabulary: a lightweight interchange format for machine learning experiments." Proc. 11th Int. Conf. Semantic Systems (SEMANTICS '15), p. 169-176, 16 set. 2015. Disponível em: <https://doi.org/10.1145/2814864.2814883>.

FAYYAD, U. M.; PIATETSKY-SHAPIRO, G.; SMYTH, P.; UTHURUSAMY, R. Advances in Knowledge Discovery and Data Mining. Menlo Park, CA, EUA: AAAI Press, 2003. ISBN: 9780262560979.

FREIRE, J., KOOP, D., SANTOS, E., and SILVA, C. T. (2008). Provenance for computational tasks: A survey. Computing in science & engineering, 10(3): p. 11–21. Disponível em: <https://doi.org /10.1109/MCSE.2008.79>.

GARIJO, D.; GIL, Y. "The P-PLAN Ontology." 12 mar. 2014. Disponível em: <http://vocab.linkeddata.es/p-plan>. Acesso em: 18 dez. 2024.

HUYNH, T.D., MOREAU, L. (2015). ProvStore: A Public Provenance Repository. In: Ludäscher, B., Plale, B. (eds) Provenance and Annotation of Data and Processes. IPAW 2014. Lecture Notes in Computer Science(), vol 8628. Springer, Cham. Disponível em: <https://doi.org/10.1007/978-3-319-16462-5_32>.

GOBLE, C. "Position statement: Musings on provenance, workflow and (semantic web) annotations for bioinformatics." Workshop on Data Derivation and Provenance, 2002.

GROTH, P.; MOREAU, L. "An Overview of the PROV Family of Documents." W3C, 2013. Disponível em: <https://www.w3.org/TR/prov-overview/>. Acesso em: 10 dez. 2024.

GRUBER, T. R. A Translation Approach to Portable Ontology Specifications. J. Knowledge Acquisition - Special issue: Current issues in knowledge modeling, v. 5, n. 2, p. 199–220, 1993. Disponível em: <https://doi.org/10.1006/knac.1993.1008>.

HAMDARD, M. S.; LODIN, H. Effect of Feature Selection on the Accuracy of Machine Learning Model. International Journal of Management, Research and Applications, 6(9), 66–70, 2023. Disponível em: <https://doi.org/10.47191/ijmra/v6-i9-66>.

HASTIE, T.; TIBSHIRANI, R.; WAINWRIGHT, M. Statistical Learning with Sparsity: The Lasso and Generalizations (1st ed.). Chapman and Hall/CRC. 2015. Disponível em: <https://doi.org/10.1201/b18401>.

HERSCHEL, M.; DIESTELKÄMPER, R.; BEN LAHMAR, H. "A survey on provenance: What for? What form? What from?" The VLDB Journal, v. 26, p. 881–906, 2017. Disponível em: <https://doi.org/10.1007/s00778-017-0486-1>.

JIANG, T.; GRADUS, J. L.; ROSELLINI, A. J. "Supervised Machine Learning: A Brief Primer." Behav. Ther., v. 51, n. 5, p. 675-687, set. 2020. Disponível em: <https://doi.org/10.1016/j.beth.2020.05.002>.

KERY, M. B.; HORVARTH, A.; MYERS, B. Variolite: Supporting Exploratory Programming by Data Scientists. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (Denver, Colorado, USA) (CHI ’17). Association for Computing Machinery, New York, NY, USA, p. 1265–1276. Disponível em: <https://doi.org/10.1145/3025453.3025626>.

KERZEL, D.; SAMUEL, S.; KÖNIG-RIES, B. Towards tracking provenance from machine learning notebooks. Proc. 13th Int. Conf. Knowledge Discovery and Information Retrieval, 2021, p. 1-9. Disponível em: <https://doi.org/10.5220/0010681400003064>.

KOHAVI, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. Proc. International Joint Conference on Artificial Intelligence (IJCAI), v. 2, p. 1137-1143, 1995. Disponível em: <https://www.researchgate.net/publication/262566867_A_Study_of_Cross-Validation_and_Bootstrap_for_Accuracy_Estimation_and_ Model_Selection>

KOTSIANTIS, S. Supervised Machine Learning: A Review of Classification Techniques. Informatica (Slovenia), v. 31, p. 249-268, 2007. Disponível em: <https://dl.acm.org/doi/10.5555/1566770.1566773>

KUHN, M.; JOHNSON, K. (2013). Applied predictive modeling. Springer. Disponível em: <https://doi.org/10.1007/978-1-4614-6849-3>.

LAUFER, C. Guia de Web Semântica. Centro de Estudos sobre Tecnologia Web – CeWeb.br, 2015. Disponível em: <http://ceweb.br/guias/web-semantica/>. Acesso em: 10 dez. 2017.

LEBO, T.; SAHOO, S.; MCGUINNESS, D. PROV-O: The PROV Ontology. W3C Candidate Recommendation, 30 de abril de 2013. Disponível em: < https://www.w3.org/TR/prov-o/>. Acessado em: 10 de janeiro de 2025.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. Nature 521, 436–444, 2015. Disponível em: <https://doi.org/10.1038/nature14539>.

LI, J.; CHENG, K.; WANG, S.; MORSTATTER, F.; TREVINO, R. P.; TANG, J.; LIU, H. Feature selection: A data perspective. ACM Comput. Surv., v. 50, n. 6, art. 94, p. 1-45, dez. 2017. Disponível em: <https://doi.org/10.1145/3136625>.

LI, Z.; KANT, H.; HAI, R.; KATSIFODIMOS, A.; BRAMBILLA, M.; BOZZON, A. Metadata representations for queryable repositories of machine learning models. IEEE Access, v. 11, p. 125616-125630, 2023. Disponível em: <https://doi.org/10.1109/ACCESS.2023.3330647>.

LÓSCIO, B. F.; BURLE, C.; CALEGARI, N. Data on the Web Best Practices. The World Wide Web Consortium (W3C), 2017. Disponível em: <https://www.w3.org/TR/2017/REC-dwbp-20170131/>. Acesso em: 10 dez. 2017.

LOURENÇO, R.; FREIRE, J.; SHASHA, D. BugDoc: A system for debugging computational pipelines. Proc. ACM SIGMOD International Conference of Data (SIGMOD '20), 2020, p. 2733–2736. Disponível em: <https://doi.org/10.1145/3318464.3384692>.

LUDERMIR, T. B. Inteligência Artificial e Aprendizado de Máquina: estado atual e tendências. Estudos Avançados, v. 35, n. 101, p. 85–94, 2021. Disponível em: <https://doi.org/10.1590/s0103-4014.2021.35101.007>.

MARSLAND, S. Machine Learning: An Algorithmic Perspective. 2nd ed. Boca Raton, FL, USA: CRC Press, 2015. ISBN: 1466583282.

MIAO, J.; NIU, L. A Survey on Feature Selection. Procedia Computer Science, v. 91, p. 919-926, 2016. Disponível em: <https://doi.org/10.1016/j.procs.2016.07.111>.

MOREAU, L.; CLIFFORD, B.; FREIRE, J.; FUTRELLE, J.; GIL, Y.; GROTH, P.; KWASNIKOWSKA, N.; MILES, S.; MISSIER, P.; MYERS, J. et al. The Open Provenance Model Core Specification (v1.1). Future Generation Computer Systems, v. 27, n. 6, p. 743–756, 2011. Disponível em: <https://doi.org/10.1016/j.future.2010.07.005>.

NAMAKI, M. H.; FLORATOU, A.; PSALLIDAS, F.; KRISHNAN, S.; AGRAWAL, A.; WU, Y.; ZHU, Y.; WEIMER, M. Vamsa: Automated Provenance Tracking in Data Science Scripts. Proc. 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 2020, p. 1542–1551. Disponível em: <https://doi.org/10.1145/3394486.3403205>.

OLIVEIRA, R. L.; DUARTE, J. C.; CORDEIRO, K. F. Machine Learning Model Explainability Supported by Data Explainability: A Provenance-Based Approach. Journal of Information and Data Management, v. 15, n. 1, p. 93–102, 2024. Disponível em: <https://doi.org/10.5753/jidm.2024.3337>.

OMITAOMU, O. A.; NIU, H. Artificial Intelligence Techniques in Smart Grid: A Survey. Smart Cities, v. 4, p. 548–568, 2021. Disponível em: <https://doi.org/10.3390/smartcities4020029>.

PATIL, D. ., RANE, N. L., DESAI, P. ., & RANE, J. Machine learning and deep learning: Methods, techniques, applications, challenges, and future research opportunities. In D. . Patil, N. L. Rane, P. . Desai, & J. . Rane (Eds.), Trustworthy Artificial Intelligence in Industry and Society (pp. 28-81), 2024. Deep Science Publishing. Disponível em: <https://doi.org/10.70593/978-81-981367-4-9_2>.

PIMENTEL, J. F.; FREIRE, J.; BRAGANHOLO, V.; MURTA, L. Tracking and Analyzing the Evolution of Provenance from Scripts. Proc. 6th Int. Workshop Provenance and Annotation of Data and Processes (IPAW), v. 9672, p. 16-28, jun. 2016. Disponível em: <https://doi.org/10.1007/978-3-319-40593-3_2>.

PIMENTEL, J. F.; MURTA, L.; BRAGANHOLO, V.; FREIRE, J. noWorkflow: A Tool for Collecting, Analyzing, and Managing Provenance from Python Scripts. Proceedings of the VLDB Endowment, v. 10, n. 12, p. 1841 – 1844, 2017. Disponível em: <https://doi.org/10.14778/3137765.3137789>.

PRIYATNO, Arif & WIDIYANINGTYAS, Triyanna. A Systematic Literature Review: Recursive Feature Elimination Algorithms. JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer). 9. p. 196-207. 2024. Disponível em: <https://doi.org/10.33480/jitk.v9i2.5015>.

PURBASARI, A.; RINAWAN, F.; ZULIANTO, A.; KOMARA, H. CRISP-DM for Data Quality Improvement to Support Machine Learning of Stunting Prediction in Infants and Toddlers. Proc. 2021 8th Int. Conf. Advanced Informatics: Concepts, Theory and Applications (ICAICTA), pp. 1-6, 2021. Disponível em: <https://doi.org/10.1109/ICAICTA53211.2021.9640294>.

WORLD WIDE WEB CONSORTIUM (W3C). (2012) Web Ontology Language (OWL). . Disponível em: <https://www.w3.org/OWL/> Último Acesso: 14 de jan. de 2025.

WORLD WIDE WEB CONSORTIUM (W3C). SPARQL 1.1 Federated Query. 2013. Disponível em: <https://www.w3.org/TR/2013/REC-sparql11-federated-query-20130321/>. Acesso em: 22 nov. 2024.

WORLD WIDE WEB CONSORTIUM (W3C). (2014) Resource Description Framework (RDF). Disponível em: <https://www.w3.org/2001/sw/wiki/RDF> Último Acesso: 27 de jan. de 2025.

SCHELTER, S.; GRAFBERGER, S.; GUHA, S.; KARLAS, B.; ZHANG, C. Proactively Screening Machine Learning Pipelines with ARGUSEYES. Proc. SIGMOD '23, International Conference on Management of Data, New York, NY, USA, 2023, p. 91-94. Disponível em: <https://doi.org/10.1145/3555041.3589682>.

SCHNEIDER, L.; BISCHL, B.; THOMAS, J. Multi-Objective Optimization of Performance and Interpretability of Tabular Supervised Machine Learning Models. Proc. Genetic and Evolutionary Computation Conference, New York, NY, USA, 2023, p. 538–547. Disponível em: <https://doi.org/10.1145/3583131.3590380>.

SCHLEGEL, M.; SATTLER, K. Extracting Provenance of Machine Learning Experiment Pipeline Artifacts. Advances in Databases and Information Systems: 27th ADBIS 2023, Barcelona, Spain, 2023, p. 238–251. Disponível em: <https://doi.org/10.1007/978-3-031-42914-9_17>.

SAMUEL, S.; LÖFFLER, F.; KÖNIG-RIES, B. Machine Learning Pipelines: Provenance, Reproducibility and FAIR Data Principles. Lecture Notes in Computer Science, p. 226-230. 2021. Disponível em: <https://doi.org/10.1007/978-3-030-80960-7_17>.

SICSÚ, A. L.; SAMARTINI, A.; BARTH, N. L. Técnicas de Machine Learning. São Paulo: Blucher, 2023. 394 p. ISBN 978-65-5506-396-7.

SCIKIT-LEARN DEVELOPERS. Feature selection. Scikit-learn Documentation, 2023. Disponível em: <https://scikit-learn.org/stable/modules/feature_selection.html#rfe>. Acesso em: 07 jan. 2025.

SINGH, V.; SINGH, A.; JOSHI, K. Fair CRISP-DM: Embedding Fairness in Machine Learning (ML) Development Life Cycle. Proc. Hawaii Int. Conf. System Sciences, p. 1531-1540, 2022. Disponível em: <https://doi.org/10.24251/HICSS.2022.190>.

SOUZA, R., AZEVEDO, L. G., LOURENÇO, V., SOARES, E., THIAGO, R., BRANDÃO, R., CIVITARESE, D., VITAL BRAZIL, E., MORENO, M., VALDURIEZ, P., MATTOSO, M., CERQUEIRA, R., NETTO, M. A. S. (2021). Workflow Provenance in the Lifecycle of Scientific Machine Learning. Concurrency Computation: Practice and Experience, pp. 1–21, 2021. Disponível em: <https://doi.org/10.1002/cpe.6544>.

SVETNIK, V.; LIAW, A.; TONG, C.; CULBERSON, C.; SHERIDAN, R.; FEUSTON, B. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. Journal of Chemical Information and Computer Science, v. 43, p. 1947-1958, 2003. Disponível em: <https://doi.org/10.1021/ci034160g**>**.

TEUBL, M. S.; MEZHUYEV, V.; TSCHANDL, M. Development of an ML Model for the Classification of Surface Quality in a Milling Process. Proc. 2023 9th Int. Conf. Computer Technology Applications, New York, NY, USA, 2023, p. 214–219. Disponível em: <https://doi.org/10.1145/3605423.3605449>.

WILCOXON, F. Individual Comparisons by Ranking Methods. Biometrics Bulletin,1, p. 80–83, 1945. Disponível em: <https://doi.org/10.2307/3001968>.

ZAHARIA, M., CHEN, A., DAVIDSON, A., GHODSI, A., HONG, S. A., KONWINSKI, A., MURCHING, S., NYKODYM, T., OGILVIE, P., PARKHE, M., XIE, F., & ZUMAR, C. Accelerating the Machine Learning Lifecycle with MLflow. IEEE Data Engineering Bulletin, 41(4):7, Dezembro, 2018. Disponível em: <https://api.semanticscholar.org/CorpusID:83459546>.

ZHAO, Y., Wilde, M., & Foster, I. Applying the virtual data provenance model. In Proceedings of the 2006 International Conference on Provenance and Annotation of Data, p. 148–161, 2006. Springer. Disponível em: <https://doi.org/10.1007/11890850_16>.

# APÊNDICE A – PROVX: Detalhamento das classes

## Sobre a Ontologia

A ontologia PROVX foi desenvolvida para representar os dados de proveniência em execuções de *pipelines* de aprendizado de máquina. A PROVX fornece um modelo extensível e interoperável baseado em padrões, incluindo PROV-O e P-Plan, contemplando tanto a proveniência prospectiva quanto a proveniência retrospectiva de execuções de *pipelines* de AM.

## Classes Principais

**provx:BasicProvenance**

| | |
|---|---|
| **Definição** | Entidade que representa a proveniência básica de um recurso, podendo se referir, por exemplo, a conjuntos de dados (provx:DatasetBasicProvenance) ou a *pipelines* (provx:PipelineBasicProvenance). |
| **Subclasse de** | prov:Entity |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável.<br>prov:wasGeneratedBy: Atividade responsável por sua geração.<br>prov:wasDerivedFrom: Entidade da qual os dados são derivados. |
| **Propriedades de Dados** | dcterms:created: Data de criação.<br>dcterms:modified: Data da última modificação.<br>owl:versionInfo: Versão do recurso.<br>dcat:accessURL: URL para acessar os dados. |

**provx:Dataset**

| | |
|---|---|
| **Definição** | Coleção organizada de dados utilizados em uma execução (provx:Run). |
| **Subclasse de** | prov:Entity |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável.<br>provx:mustPresent: Necessidade de apresentar os metadados do conjunto de dados relacionados. |
| **Propriedades de Dados** | dcterms:title: Título do conjunto de dados. |

**provx:DatasetMetadata**

| | |
|---|---|
| **Definição** | Coleção de metadados que descreve o conjunto de dados (provx:Dataset), contendo informações como sua origem e estrutura. |
| **Subclasse de** | prov:Entity |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável. |
| **Propriedades de Dados** | dcterms:title: Título do conjunto de metadados.<br>dcterms:created: Data de criação.<br>dcterms:modified: Data da última modificação. |

**provx:DatasetCreator/UpdaterAgent**

| | |
|---|---|
| **Definição** | Agente responsável por criar ou atualizar o conjunto de dados (provx:Dataset). |
| **Subclasse de** | prov:Agent |
| **Propriedades da Classe** | - |
| **Propriedades de Dados** | foaf:name: Nome do agente.<br>foaf:contact: Contato do agente (opcional). |

**provx:DatasetPublisherAgent**

| | |
|---|---|
| **Definição** | Agente responsável por publicar o conjunto de dados (provx:Dataset). |
| **Subclasse de** | prov:Agent |
| **Propriedades da Classe** | prov:actedOnBehalfOf: Agente em nome do qual atua, no caso do DatasetCreator/UpdaterAgent. |
| **Propriedades de Dados** | foaf:name: Nome do agente.<br>foaf:contact: Contato do agente (opcional). |

**provx:ExtendedProvenance**

| | |
|---|---|
| **Definição** | Permite detalhar o planejamento e execução do *pipeline*, sendo uma entidade composta por aspectos de proveniência prospectiva (provx:ProspectiveProvenance) e proveniência retrospectiva (provx:RetrospectiveProvenance). |
| **Subclasse de** | prov:Entity |
| **Propriedades da Classe** | prov:composedBy: Componentes da proveniência estendida, formado por . provx:ProspectiveProvenance e provx:RetrospectiveProvenance. |
| **Propriedades de Dados** | - |

**provx:MLModel**

| | |
|---|---|
| **Definição** | Descreve o modelo de aprendizado de máquina gerado durante a execução de um *pipeline* (provx:Pipeline). |
| **Subclasse de** | Prov:Entity |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável.<br>prov:wasGeneratedBy: Atividade responsável por sua geração. |
| **Propriedades de Dados** | provx:Model: Cada modelo de AM gerado durante a execução do *pipeline*, sendo composto por:<br>mlonto:Algorithm: Algoritmo de AM utilizado;<br>provx:Parameters: Parâmetros do algoritmo; e<br>provx:Scores: Resultados das métricas de avaliação.<br>provx:setPercent: Especificação das partições de treinamento, teste e validação. |

**provx:*Pipeline***

| | |
|---|---|
| **Definição** | Sequência organizada de instruções que especifica as etapas e atividades de um processo de AM, executada pela atividade provx:Run. |
| **Subclasse de** | prov:Entity |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável pela criação/atualização. |
| **Propriedades de Dados** | dcterms:title: Nome do *pipeline*. |

**provx:*Pipeline*Creator/UpdaterAgent**

| | |
|---|---|
| **Definição** | Agente responsável pela criação/atualização do *pipeline* (provx:Pipeline). |
| **Subclasse de** | prov:Agent |
| **Propriedades da Classe** | - |
| **Propriedades de Dados** | foaf:name: Nome do agente.<br>foaf:contact: Contato do agente (opcional). |

**provx:ProspectiveProvenance**

| | |
|---|---|
| **Definição** | Representa a descrição da estrutura do *pipeline (provx:Pipeline)*, com suas respectivas etapas, formando um p-plan:Plan. Cada etapa que compõe o *pipeline* corresponde a um p-plan:Step. |
| **Subclasse de** | p-plan:Plan |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável.<br>prov:wasGeneratedBy: Atividade responsável por sua geração.<br>prov:wasDerivedFrom: *Pipeline* correspondente. |
| **Propriedades de Dados** | provx:identifier: Identificador único.<br>p-plan:hasStep: Etapas que compõe o *pipeline*. |

**provx:RetrospectiveProvenance**

| | |
|---|---|
| **Definição** | Representa a proveniência retrospectiva obtida a partir de provx:Run, descrevendo a execução das etapas previstas em provx:ProspectiveProvenance. |
| **Subclasse de** | prov:Entity |
| **Propriedades da Classe** | prov:wasAttributedTo: Agente responsável.<br>prov:wasGeneratedBy: Atividade responsável por sua geração.<br>provx:isResultOfExecutionOf: Indica a provx:ProspectiveProvenance que foi executada. |
| **Propriedades de Dados** | provx:identifier: Identificador único.<br>provx:outputs: Detalhamento da execução de provx:ProspectiveProvenance. |

**provx:Run**

| | |
|---|---|
| **Definição** | Atividade realizada utilizando o *pipeline* (provx:Pipeline), o conjunto de dados (provx:Dataset) e seus metadados (provx:DatasetMetadata). Compreende as subatividades de execução do *pipeline* (provx:PipelineExecution) e processamento de proveniência (provx:ProvenanceProcessing), gerando informações gerais da execução, dados de proveniência estendida (provx:ExtendedProvenance) e informações sobre o modelo de aprendizado de máquina (provx:MLModel). |
| **Subclasse de** | prov:Activity |
| **Propriedades da Classe** | prov:wasAssociatedWith: Agente responsável pela execução.<br>prov:used: Artefato(s) utilizados na execução.<br>provx:hasPrevious: Execuções anteriores relacionadas. |
| **Propriedades de Dados** | provx:identifier: Identificador único da execução.<br>provx:duration: Duração da execução.<br>provx:status: Status da execução (ex.: Finalizado).<br>prov:startedAtTime: Início da execução.<br>prov:endedAtTime: Fim da execução.<br>provx:hasPreviousRun: Execuções anteriores relacionadas.<br>prov:used: Artefatos utilizados na execução<br>provx:ExtendedProvenance: Proveniência estendida da execução. |

**provx:RunExecutorAgent**

| | |
|---|---|
| **Definição** | Agente responsável pela execução da atividade provx:Run. |
| **Subclasse de** | prov:Agent |
| **Propriedades da Classe** | - |
| **Propriedades de Dados** | foaf:name: Nome do agente.<br>foaf:contact: Contato do agente (opcional). |

# APÊNDICE B – NFLOWX: Consultas Predefinidas

Detalhamento das demais consultas predefinidas utilizadas no trabalho (definidas na Seção 5.4.1), detalhando sua estrutura, propósito e aplicação. Além disso, são fornecidos exemplos de uso e os resultados obtidos a partir das execuções, permitindo uma melhor compreensão do comportamento e da eficácia de cada consulta no contexto analisado.

- Consulta predefinida 02: Qual iteração apresentou o melhor resultado para a métrica de acurácia, considerando todas as execuções de um mesmo *pipeline*?

  Para realizar essa consulta é necessário que o usuário informe o título do *pipeline*, responsável e a sua data de criação. Isso permite acessar as execuções relacionadas com o *pipeline* especificado para então identificar qual execução apresentou o valor mais alto para a métrica de acurácia. Como resultado, é exibido para o usuário todos os detalhes da execução e informações úteis para o desenvolvimento do *pipeline*, conforme observado na Figura 1, a qual mostra o retorno dos dados da melhor execução do *pipeline* de título "diabetes_dataset_classification_model_eda".

Figura 1 – Resultado obtido para a Consulta 02

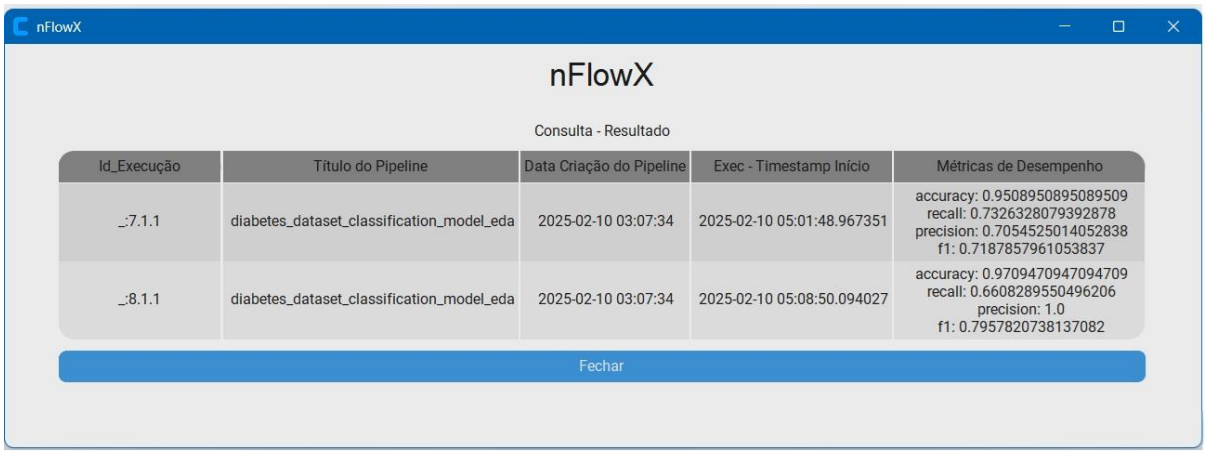

Fonte: O Autor

- Consulta predefinida 03 - Quais execuções de *pipelines* de AM foram realizadas utilizando um mesmo conjunto de dados específico?

  Nessa execução, é preciso que o usuário forneça dados de proveniência básica do conjunto de dados, sendo o título do conjunto de dados, o responsável e a sua data de criação, os quais permitem identificar todas as execuções de *pipelines* que utilizam desse mesmo conjunto de dados especificado. Nessa consulta foi realizada

a verificação de execuções utilizando o conjunto de dados “Comprehensive Diabetes Clinical Dataset(100k rows)", resultando no retorno de informações referentes a duas execuções de *pipelines* que utilizaram esse mesmo conjunto de dados, conforme ilustrado na Figura 2.

Figura 2– Resultado obtido para a Consulta 03




Fonte: O Autor

- Consultas predefinida 04 - Quais foram os dados obtidos para uma execução específica?

  Para essa consulta é necessário o id da execução desejada, o qual pode ser obtido a partir das consultas exemplificadas nas seções anteriores. Como resultado, é possível acessar todos os detalhes da execução, onde são apresentados os dados de proveniência básica do *pipeline* e do conjunto de dados, dados do modelo de AM gerado e os dados referentes à execução do *pipeline*. Essas informações são retornadas da mesma forma em que as informações da Consulta 2, conforme apresentadas na Figura 1.

# APÊNDICE C – Consultas SPARQL

Este apêndice apresenta um conjunto de consultas SPARQL desenvolvidas e utilizadas ao longo deste trabalho para explorar os dados de proveniência obtidos a partir da abordagem proposta. As consultas possibilitam a recuperação de informações específicas relacionadas às execuções dos *pipelines* de AM, como as métricas que avaliaram os modelos, os parâmetros utilizados, os detalhes sobre os modelos gerados e os agentes envolvidos no processo.

**Consulta I: (Seção 4.3.1) Identifica se existe execução anterior e retorna a melhor avaliada (utiliza dados do *pipeline P* e do conjunto de dados *D* da execução atual).**

**Observação:** Nesta consulta, são utilizadas variáveis definidas em Python, que são preenchidas automaticamente com os dados do *pipeline* e do conjunto de dados da execução atual. Essas variáveis são utilizadas para identificar e recuperar execuções relacionadas no repositório de dados do protótipo.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>


SELECT DISTINCT ?exec
    WHERE {
        GRAPH ?exec {
            ?pipeline a provx:PipelineBasicProvenance ;
                    dcterms:created \"""+pip_creation+"""\" ;
                            prov:wasAttributedTo ?pip_resp .

            ?pip_resp foaf:Person ?person;
            foaf:name \"""+pip_resp+"""\" .

            ?pp a provx:Pipeline ;
                dcterms:title  \"""+pip_title+"""\" .


            ?dt a provx:Dataset ;
                    dcterms:title \"""+data_title+"""\" .

            ?dataset a provx:DatasetBasicProvenance ;
                    dcterms:created \"""+data_creation+"""\" ;
                            prov:wasAttributedTo ?data_resp .
```

```
            ?data_resp prov:actedOnBehalfOf ?other.
            ?outro foaf:Person ?person2;
            foaf:name \"""+resp_creation+"""\" .

            ?mlModel a provx:MLModel ;
                        provx:Model ?modelDetails .

            ?modelDetails provx:Scores ?modelScoreDetails .

            ?modelScoreDetails rdfs:label\"""+metric+"""\" . ;
                           provx:value ?modelScoreValue.

        }
    }ORDER BY DESC (?modelScoreValue) LIMIT 1
```

**Consulta II: (Seção 5.4.1) Consulta aos dados de proveniência básica do conjunto de dados (Quadro 4):** Dados de Proveniência Básica do conjunto de dados das execuções do *pipeline Alzheimer's Disease.*

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX pplan: <http://purl.org/net/p-plan#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX                                                    provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>

SELECT ?execution ?created ?modified ?publisher ?accessURL
    WHERE {
        GRAPH ?execution {
            ?pipeline a provx:PipelineBasicProvenance ;
                  dcterms:created "2025-02-10 03:07:34" .

            ?pip a provx:Pipeline ;
                dcterms:title
"alzheimer_s_disease_prediction_on_patient"  .

            ?dataset a provx:DatasetBasicProvenance ;
                dcterms:created ?created ;
                dcterms:modified ?modified;
                dcterms:publisher ?publisher;
                dcat:accessURL ?accessURL.

        }
    }ORDER BY (?execution)
```

**Consulta III: (Seção 5.4.1) Dados do modelo de AM (Quadro 5):** Dados do Modelo de AM para as execuções do *pipeline Alzheimer's Disease.*

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX pplan: <http://purl.org/net/p-plan#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX                                                    mlonto:
<http://www.semanticweb.org/user/ontologies/2020/0/ml-
ontology#mlo:>
PREFIX                                                     provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>

SELECT ?execution ?algorithm
       (GROUP_CONCAT(DISTINCT      CONCAT(?paramlabel,      "=",
?paramvalue); SEPARATOR=", ") AS ?allParams)
       (GROUP_CONCAT(DISTINCT      CONCAT(?scorelabel,      "=",
?scorevalue); SEPARATOR=", ") AS ?allScores)
WHERE {
  GRAPH ?execution {
     ?pipeline a provx:PipelineBasicProvenance ;
               dcterms:created "2025-02-10 03:07:34" .

    ?pip a provx:Pipeline ;
              dcterms:title
"alzheimer_s_disease_prediction_on_patient"  .

    ?mlModel a provx:MLModel ;
             provx:Model ?model .

    ?model provx:Parameters ?param ;
           mlonto:Algorithm ?algorithm ;
           provx:Scores ?scores .

    ?scores rdfs:label ?scorelabel ;
            provx:value ?scorevalue .

    ?param rdfs:label ?paramlabel ;
           provx:value ?paramvalue .
  }
}
GROUP BY ?execution ?algorithm
ORDER BY ?execution
```

**Consulta IV: (Seção 5.4.1) Dados da execução (Quadro 6):** Dados referentes às execuções do *pipeline Alzheimer's Disease.*

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX pplan: <http://purl.org/net/p-plan#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX                                                    provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>


SELECT    ?execution    ?duracao    ?endedAtTime    ?startedAtTime
?wasAssociatedWith
    WHERE {
        GRAPH ?execution {
             ?pipeline a provx:PipelineBasicProvenance ;
               dcterms:created "2025-02-10 03:07:34" .

            ?pip a provx:Pipeline ;
                dcterms:title
"alzheimer_s_disease_prediction_on_patient"  .

            ?run a provx:Run ;
                        provx:Duration ?duracao ;
                         prov:endedAtTime ?endedAtTime ;
                         prov:startedAtTime ?startedAtTime;
                        prov:wasAssociatedWith
?wasAssociatedWith.

        }
    }ORDER BY (?execution)
```

**Consulta V: (Seção 5.4.1) Dados da execução (Quadro 7):** Dados Referentes à Proveniência Retrospectiva para uma Execução do *Pipeline Alzheimer's Disease.*

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX pplan: <http://purl.org/net/p-plan#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX                                                    provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>

SELECT ?exec ?pip_title ?retrospective
```

```
WHERE {
  GRAPH ?exec {
    ?pipeline a provx:PipelineBasicProvenance ;
              dcterms:created "2025-02-10 03:07:34" .

    ?pip a provx:Pipeline ;
         dcterms:title ?pip_title .

    FILTER(?pip_title = "alzheimer_s_disease_prediction_on_patient")

    ?run a provx:Run ;
         provx:ExtendedProvenance ?ex .

    ?ex provx:RetrospectiveProvenance ?retrospective .
  }
}ORDER BY (?exec) LIMIT 1
```

**Consulta VI: (Seção 5.4.1) Consulta predefinida 01:** Quais os resultados obtidos para as métricas avaliadas em todas as execuções de um mesmo *pipeline*?

**Observação:** Nesta consulta, são utilizadas variáveis definidas em Python, que são preenchidas com base nas entradas fornecidas pelo usuário. Essas variáveis são utilizadas para compor e executar uma consulta SPARQL predefinida sobre o repositório do protótipo.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX  provx:  <http://www.semanticweb.org/natac/ontologies/2024
/5/untitled-ontology-50>

SELECT DISTINCT ?exec ?metric ?modelMetricValue
    WHERE {
        GRAPH ?exec {
            ?pipeline a provx:PipelineBasicProvenance ;
                    dcterms:created \"""+pip_creation+"""\"  ;
                            prov:wasAttributedTo ?pip_resp .

            ?pip_resp foaf:Person ?person;
            foaf:name \"""+pip_resp+"""\"  .

            ?pp a provx:Pipeline ;
                 dcterms:title  \"""+pip_title+"""\"  .

            ?mlModel a provx:MLModel ;
                        provx:Model ?modelDetails .

            ?modelDetails  provx:Scores ?modelMetricDetails .

            ?modelMetricDetails rdfs:label ?metric ;
```

```
                                    provx:value ?modelMetricValue.
        }
    }ORDER BY ASC(?exec)
```

**Consulta VII: (Seção 5.4.1) Consulta predefinida 02:** Qual iteração apresentou o melhor resultado para a métrica de acurácia considerando todas as execuções de um mesmo *pipeline*?

**Observação:** Nesta consulta, são utilizadas variáveis definidas em Python, que são preenchidas com base nas entradas fornecidas pelo usuário. Essas variáveis são utilizadas para compor e executar uma consulta SPARQL predefinida sobre o repositório do protótipo.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX                                                    mlonto:
<http://www.semanticweb.org/user/ontologies/2020/0/ml-
ontology#mlo:>
PREFIX                                                     provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>

SELECT DISTINCT ?exec ?pipelineTitle ?datasetTitle ?algorithm
?startTime ?duration ?prospective ?retrospective
          (GROUP_CONCAT(DISTINCT  CONCAT(?param_label,  \":  \",
?param_value); SEPARATOR=\", \") AS ?ALLparameters)
          (GROUP_CONCAT(DISTINCT  CONCAT(?metric_label,  \":  \",
?metric_value); SEPARATOR=\", \") AS ?ALLmetrics)
        WHERE {
          GRAPH ?exec  {
                    ?pipeline a provx:PipelineBasicProvenance ;
                            dcterms:created
\"""+pip_creation+"""\"  ;
                                    prov:wasAttributedTo
?pip_resp .

                   ?pip_resp foaf:Person ?person;
                        foaf:name \"""+pip_resp+"""\" .

                   ?pp a provx:Pipeline ;
                        dcterms:title  ?pipelineTitle  .
               FILTER(?pipelineTitle = \"""+tpip+"""\"  )


            ?dataset a provx:Dataset ;
                     dcterms:title ?datasetTitle .

            ?run a provx:Run ;
                 prov:startedAtTime ?startTime ;
                 provx:Duration ?duration ;
```

```
                provx:ExtendedProvenance ?extendedProv .

            ?extendedProv           provx:RetrospectiveProvenance
?retrospective .
            ?extendedProv             provx:ProspectiveProvenance
?prospective .

            ?mlModel a provx:MLModel ;
                        provx:Model ?model .

            ?model mlonto:Algorithm ?algorithm .

            ?model provx:Parameters ?paramNode .
            ?paramNode rdfs:label ?param_label ;
                        provx:value ?param_value .

            ?model provx:Scores ?metricNode .
            ?metricNode rdfs:label ?metric_label ;
                        provx:value ?metric_value .

            ?model provx:Scores ?AccNode .
                ?AccNode rdfs:label \"accuracy\" ;
                       provx:value ?AccuracyValue .
        }
        }GROUP BY ?exec ?pipelineTitle ?datasetTitle ?algorithm
?startTime   ?duration   ?model   ?retrospective   ?prospective
?AccuracyValue ORDER BY DESC(?AccuracyValue) LIMIT 1
```

**Consulta VIII: (Seção 5.4.1) Consulta prédefinida 03:** Quais execuções de *pipelines* de AM são realizadas utilizando um mesmo conjunto de dados específico?

**Observação:** Nesta consulta, são utilizadas variáveis definidas em Python, que são preenchidas com base nas entradas fornecidas pelo usuário. Essas variáveis são utilizadas para compor e executar uma consulta SPARQL predefinida sobre o repositório do protótipo.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX                                                    provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>

SELECT DISTINCT ?exec ?pip_tit ?dt_created ?dini
                    (GROUP_CONCAT(DISTINCT  CONCAT(?metric,  ":  ",
?modelMetricValue); SEPARATOR=", ") AS ?metrics)
            WHERE {
                GRAPH ?exec {
                    ?dataset a provx:Dataset ;
```

```
                              dcterms:title \"""+data_title+"""\"
.
                    ?dt a provx:DatasetBasicProvenance ;
                         dcterms:created
\"""+data_creation+"""\" ;
                          prov:wasAttributedTo ?data_resp .
                        ?data_resp prov:actedOnBehalfOf ?other.

                    ?other foaf:Person ?personn;
                           foaf:name \"""+resp_creation+"""\" .

                    ?pipeline a provx:PipelineBasicProvenance ;
                              dcterms:created ?dt_created .

                    ?pp a provx:Pipeline ;
                        dcterms:title ?pip_tit .

                    ?mlModel a provx:MLModel ;
                                provx:Model ?modelDetails .

                    ?modelDetails                    provx:Scores
?modelMetricDetails .

                    ?modelMetricDetails rdfs:label ?metric ;
                                        provx:value
?modelMetricValue.

                    ?run a provx:Run ;
                         prov:startedAtTime ?dini.
                }
            }GROUP BY ?exec ?pip_tit ?dt_created ?dini ORDER BY
(?exec)
```

**Consulta IX: (Seção 5.4.1) Consultas predefinida 04:** Quais os dados obtidos para uma execução específica?

**Observação:** Nesta consulta, são utilizadas variáveis definidas em Python, que são preenchidas com base nas entradas fornecidas pelo usuário. Essas variáveis são utilizadas para compor e executar uma consulta SPARQL predefinida sobre o repositório do protótipo.

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX mlonto:
<http://www.semanticweb.org/user/ontologies/2020/0/ml-
ontology#mlo:>
PREFIX provx:
<http://www.semanticweb.org/natac/ontologies/2024/5/untitled-
ontology-50>
```

```
SELECT DISTINCT ?pipelineTitle ?datasetTitle ?algorithm
?startTime ?duration ?prospective ?retrospective
  (GROUP_CONCAT(DISTINCT CONCAT(?param_label, ": ",
?param_value); SEPARATOR=", ") AS ?parameters)
  (GROUP_CONCAT(DISTINCT CONCAT(?metric_label, ": ",
?metric_value); SEPARATOR=", ") AS ?metrics)
WHERE {
  GRAPH <"""+id_exec+"""> {
    ?pipeline a provx:Pipeline ;
              dcterms:title ?pipelineTitle .

    ?dataset a provx:Dataset ;
             dcterms:title ?datasetTitle .

    ?run a provx:Run ;
         prov:startedAtTime ?startTime ;
         provx:Duration ?duration ;
         provx:ExtendedProvenance ?extendedProv .

    ?extendedProv provx:RetrospectiveProvenance ?retrospective .
    ?extendedProv provx:ProspectiveProvenance ?prospective .

    ?mlModel a provx:MLModel ;
             provx:Model ?model .

    ?model mlonto:Algorithm ?algorithm .

    ?model provx:Parameters ?paramNode .
    ?paramNode rdfs:label ?param_label ;
             provx:value ?param_value .

    ?model provx:Scores ?metricNode .
    ?metricNode rdfs:label ?metric_label ;
             provx:value ?metric_value .
}
}GROUP BY ?pipelineTitle ?datasetTitle ?algorithm ?startTime
?duration ?model ?retrospective ?prospective
```