Pontifícia Universidade Católica
do Rio de Janeiro

**Victor Feitosa de Carvalho Souza**

# Árvores de Decisão com Regras Explicáveis

**Dissertação de Mestrado**

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

Orientador: Prof. Eduardo Sany Laber

Rio de Janeiro
Março de 2023

PUC-Rio - Certificação Digital Nº 2112508/CA

Pontifícia Universidade Católica
do Rio de Janeiro

**Victor Feitosa de Carvalho Souza**

# Árvores de Decisão com Regras Explicáveis

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

**Prof. Eduardo Sany Laber**
Orientador
Departamento de Informática – PUC-Rio

**Prof. Marco Serpa Molinaro**
Departamento de Informática – PUC-Rio

**Prof. Claudson Ferreira Bornstein**
UFRJ

Rio de Janeiro, 15 de Março de 2023

PUC-Rio - Certificação Digital Nº 2112508/CA

Todos os direitos reservados. A reprodução, total ou parcial do trabalho é proibida sem a autorização da universidade, do autor e do orientador.

**Victor Feitosa de Carvalho Souza**

Bacharel em Engenharia de Computação (2017) pelo Instituto Militar de Engenharia (IME).

PUC-Rio - Certificação Digital Nº 2112508/CA

Ficha Catalográfica

Souza, Victor Feitosa de Carvalho

Árvores de Decisão com Regras Explicáveis / Victor Feitosa de Carvalho Souza; orientador: Eduardo Sany Laber. – 2023.

65 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2023.

Inclui bibliografia

1. Informática – Teses. 2. Aprendizado de Máquina. 3. Arvores de Decisão. 4. Modelos Explicáveis. 5. Classificação. 6. Algoritmos de Aproximação. I. Laber, Eduardo. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

## Agradecimentos

Ao meu orientador, prof. Eduardo Laber, por todo o suporte e ensinamentos durante o período do mestrado.

Ao meu pai, Victor, e à minha irmã, Kátia, pelo amor e estímulo incondicionais desde meu nascimento.

À minha esposa, Thayná, pelo amor e compreensão nesse período estressante do mestrado.

Aos meus "professores-padrinhos" Maria da Penha e Raimundo, pelo exemplo de dedicação ao magistério e apoio desde a minha infância.

A todos os meus professores, especialmente Luciano Castro, por terem marcado minha vida através do Ensino.

Aos meus amigos Rodrigo Morgado, Jéssica Viamonte, Eduardo Cesar, João Gris e Antonio Pedro por serem fonte de inspiração e alegria em diversos momentos.

Ao Exército Brasileiro, por prover as condições necessárias para a realização deste trabalho.

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

PUC-Rio - Certificação Digital Nº 2112508/CA

## Resumo

Souza, Victor Feitosa de Carvalho; Laber, Eduardo. **Árvores de Decisão com Regras Explicáveis**. Rio de Janeiro, 2023. 65p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

As árvores de decisão são estruturas comumente utilizadas em cenários nos quais modelos explicáveis de Aprendizado de Máquina são desejados, por serem visualmente intuitivas. Na literatura existente, a busca por explicabilidade em árvores envolve a minimização de métricas como altura e número de nós. Nesse contexto, definimos uma métrica de explicabilidade, chamada de *explanation size*, que reflete o número de atributos necessários para explicar a classificação dos exemplos. Apresentamos também um algoritmo, intitulado `SER-DT`, que obtém uma aproximação $O(\log n)$ (ótima se $P \neq NP$) para a minimização da altura no pior caso ou caso médio, assim como do *explanation size* no pior caso ou caso médio. Em uma série de experimentos, comparamos a implementação de `SER-DT` com algoritmos conhecidos da área, como `CART` e `EC`$^2$, além de testarmos o impacto de parâmetros e estratégias de poda nesses algoritmos. `SER-DT` mostrou-se competitivo em acurácia com os algoritmos citados, mas gerou árvores muito mais explicáveis.

PUC-Rio - Certificação Digital Nº 2112508/CA

## Palavras-chave

Aprendizado de Máquina; Arvores de Decisão; Modelos Explicáveis; Classificação; Algoritmos de Aproximação.

## Abstract

Souza, Victor Feitosa de Carvalho; Laber, Eduardo (Advisor). **Decision Trees with Explainable Rules**. Rio de Janeiro, 2023. 65p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Decision trees are commonly used structures in scenarios where explainable Machine Learning models are desired, as they are visually intuitive. In the existing literature, the search for explainability in trees involves minimizing metrics such as depth and number of nodes. In this context, we define an explainability metric, called explanation size, which reflects the number of attributes needed to explain the classification of examples. We also present an algorithm, called `SER-DT`, which obtains an $O(\log n)$ approximation (optimal if $P \neq NP$) for the minimization of depth in the worst/average case, as well as of explanation size in the worst/average case. In a series of experiments, we compared the `SER-DT` implementation with well-known algorithms in the field, such as `CART` and `EC`$^2$ in addition to testing the impact of parameters and pruning strategies on these algorithms. `SER-DT` proved to be competitive in terms of accuracy with the aforementioned algorithms, but generated much more explainable trees.

PUC-Rio - Certificação Digital Nº 2112508/CA

## Keywords

Machine Learning; Decision Trees; Explainable Models; Classification; Approximation Algorithms.

# Sumário



PUC-Rio - Certificação Digital Nº 2112508/CA

# Lista de figuras



PUC-Rio - Certificação Digital Nº 2112508/CA

PUC-Rio - Certificação Digital Nº 2112508/CA

# Lista de tabelas



PUC-Rio - Certificação Digital Nº 2112508/CA

PUC-Rio - Certificação Digital Nº 2112508/CA

## Lista de Abreviaturas

`CART` – *Classification and Regression Tree*

`EC`$^2$ – *Equivalence Class Edge Cutting Algorithm*

IA – Inteligência Artificial

`ID3` – *Iterative Dichotomiser* 3

`SER-DT` – *Short Explainable Rules for Decision Trees*

PUC-Rio - Certificação Digital Nº 2112508/CA

PUC-Rio - Certificação Digital Nº 2112508/CA

*Amo e adoro tudo o que é simples, tanto que às vezes pareço exigente!*

**Clarice Lispector**

# 1
# Introdução

Interpretabilidade em Inteligência Artificial é definida como a capacidade de descrever os elementos internos de um sistema de modo compreensível a um ser humano [1]. A interpretabilidade está intimamente relacionada com a **explicabilidade**, um termo mais geral que abrange tanto a busca por interpretabilidade quanto por completeza, que é a capacidade de descrever a operação de um sistema precisamente [1]. No presente trabalho, utilizaremos os termos "interpretabilidade" e "explicabilidade" como sinônimos.

A busca por explicabilidade em IA tem crescido nos últimos anos, o que se explica pela evolução de áreas como Ética em IA, sistemas militares de defesa, aplicações medicinais, entre outros campos de pesquisa que exigem o julgamento e interferência humana. A Figura 1.1 mostra o comportamento da busca no *Google* sobre o assunto "Inteligência Artificial Explicável" nos últimos 5 anos. Note que o gráfico apresenta uma clara tendência de aumento do interesse pelo tema.

Nesse contexto, as árvores de decisão são estruturas comumente usadas em situações nas quais modelos explicáveis são desejados. A Figura 1.2 ilustra uma árvore de decisão para a classificação de frutas. Cada passo no caminho da árvore é resultado de um teste explícito. Nesse sentido, podemos afirmar que as árvores de decisões já possuem naturalmente um grau de explicabilidade por serem facilmente compreendidas por um ser humano.

PUC-Rio - Certificação Digital Nº 2112508/CA

## 1.1
## Contribuições

O presente trabalho apresenta uma nova métrica de explicabilidade e um algoritmo que busca otimizá-la. Ademais, fornece um estudo experimental robusto que fornece uma comparação deste algoritmo com outros métodos amplamente utilizados na área.

A nova métrica criada, denominada *explanation size* (expl.), é definida como o número de atributos diferentes nos testes contidos no caminho da raiz de uma árvore até uma folha [1].

[1] *Vide* Seção 2 para uma ambientação com os conceitos básicos sobre árvores de decisão e Seção 3.1 para uma definição mais precisa de *explanation size*.

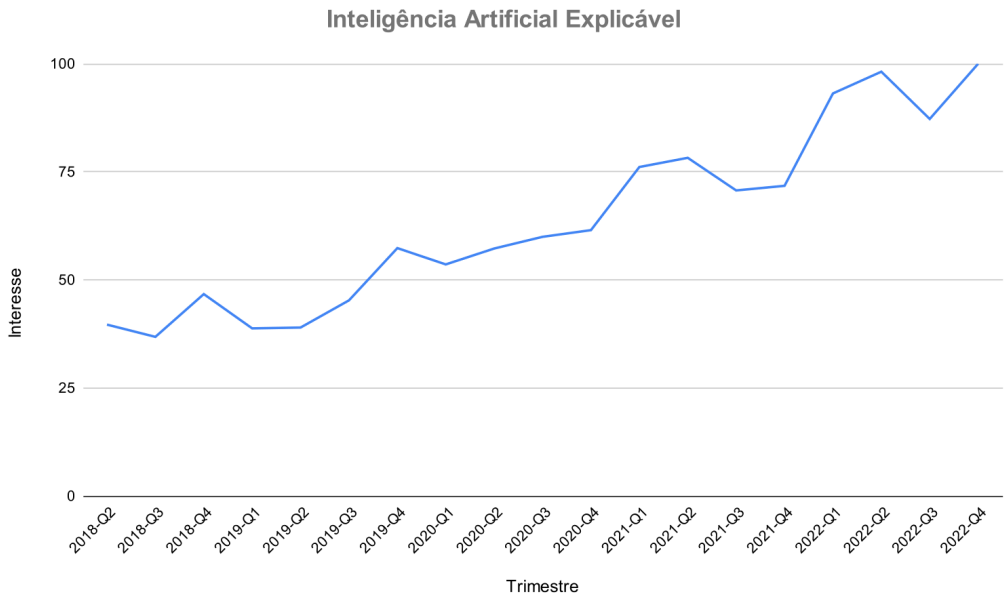


Figura 1.1: Interesse do assunto "Inteligência Artificial Explicável" nos últimos 5 anos, a cada trimestre. O eixo y (interesse) representa o volume de pesquisa no Google: um valor de 100 representa o pico de popularidade do assunto no período, e os outros valores representam o número proporcional de pesquisas do assunto em comparação ao ponto máximo. Fonte dos dados: [2].

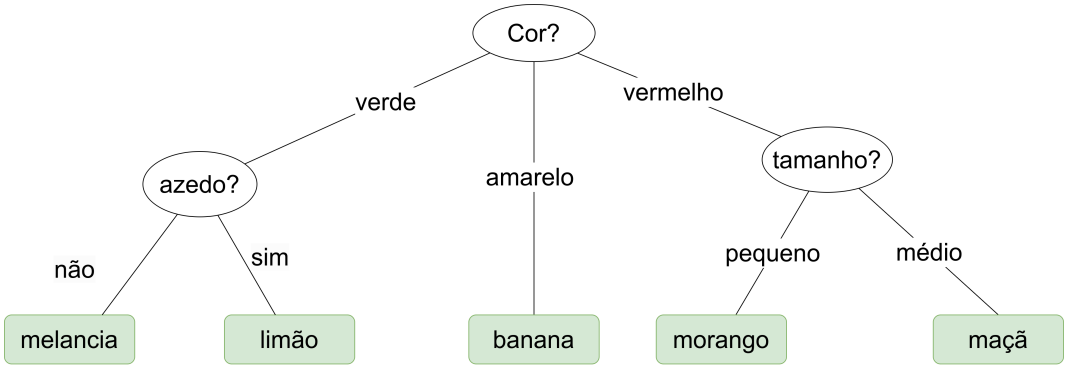


Figura 1.2: Exemplo de árvore de decisão para classificação de frutas. Cada nó da árvore é um teste e as arestas correspondem aos possíveis resultados de cada teste.

Como exemplo, a Figura 1.3 apresenta uma árvore gerada por `CART` [3], um famoso algoritmo desenvolvido para se construir árvore de decisões. A folha assinalada com uma borda retangular em destaque possui *explanation size* igual a 3. Nessa figura, cores diferentes representam atributos diferentes e as cores nas folhas representam os atributos que aparecem no caminho desde a raiz. Intuitivamente, quanto menor o explanation size, mais simples será a "explicação" da classificação dos exemplos contidos em uma folha; portanto, maior será o grau de explicabilidade.

PUC-Rio - Certificação Digital Nº 2112508/CA

O novo algoritmo, chamado `SER-DT` (*Short Explainable Rules for Decision Trees*), foi desenvolvido com o intuito de criar árvores que otimizem *explanation size*. Sendo $\boldsymbol{n}$ o número de exemplos a serem classificados, e considerando um cenário em que todos os exemplos são classificados corretamente, este algoritmo provê uma aproximação $O(\log n)$ para a minimização no pior caso e caso médio, tanto para *explanation size* como altura – o que é ótimo assumindo $P \neq NP$ ([4] e [5]). A Figura 1.4 apresenta uma árvore equivalente à Figura 1.3 (utilizando os mesmos hiperparâmetros), mas gerada por `SER-DT`. Note que esta árvore apresenta folhas com regras de classificação mais simples do que àquelas da figura anterior, o que dá uma noção das capacidades do algoritmo.

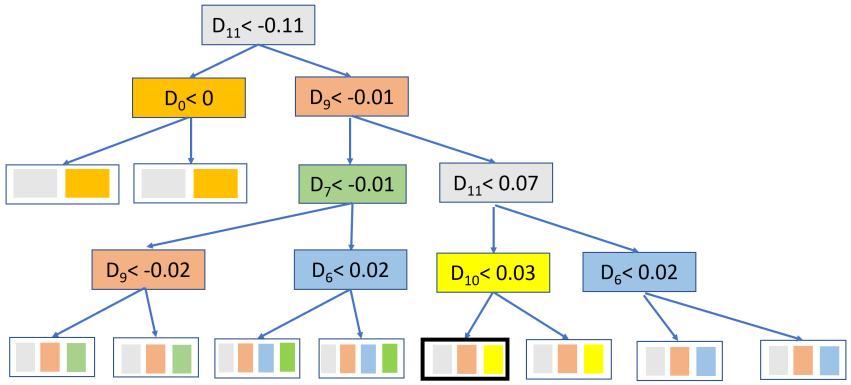


Figura 1.3: Árvore gerada por `CART` com altura máxima definida de 4. Nós internos com mesmos atributos são pintados da mesma cor. As cores nas folhas indicam os atributos utilizados para obter sua classificação. A folha assinalada possui *explanation size* de 3, por englobar 3 atributos distintos no caminho da raiz até a folha.

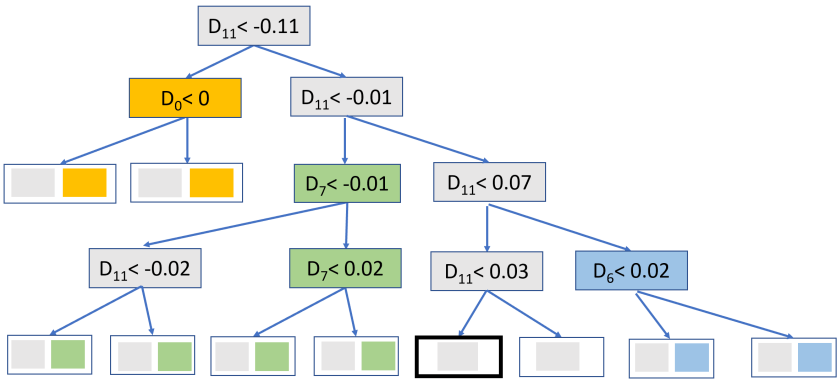


Figura 1.4: Árvore gerada por `SER-DT` com altura máxima definida de 4. Nós internos com mesmos atributos são pintados da mesma cor. As cores nas folhas indicam os atributos utilizados para obter sua classificação. Note que a árvore é mais simples do que a anterior. A folha assinalada possui *explanation size* de 1, menor do que a folha equivalente na árvore da figura anterior.

A Seção 2 expõe alguns conceitos básicos sobre árvore de decisões que serão utilizados ao longo deste trabalho. Logo em seguida, a Seção 3 apresenta

PUC-Rio - Certificação Digital Nº 2112508/CA

os modelos teóricos necessários para a definição do problema de classificação em árvores de decisão e exibe um pseudocódigo descrevendo `SER-DT`. As provas dos teoremas apresentados são referenciadas ao artigo [6]; a escolha pela sua omissão no presente trabalho deve-se à pouca participação deste autor na obtenção destas. Ainda assim, apresentamos, logo após um teorema, as estratégias utilizadas e intuições subjacentes na sua prova.

Em relação aos resultados experimentais, a Seção 4 detalha os procedimentos adotados. Foram utilizados 20 datasets reais obtidos de fontes públicas. Para cada um deles, foram executados experimentos utilizando nosso algoritmo e outros conhecidos da área, como `CART` [3] e `EC`$^2$ [7]. A comparação com esses algoritmos mostrou que `SER-DT` é competitivo em relação à acurácia no conjunto de teste, além de gerar árvores mais explicáveis (Seções 4.4 e 4.9). Além disso, experimentos adicionais foram realizados com a finalidade de avaliar `SER-DT` em cenários diversos, que englobam: averiguação do efeito da variação de hiperparâmetros (Seções 4.3 e 4.5), critérios de parada (Seção 4.6) e de impureza dos nós (Seção 4.7); implementação e análise de uma estratégia de pós-poda para simplificação das árvores (Seção 4.10).

PUC-Rio - Certificação Digital Nº 2112508/CA

## 1.2 Trabalhos Relacionados

Nosso trabalho pode ser relacionado a uma gama de estudos que apresentam métodos com garantias teóricas para a construção de árvores de decisões de "baixa complexidade" em relação ao número de folhas, número de nós e variantes ([7], [8], [9], [10], [11]).

No campo da interpretabilidade, [12] propõe que a homogeneidade na divisão dos nós (no sentido de nós filhos possuírem objetos com classes igualmente distribuídas) é um fator importante para interpretabilidade, e apresenta um critério de divisão de nós que considera homogeneidade e altura das folhas. [13] e [14] utilizam Programação Linear Inteira para construir árvores com altura máxima definida; já [15] emprega Programação Dinâmica para construir árvores de decisão com poucas folhas e que otimizem métricas como F-score e AUC. Em contraste ao nosso caso, esses trabalhos não apresentam algoritmos que executem em tempo polinomial, o que pode ser um limitante para cenários com datasets grandes.

Considerando garantias teóricas, geralmente os trabalhos na área se dedicam a minimizar métricas como o número de nós ou altura da árvore no pior caso ou caso médio [16]. É fato conhecido que construir uma árvore de decisão que minimiza o pior caso ou caso médio, dentre aquelas que se ajustam ao conjunto de treino, não admite uma aproximação $o(\log n)$ a menos

que $P = NP$ ([4] e [5]). Existem, no entanto, alguns algoritmos com uma aproximação ótima de $O(\log n)$ ([7], [8], [9], [10]). Enquanto esses métodos fornecem garantias em cenários com ou sem ruído, com custos dos testes não uniformes, entre outros, o nosso algoritmo é o único que provê garantias em relação a *explanation size*.

PUC-Rio - Certificação Digital Nº 2112508/CA

# 2
# Fundamentos

Nesta seção, apresentaremos alguns conceitos relacionados às árvores de decisão necessários para a compreensão das discussões subsequentes.

## 2.1
## A Tarefa de Classificação

Neste trabalho, estamos interessados no problema de classificação de objetos. Mais especificamente, dado um conjunto de entrada (chamado também de treino) composto por $n$ objetos rotulados com uma classe dentre um conjunto $C$, queremos encontrar um modelo que possa classificar precisamente novos objetos contidos em outro conjunto (denominado conjunto de teste). Esse modelo, no nosso caso, será uma árvore de decisão.

Para exemplificar, tomemos o seguinte exemplo, adaptado de [17, pág. 19]. A tabela 2.1 representa o conjunto de treino. Cada linha na tabela caracteriza um objeto (ou elemento) e as 4 primeiras colunas representam atributos que um elemento pode assumir. A última coluna representa a classe com a qual o objeto é rotulado. Neste exemplo, a classe significa a recomendação de exercitar-se (ou não) conforme as condições do tempo, temperatura, umidade e vento. Dados os exemplos de treino, a tarefa é encontrar uma árvore de decisão que possa responder à pergunta "Devo me exercitar ou não?" para outras condições de tempo, temperatura, etc., que não se encontram na tabela.

Uma das possíveis árvores de decisão para esse problema é apresentada na Figura 2.1. Nela, cada nó está associado a um teste sobre um atributo, e as ramificações ocorrem segundo as possíveis respostas a esse teste. O nó no topo da figura é chamado "raiz"; os de último nível, são as "folhas". Cada folha está associada a um dos rótulos possíveis. Para se classificar um objeto, basta aplicar os testes para esse objeto começando pela raiz da árvore, seguindo os resultados dos testes, até se chegar a uma folha. O rótulo da folha determina qual a classe que tal objeto é rotulado pela árvore.

Note que neste exemplo, a árvore classifica corretamente todos os objetos do conjunto de teste. Por isso, consideramos este um exemplo de **classificação**

PUC-Rio - Certificação Digital Nº 2112508/CA

**exata**, o que nem sempre é o caso. Nesse sentido, podemos definir a **acurácia** nos conjuntos de treino e teste.

**Definição 2.1** ***Acurácia no conjunto de treino ou teste***

*Dada uma árvore de decisão $T$ e um objeto $o$, seja $t(o)$ a classe rotulada atribuída ao objeto pela árvore, e $c(o)$ a sua classificação real. Para um conjunto $O$ com $n$ objetos de treino (respectivamente teste), definimos a acurácia no conjunto de treino (respectivamente teste) como a fração dos objetos em $O$ rotulados corretamente pela árvore, ou seja,*

$$\frac{|o \in O; c(o) = t(o)|}{|O|}$$

Tabela 2.1: Exemplo de dados de treinamento com 4 atributos. Cada linha da tabela é um exemplo, e a última coluna representa a classe correspondente.

| Condição do tempo | Temperatura (°C) | Umidade (%) | Vento? | Classe |
|---|---|---|---|---|
| ensolarado | 24 | 70 | verdadeiro | Exercite-se |
| ensolarado | 27 | 90 | verdadeiro | Não se exercite |
| ensolarado | 29 | 85 | falso | Não se exercite |
| ensolarado | 22 | 95 | falso | Não se exercite |
| ensolarado | 20 | 70 | falso | Exercite-se |
| nublado | 22 | 90 | verdadeiro | Exercite-se |
| nublado | 28 | 78 | falso | Exercite-se |
| nublado | 18 | 65 | verdadeiro | Exercite-se |
| nublado | 27 | 75 | falso | Exercite-se |
| chuvoso | 22 | 80 | verdadeiro | Não se exercite |
| chuvoso | 18 | 70 | verdadeiro | Não se exercite |
| chuvoso | 24 | 80 | falso | Exercite-se |
| chuvoso | 20 | 80 | falso | Exercite-se |
| chuvoso | 21 | 96 | falso | Exercite-se |

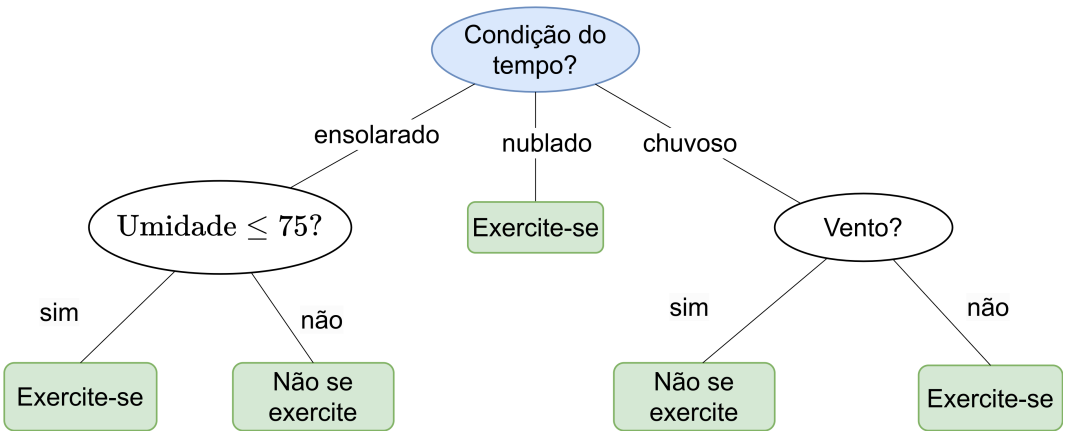


Figura 2.1: Árvore de decisão relacionada aos dados da Tabela 2.1.

PUC-Rio - Certificação Digital Nº 2112508/CA

## 2.2 Atributos Categóricos e Ordinais

A construção de árvores de decisão é feita a partir de um conjunto de elementos (ou objetos) de treinamento, como exposto na seção anterior. Cada um destes elementos são caracterizados através de um conjunto de atributos. Alguns dos tipos desses atributos são:

- **atributos categóricos** – usados para representar categorias ou rótulos e não guardam relação de ordem intrínseca. Um exemplo são os tipos de cor dos olhos: azul, verde, castanho, etc. São usualmente não numéricos.
- **atributos ordinais** – guardam uma relação de ordem entre si. Um exemplo é a altura de uma pessoa expressa em centímetros. São geralmente numéricos.

Muitos dos algoritmos em Machine Learning esperam atributos numéricos (e algumas vezes ordinais) nos exemplos de entrada. Uma das possibilidades para se converter um atributo categórico em numérico é o procedimento conhecido por *One-Hot Encoding*.

**One-Hot Encoding** [18, pág.78]: dado um atributo categórico $a$ que possui $k$ opções $\{a_1, ...a_k\}$, substituímo-lo por $k$ novos atributos binários de nomes iguais a $a_k, ..., a_k$. Para algum objeto $\boldsymbol{o}$, o valor de $a_i$ no objeto $\boldsymbol{o}$ será 1 se o valor de $a$ em $\boldsymbol{o}$ for igual a $a_i$; será 0, caso contrário.

Prefere-se utilizar *One-Hot Encoding* em vez de simplesmente se atribuir inteiros de 1 a $n$ para um atributo categórico que possui $n$ possibilidades. Caso isso fosse feito, estaríamos criando uma relação de ordem que muitas vezes não é desejada nos atributos categóricos.

## 2.3 Construção de Árvores de Decisão

Vários algoritmos foram propostos para se construir árvores de decisão, dentre eles `ID3` [19], `C4.5` [17] e `CART` [3]. No entanto, a maioria destes são similares ao método de Hunt [20], criado na década de 1960, cujas principais ideias são apresentadas no Algoritmo 1.

PUC-Rio - Certificação Digital Nº 2112508/CA

**Algoritmo 1:** Hunt

**Input:** Um nó $\nu$ alcançado por um conjunto $S$ de objetos.

**if** *todos os objetos em $S$ são da mesma classe $c$* **then**
  $\nu$ é uma folha com rótulo $c$
  **return**
**end**
**if** $S = \varnothing$ **then**
  $\nu$ é uma folha com rótulo igual à classe dominante do pai de $\nu$.
  **return**
**end**
**if** *$S$ contém exemplos de classes diferentes* **then**
  Utilize um atributo $a$ para dividir $S$ em uma partição $P = \{S_i\}$.
  Para cada subconjunto $S_i \in P$, crie um nó $\nu_i$ e aplique o algoritmo recursivamente para cada um desses nós.
**end**

Note que o Algoritmo de Hunt procede até que não haja folha contendo exemplos de classes distintas. Tomemos como exemplo novamente os dados de treinamento da tabela 2.1. Utilizando o Algoritmo de Hunt e esses dados de exemplo, podemos construir a árvore da Figura 2.1. O algoritmo primeiramente seleciona o atributo "Condição do tempo" para dividir o conjunto inicial de exemplos em três subconjuntos. Dos exemplos que possuem valor "nublado", todos possuem classe "Exercite-se" e uma folha é formada. Dos exemplos que são "ensolarados", selecionamos o atributo "Umidade" para dividir os exemplos que possuem um valor menor ou maior que 75. Similarmente, dentre os exemplos que são "chuvosos", selecionamos o atributo "vento" para criar as últimas folhas. Desse modo, todas as folhas possuem exemplos com a mesma classe, e o algoritmo é finalizado.

Observe que o Algoritmo de Hunt não especifica qual o critério utilizado para a seleção de atributos. A próxima seção descreve o procedimento adotado por alguns algoritmos para essa finalidade.

## 2.4 Medidas de Impureza

A seleção de atributos em alguns algoritmos de construção de árvores de decisão utiliza a noção de **impureza** na escolha de divisão dos nós. Um nó é considerado mais "impuro" quanto mais homogênea for a proporção das classes dos objetos contidos no nó. Em geral, busca-se encontrar divisões que gerem nós menos impuros, ou seja, que apresentem uma ou mais classes dominantes em relação às outras presentes no nó. Intuitivamente, caso possua somente

PUC-Rio - Certificação Digital Nº 2112508/CA

uma classe dominante, um nó poderia ser rotulado com essa classe; se houver mais de uma, o nó poderia ser subdivido de modo a separar tais classes.

Como exemplo de medidas de impureza há o **Índice Gini**, utilizado pelo algoritmo `CART` e definido na seção 4.2. Outra medida é a **Entropia**, definida na seção 4.7. Os algoritmos `ID3` e `C4.5`, por exemplo, utilizam o **Ganho de Informação**, que é a diferença de entropia entre um nó e seus filhos.

Cabe salientar que os algoritmos para árvores de decisão são geralmente "gulosos" porque buscam minimizar uma medida de impureza a cada divisão dos nós.

## 2.5 Poda

Um questionamento ao se construir árvores de decisão é o critério de "parada" ou, melhor dizendo, a situação em que um algoritmo não subdivide mais um nó e uma folha é criada. No Algoritmo de Hunt, um nó não é mais subdivido se todos os objetos pertencerem à mesma classe ou se não contiver objetos. Outros critérios podem ser adotados, como uma altura máxima pré estabelecida; comparação de impureza entre nó pai e filhos; e número mínimo de objetos em um nó. Os critérios de parada são considerados mecanismos de **pré-poda** porque são procedimentos de simplificação realizados durante o processo de construção de árvores. Eles são interessantes para não gerar uma árvore complexa demais, o que pode levar tanto a *overfitting* como perda em explicabilidade.

Outra possibilidade de simplificação de árvores é a **pós-poda**, que é a remoção de parte da estrutura da árvore após a sua criação. Essa remoção geralmente envolve a junção recursiva de folhas até um certo limite no caminho da folha à raiz. A Seção 4.10 apresenta um método de pós-poda implementado e testado em uma série de experimentos.

PUC-Rio - Certificação Digital Nº 2112508/CA

# 3
# Algoritmos para Minimizar *Explanation Size*

## 3.1
## Modelo

Apresentaremos a seguir o modelo teórico que será utilizado no decorrer das análises presentes nesta dissertação. Tal modelo emprega uma terminologia similar aos trabalhos relacionados da área, como [9] e [21]. Salientamos que neste trabalho estamos interessados somente no problema de classificação em árvores de decisão binárias, pois o aumento na ramificação dos nós tende a gerar árvores menos explicáveis [22].

Definimos uma instância $I$ como sendo o conjunto de atributos ordinais $A$ e um conjunto de objetos $O$. Cada objeto $o \in O$ é descrito pelo valor que possui em cada atributo $a \in A$ (denotado por $a(o)$). Da mesma forma, cada objeto $o$ possui uma classificação $c(o)$ em um conjunto de classes $C$. Para classificar os objetos, utilizamos testes sequenciais da forma "$a(o) \leq t$?" para algum atributo $a$ e um valor $t$, denominado *threshold*.

A figura 3.1 ilustra uma árvore de decisão binária $T$ em que cada nó interno $\tau$ é representado por um teste "$a(o) \leq t$?" e cada aresta de um nó para seus filhos está associada a um dos resultados "$a(o) \leq t$" ou "$a(o) > t$". Como os testes envolvem uma inequação, ressaltamos que os atributos precisam ser ordinais. Caso haja atributos categóricos, podemos transformá-los via *one-hot encoding* (*vide* Seção 2.2).

Representamos um nó pelo par atributo/*threshold* $(a, t)$. Cada folha $\ell$ em $T$ é associada a uma classe $c \in C$. Dizemos que um objeto $o$ alcança um nó $\tau$ de $T$ se ele satisfaz todos os testes existentes da raiz até o nó $\tau$. Além disso, para todo $o \in O$, denotamos $\ell(o)$ como a folha que o objeto $o$ alcança. Finalmente, consideramos um modelo de **classificação exata**, isto é, uma árvore de decisão deve classificar corretamente todos os objetos da instância – mais especificamente, a classe associada a $\ell(o)$ é igual a $c(o)$, $\forall o \in O$.

PUC-Rio - Certificação Digital Nº 2112508/CA

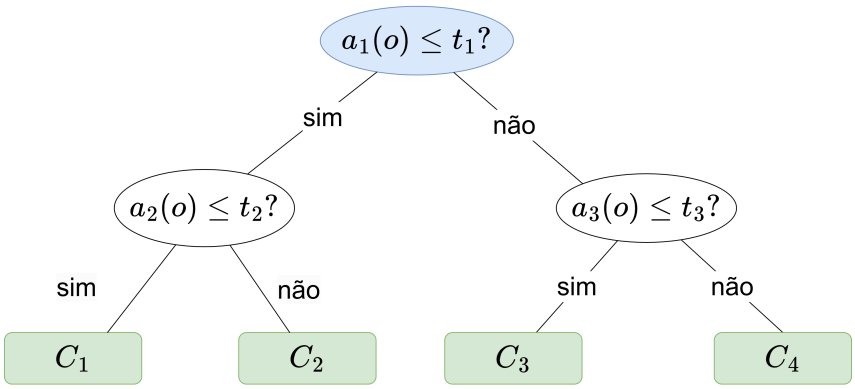


Figura 3.1: Exemplo de árvore de decisão binária. A raiz está representada pelo nó azul e as folhas pelos nós verdes. Cada nó interno está associado a um teste da forma "$a(o) \leq t$?" e cada folha está associada a uma classe.

Apresentaremos agora as métricas de explicabilidade utilizadas neste trabalho.

Dada uma árvore $T$, a altura de uma folha $\ell$ é o número de testes (nós internos) no caminho da raiz da árvore até a folha $\ell$, denotada por $\text{depth}(\ell)$. A seguir definiremos a altura no pior caso e caso médio.

**Definição 3.1** ***Altura de uma árvore no pior caso.***

*A altura no pior caso da árvore $T$ é dada pela altura máxima das folhas:*

$$\text{depth}_{\text{wc}}(T) := \max_{\ell \in \text{leaf}(T)} \text{depth}(\ell),$$

*onde* $\text{leaf}(T)$ *é o conjunto de folhas presentes na árvore $T$.*

Para definir a altura média de uma árvore, consideramos também que cada objeto $o$ possui um peso não negativo $w(o) \in \mathbb{R}_+$ indicando sua importância ou probabilidade de ocorrência. Denotamos por $w(\ell)$ a soma dos pesos de todos os objetos que alcançam uma folha $\ell$.

**Definição 3.2** ***Altura de uma árvore no caso médio.***

*Definimos a altura média da árvore $T$ como a soma ponderada das alturas das folhas em relação aos pesos:*

$$\text{depth}_{\text{avg}}(T) := \sum_{\ell \in \text{leaf}(T)} w(\ell) \cdot \text{depth}(\ell),$$

*onde* $\text{leaf}(T)$ *é o conjunto de folhas presentes na árvore $T$.*

Apresentamos agora novas métricas de explicabilidade considerando a noção de *explanation size*. O *explanation size* de uma folha $\ell$, representado por $\text{expl}(\ell)$, é definido como o número de atributos distintos existentes nos testes do caminho da raiz de uma árvore $T$ até $\ell$.

PUC-Rio - Certificação Digital Nº 2112508/CA

**Definição 3.3** ***Explanation size no pior caso***

*O explanation size no pior caso de uma árvore $T$ é definido como o valor máximo dos explanation sizes das folhas:*

$$\mathrm{expl}_{\mathrm{wc}}(T) := \max_{\ell \in \mathrm{leaf}(T)} \mathrm{expl}(\ell),$$

*onde* $\mathrm{leaf}(T)$ *é o conjunto de folhas presentes na árvore* $T$*.*

**Definição 3.4** ***Explanation size no caso médio***

*O explanation size no caso médio de uma árvore $T$ é definido como a média ponderada dos explanation sizes das folhas:*

$$\mathrm{expl}_{\mathrm{avg}}(T) := \sum_{\ell \in \mathrm{leaf}(T)} w(\ell) \cdot \mathrm{expl}(\ell)$$

*onde* $\mathrm{leaf}(T)$ *é o conjunto de folhas presentes na árvore* $T$*.*

Nosso objetivo é obter árvores com *explanation sizes* no pior caso ou caso médio tão pequenos quanto possível. Assinalamos por $\mathrm{depth}^*_{\mathrm{wc}} = \mathrm{depth}^*_{\mathrm{wc}}(I)$ como a menor altura de pior caso de uma árvore que resolve a instância $I$; definimos os valores $\mathrm{depth}^*_{\mathrm{avg}}$, $\mathrm{expl}^*_{\mathrm{wc}}$ e $\mathrm{expl}^*_{\mathrm{avg}}$ de forma análoga.

## 3.2 Garantias Teóricas

Como exposto na subseção anterior, nosso objetivo é encontrar árvores com as menores métricas possíveis de altura e *explanation size*. No entanto, intuitivamente, poderíamos imaginar que para se obter uma árvore com altura pequena talvez fosse necessário utilizar vários atributos distintos para se alcançar uma classificação efetiva, o que desencadearia em um *explanation size* grande. Da mesma forma, para conseguirmos uma árvore com *explanation size* pequeno, talvez fosse necessário um número grande de testes para uma classificação utilizando poucos atributos, o que resultaria em uma altura grande. Apesar dessa suspeita, é possível mostrar que existem árvores que possuem *explanation size* ótimo e altura quase ótima.

**Teorema 1** *Dada uma instância do Problema de Classificação em Árvores de Decisão:*

1. *(Métricas de pior caso) Existe uma árvore binária $T$ que satisfaz simultaneamente*

   - $\mathrm{expl}_{\mathrm{wc}}(T) = \mathrm{expl}^*_{\mathrm{wc}}$
   - $\mathrm{depth}_{\mathrm{wc}}(T) \leq 2\ \mathrm{depth}^*_{\mathrm{wc}} + \log n$

PUC-Rio - Certificação Digital Nº 2112508/CA

2. *(Métricas de caso médio) Existe uma árvore binária $T$ que satisfaz simultaneamente*

   - $\text{expl}_{\text{avg}}(T) = \text{expl}^*_{\text{avg}}$
   - $\text{depth}_{\text{avg}}(T) \leq 2\ \text{depth}^*_{\text{avg}} + W \log n$,

*onde $W$ é a soma dos pesos dos objetos da instância.*

**Prova**: *vide* [6, Apêndice B].

A seguir apresentaremos uma intuição da prova do pior caso, por ser mais simples. Para tal, definimos uma **árvore multivia** como sendo uma árvore onde são usados testes $k$-ários em vez de testes binários. Cada teste utilizando um atributo $a$ separa um nó $\nu$ em todas as possibilidades que $a$ admite entre os exemplos contidos em $\nu$. Como exemplo, se um atributo $a$ possui 5 valores distintos entre os objetos pertencentes ao nó, serão criados 5 nós filhos a partir deste nó considerado.

Primeiramente, mostraremos que existe uma equivalência entre uma árvore de decisão binária e um árvore multivia em relação ao *explanation size* no pior caso. Para comprovar isso, note que existe uma árvore multivia $M^*$ que é simultaneamente ótima em relação à altura no pior caso e *explanation size* no pior caso, já que um atributo só precisa aparecer uma vez no caminho da raiz até um nó. Além disso, $M^*$ apresenta $\text{depth}_{\text{wc}}(M^*) = \text{expl}_{\text{wc}}(M^*) \leq \text{expl}^*_{\text{wc}}$, já que, intuitivamente as árvores multivias são mais informativas que as árvores binárias. Por outro lado, podemos transformar uma árvore multivia em uma árvore binária $T$ simulando cada teste $k$-ário no atributo $a$ usando múltiplos testes "$a(o) \leq t$?" variando-se $t$ mas mantendo-se o atributo $a$. Como o *explanation size* conta somente o número de atributos distintos em um caminho, esta transformação não altera o *explanation size*, e daí $\text{expl}_{\text{wc}}(M^*) = \text{expl}_{\text{wc}}(T) \geq \text{expl}^*_{\text{wc}}$. Daí, temos que $\text{expl}_{\text{wc}}(M^*) = \text{expl}^*_{\text{wc}}$

Para provar o Teorema 1, tomamos uma árvore multivia ótima $M^*$ e a transformamos em uma árvore binária, como mostrado acima. No entanto, essa transformação pode não ser eficiente: enquanto o *explanation size* é preservado, pode-se aumentar demais a altura da árvore binária gerada. Podemos usar uma ideia baseada nos **códigos alfabéticos**, uma noção da Teoria de Códigos. O seguinte resultado, adaptado de [23, pág. 19] para árvores de decisões, fornece uma condição suficiente para a existência de códigos alfabéticos com tamanhos predefinidos.

**Lema 1.1** *Considere uma instância do problema de classificação $I$ com $n$ objetos $o_1, ..., o_n$, mas somente um atributo. Então, para quaisquer inteiros positivos $d_1, ..., d_n$ tais que $\sum_{i=1}^{n} 2^{-d_i} \leq \frac{1}{2}$, existe uma árvore de decisão binária*

PUC-Rio - Certificação Digital Nº 2112508/CA

*com n folhas de alturas $d_1, ..., d_n$ indo da esquerda para a direita na árvore, tal que o objeto $o_i$ alcança a folha de altura $d_i$, $\forall i \in \{1, ..., n\}$.*

Para cada nó $\nu$ em $M^*$ (correspondendo a um atributo $a$, com filhos $ch_1, ch_2, ....$), consideramos todos os objetos que alcançam um filho $ch_i$ como sendo um único objeto, com o valor correspondente no atributo $a$. Utilizando o lema anterior, substituímos $\nu$ por uma árvore binária $A$ onde cada folha corresponde aos objetos em $ch_i$ a uma altura em $A$ de $\ell_i = \lceil \log \frac{n(\nu)}{n(ch_i)} \rceil + 1$, onde $n(\alpha)$ é o número de exemplos que alcançam um nó $\alpha$ em $M^*$. A Figura 3.2 ilustra essa transformação de árvore multivia para uma binária.

Aplicando esse processo, a árvore binária $T$ obtida possui os mesmos *explanation sizes* que $M^*$, logo $\text{expl}_{\text{wc}}(T) = \text{expl}^*_{\text{wc}}$. Além disso, em relação à altura no pior caso, qualquer caminho $P$ da raiz a uma folha em $T$ tem um caminho correspondente $P_{M^*} = \nu_0, \nu_1, ...$ em $M^*$, e o tamanho de $P$ é dado por:

$$\sum_{i=0}^{|P_{M^*}|-1} \left( \left\lceil \log \frac{n(\nu_i)}{n(\nu_{i+1})} \right\rceil + 1 \right) \leq \sum_{i=0}^{|P_{M^*}|-1} \left( \log \frac{n(\nu_i)}{n(\nu_{i+1})} + 2 \right) =$$
$$\log n - \log n(\nu_{|P_{M^*}|}) + 2|P_{M^*}| \leq$$
$$\log n + 2|P_{M^*}|$$

Olhando para o maior caminho da raiz até uma folha, temos que:

$$\text{depth}_{\text{wc}}(T) \leq \log n + 2 \ \text{depth}_{\text{wc}}(M^*) \leq \log n + 2 \ \text{depth}^*_{\text{wc}},$$

onde a última desigualdade decorre do fato de $M^*$ ser ótima em relação à altura no pior caso. Isso resulta no Item 1 do Teorema 1.

Nessa prova, a construção de árvores pode ser custosa, na prática, pois envolve obter os códigos alfabéticos para transformar cada árvore multivia em uma sequência de testes binários. Uma alternativa é mostrada na Seção 3.3, que apresenta um algoritmo que obtém a mesma garantia teórica e apresenta bom desempenho nos experimentos listados na Seção 4.

**Observação 1.1** *Os limites aditivos descritos no Teorema 1 implicam, da mesma forma, aproximações multiplicativas de $O(\log n)$:*

$$\text{depth}_{\text{wc}}(T) \leq \frac{3 \log n}{\log \ c} \cdot \text{depth}^*_{\text{wc}}$$

$$\text{depth}_{\text{avg}}(T) \leq 3 \log n \cdot \text{depth}^*_{\text{avg}},$$

*onde c é o número de classes.*

PUC-Rio - Certificação Digital Nº 2112508/CA

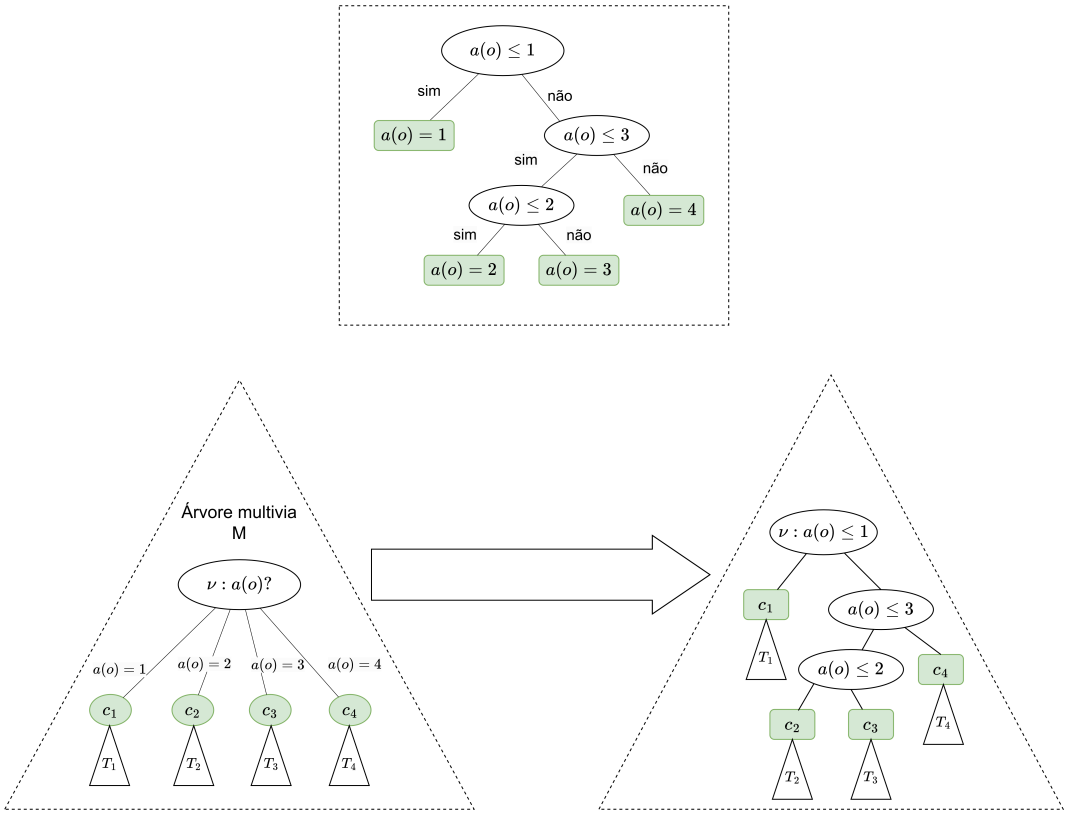


Figura 3.2: Transformação de uma árvore multivia para binária. A árvore superior representa a árvore de decisão obtida pelo Lema 1.1. A árvore multivia da esquerda é transformada na árvore binária da direita através da utilização dos *thresholds* obtidos. Note que os testes necessários na árvore binária para se obter os nós $c_1, c_2, c_3$ e $c_4$ são todos realizados sobre o mesmo atributo $a$.

Dados os resultados existenciais listados no Teorema 1, um questionamento é se é possível obter uma árvore que é ótima simultaneamente em altura e *explanation size*. O resultado a seguir mostra que a resposta é negativa, além de que os limites apresentados não podem ser melhorados.

**Teorema 2** *Seja $c$ e $n$ tais que $n \geq k \cdot c$ para uma constante $k$ suficientemente grande. Então, $\forall \alpha \in [\frac{1}{2 \log c}, \frac{1}{2}(\frac{\log(n/2)}{\log c} - 1)]$, existe uma instância $I$ com $n$ objetos e $c$ classes tais que toda árvore de decisão $T$ para essa instância possui*

$$\text{depth}_{\text{wc}}(T) > \frac{\alpha}{2} \cdot \text{depth}^*_{\text{wc}}$$

*ou*

$$\text{expl}_{\text{wc}}(T) > \frac{1}{4\alpha} \cdot \log\left(\frac{n}{c}\right) \cdot \text{expl}^*_{\text{wc}}$$

**Prova**: *vide* [6, Anexo C].

**Observação 2.1** *Para se compreender melhor esse limite inferior, considere tomar $\alpha$ como o máximo possível, isto é, $\alpha \approx \frac{1}{2}\frac{\log n}{\log c}$. Neste caso, o teo-*

PUC-Rio - Certificação Digital Nº 2112508/CA

*rema diz que qualquer árvore $T$ com* $\text{depth}_{\text{wc}}(T) \lesssim \frac{1}{4}\frac{\log n}{\log c}\text{depth}^*_{\text{wc}}$ *tem de possuir* $\text{expl}_{\text{wc}}(T) \geq \Omega((1 - \frac{\log c}{\log n})\log c))\text{expl}^*_{\text{wc}}$*, que é* $\Omega(\log n)\text{expl}^*_{\text{wc}}$*, se definirmos* $n = c^k$ *para alguma constante* $k$ *suficientemente grande. Comparando isso ao Teorema 1 e à Observação 1.1, enquanto pode haver uma árvore com* $\text{depth}_{\text{wc}}(T) \leq \frac{3\log n}{\log c}\text{depth}^*_{\text{wc}}$ *e* $\text{expl}_{\text{wc}}(T)$ *ótimo, se o fator de aproximação na altura for uma constante menor, perde-se um fator logarítmico na aproximação do explanation size.*

Para se provar o Teorema 2, é construída uma instância que satisfaça os limites mínimos estabelecidos para $\text{depth}_{\text{wc}}(T)$ e $\text{expl}_{\text{wc}}(T)$ para qualquer árvore de decisão $T$. As principais ideias utilizadas para a construção dessa instância são expostas a seguir.

Definiremos um conjunto de $n$ objetos e de classes $C = \{0, 1, ..., c-1\}$ com alternância de classes entre os objetos. Isto é, o objeto 1 possui classe 0, o objeto 2 possui classe 1, ..., o objeto $c$ possui classe $c-1$; então, ocorre uma repetição: o objeto $c+1$ possui classe 0, e assim por diante.

O primeiro atributo da instância, denominado $A$, possui $n$ valores possíveis e fornece a identidade dos objetos. Ou seja, o objeto $i$ possui um valor no atributo $A$ igual a $i$.

Para definir os outros atributos, primeiramente consideramos um parâmetro $t \in (\frac{2c}{n}, \frac{1}{2})$. Esses outros atributos são especificados de forma a separar uma fração $1-t$ do número de exemplos que possuem uma classe $\neq 0$ de todos os outros objetos. A ideia é que os nós de maior altura em uma árvore binária de altura ótima satisfaçam as seguintes condições:

- Todos os objetos de classe 0 percorrem o caminho mais à esquerda;
- Somente uma fração $t\frac{n}{c}$ dos objetos de cada classe $\neq 0$ seguirão esse caminho mais à esquerda;
- Todos os outros $(1-t)\frac{n}{c}$ objetos de cada classe $\neq 0$ são classificados corretamente pelos nós de maior altura na árvore.

Essas condições são ilustradas na Figura 3.3. Nesta figura, os nós de maior altura classificam corretamente uma fração $(1-t)$ dos objetos para cada classe distinta de zero, e o restante é deixado para ser resolvido no nó $\nu$ mais à esquerda. Então, uma nova subárvore com raiz nesse nó mais à esquerda é construída para resolver uma nova fração $(1-t)$ dos objetos restantes, para cada classe distinta de zero, e assim por diante.

A ideia da prova dos limites mínimos expressos no Teorema 2 é que, para toda árvore que não emprega todos os testes nesses outros atributos binários (na tentativa de reduzir o número de testes, e consequentemente, o *explanation*

PUC-Rio - Certificação Digital Nº 2112508/CA

*size*), os objetos de classe distinta de zero que seriam classificados corretamente por esses testes removidos acabariam alcançando o caminho mais à esquerda, juntos aos objetos de classe 0. Então, deveriam ser utilizados testes com o atributo $A$ para classificar esses objetos corretamente. No entanto, tais objetos estariam "uniformemente" distribuídos em $[n]$, de modo que apareceriam alternadamente junto aos objetos de classe 0. A consequência é que um número grande de testes com o atributo $A$ deveriam ser utilizados para separar os objetos de classe 0 daqueles de classe diferente de 0 nesse caminho mais à esquerda, o que resultaria em um aumento significativo na altura da árvore.

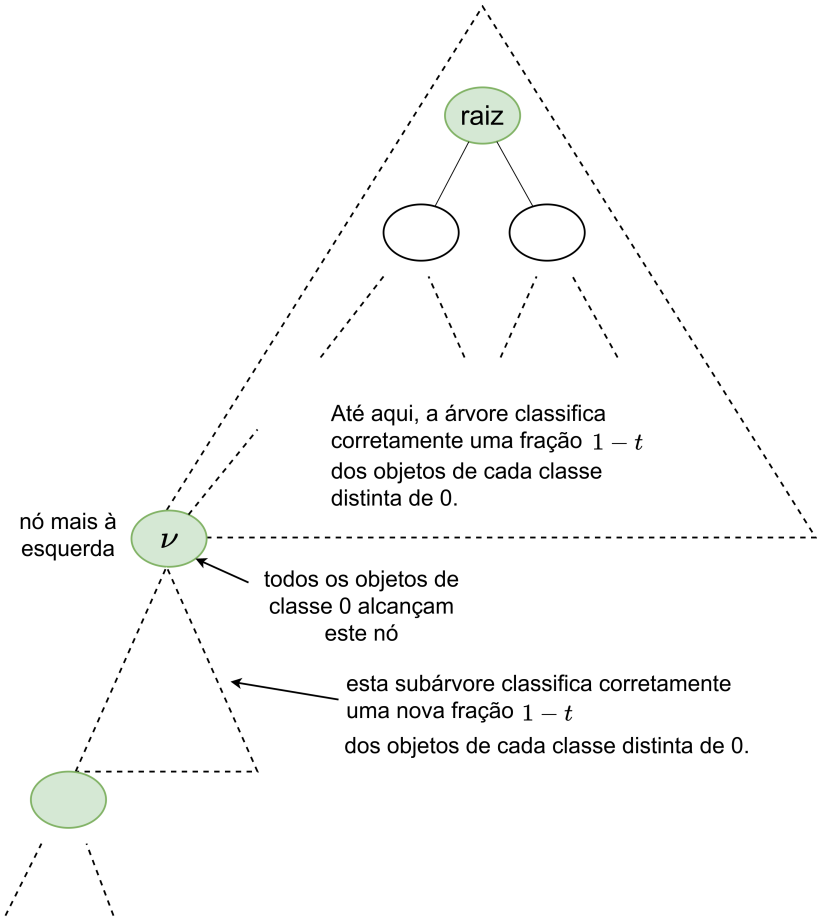


Figura 3.3: Construção de uma árvore binária ótima de uma instância criada para provar o Teorema 2.

## 3.3 Um Algoritmo Prático

Apresentaremos a seguir um algoritmo, intitulado `SER-DT` (*Short Explainable Rules for Decision Trees*), que gera árvores com aproximações ótimas em relação à altura e *explanation size* no pior caso e no caso médio. Além disso, o método inclui um hiperparâmetro que possibilita ajustar explicabilidade e acurácia, como observado na Seção 4.2.

PUC-Rio - Certificação Digital Nº 2112508/CA

Para enunciar o algoritmo, primeiramente teremos de definir alguns conceitos.

**Definição 3.5** ***Partições de um nó sobre um atributo.***
*Seja um conjunto de objetos $S \subseteq O$, um atributo $a$ e um valor $i \in \mathbb{R}$. Definimos:*

- $S(a, i) := \{o \in S \mid a(o) = i\}$
- $S(a, \leq i) := \{o \in S \mid a(o) \leq i\}$
- $S(a, > i) := \{o \in S \mid a(o) > i\}$

**Definição 3.6** ***Par de objetos mal classificado***
*Chamamos o par de objetos $(o, o')$ como **mal classificado** se $c(o) \neq c(o')$. O nome "mal classificado" se deve ao fato de que, se ambos os objetos $o$ e $o'$ alcançarem a mesma folha, pelo menos um deles será classificado erroneamente.*

**Definição 3.7** ***Medida de má classificação de pares ponderada***
*Seja um conjunto de objetos $S \subseteq O$. Sendo $w(S) := \sum_{o \in S} w(o)$ e $P(S)$ o número de pares mal classificados em $S$, definimos a medida de má classificação de pares ponderada (weighted pair-wise misclassification) de $S$ como:*

$$\texttt{wpm}(S) = P(S) \cdot w(S)$$

Podemos agora descrever o algoritmo `SER-DT`. Como outros algoritmos da área, nosso algoritmo, em cada passo, escolhe uma divisão dos nós que cria uma subárvore de "impureza pequena" (no nosso caso, utilizando o índice Gini, como exposto na Seção 4.2). Entretanto, ao contrário da maioria dos algoritmos, não é completamente guloso e permite uma flexibilidade na escolha das divisões.

Como pré-processamento, cada peso $w(o)$ menor do que $w(O)/n^3$ é substituído por $w(O)/(w_{min}n^3)$, onde $w_{min}$ é o menor peso dentre os objetos de $O$. Essa ideia, advinda de [24], é importante para garantir uma dependência logarítmica em $n$, em vez de $w(O)$. Após esse pré-processamento, `SER-DT` é chamado para o conjunto de objetos O.

O Algoritmo 2 apresenta um pseudocódigo para `SER-DT`. Primeiramente, o algoritmo procura qualquer **teste balanceado**, isto é, um teste que reduza a medida `wpm` do conjunto de objetos atuais na metade, pelo menos. Se não existir um teste balanceado, então o algoritmo procura um atributo $a^*$ e um teste $t^*$

PUC-Rio - Certificação Digital Nº 2112508/CA

tais que, nas divisões $S(a^*, < t)$, $S(a^*, t)$ e $S(a^*, > t)$, somente a divisão do meio ($S(a^*, t)$) possua `wpm` maior que o valor desejado de $\frac{1}{2}\texttt{wpm}(S)$. Essa divisão em três é obtida utilizando-se de duas divisões binárias. Então, o algoritmo procede recursivamente em cada um desses três nós.

Um questionamento é se algum progresso é feito na recursão no nó $S(a^*, t^*)$, quando $\texttt{wpm}(S(a^*, t^*)) > \frac{1}{2}\texttt{wpm}(S)$. Na verdade, a escolha do atributo $a^*$ é feita de modo que o conjunto $S(a^*, t^*)$ possua um `wpm` mínimo dentre todas as possíveis tripartições variando-se os atributos. Desse modo, podemos empregar um limite inferior (presente em [6, Lema 8]), que resulta na garantia logarítmica.

**Algoritmo 2:** `SER-DT`

**Input:** Um nó $\nu$ alcançado por um conjunto $S$ de objetos.

**if** *todos os objetos em S são da mesma classe* **then**
  crie uma folha correspondente a tal classe;
  **return**
**end**
**if** *existe um teste* $\tau$ *que divide* $S$ *em* $S^L$ *and* $S^R$ *tais que* $\max\{\texttt{wpm}(S^L), \texttt{wpm}(S^R)\} \leq \frac{1}{2}\texttt{wpm}(S)$ **then**
  Use um teste qualquer $\tau = (a, t)$ como raiz da árvore;
  Execute `SER-DT` recursivamente nos nós filhos $S(a, \leq t)$ e $S(a, > t)$;
**else**
  Seja $a^*$ um atributo em $\underset{a}{\operatorname{argmin}}\{\max_i \texttt{wpm}(S(a, i))\}$;
  Seja $t^*$ o menor valor de um atributo $a^*$ tal que o "filho à esquerda" $S(a^*, \leq t^*)$ satisfaça

$$\texttt{wpm}(S(a^*, \leq t^*)) \geq \frac{1}{2} \cdot \texttt{wpm}(S)$$

  Utilize dois testes binários para simular uma divisão em três: $S(a^*, < t^*)$, $S(a^*, t^*)$, $S(a^*, > t^*)$. Mais precisamente, na raiz da árvore use um teste no atributo $a^*$ que divide $S$ nos conjuntos $S(a^*, \leq t^*)$ e $S(a^*, > t^*)$. Então, aplique um teste no filho à esquerda da raiz, associado com $S(a^*, \leq t^*)$, criando-se assim dois novos nós filhos com objetos $S(a^*, < t^*)$ e $S(a^*, t^*)$;
  Execute `SER-DT` nas três folhas da árvore atual;
**end**

O próximo teorema fornece as garantias teóricas para altura e *explanation size* no caso médio e pior caso para as árvores gerados por `SER-DT`.

PUC-Rio - Certificação Digital Nº 2112508/CA

**Teorema 3** *Dada uma instância $I$, o algoritmo* `SER-DT` *produz uma árvore $T$ que satisfaz:*

1. *(Caso médio)*
   - $\text{depth}_{\text{avg}}(T) \leq O(\log n)\ \text{depth}^*_{\text{avg}}$
   - $\text{expl}_{\text{avg}}(T) \leq O(\log n)\ \text{expl}^*_{\text{avg}}$
2. *(Pior caso)*
   - $\text{depth}_{\text{wc}}(T) \leq O(\log n)\ \text{depth}^*_{\text{wc}}$
   - $\text{expl}_{\text{wc}}(T) \leq O(\log n)\ \text{expl}^*_{\text{wc}}$

**Prova**: *vide* [6, Anexo D].

A seguir, resumiremos a prova do teorema acima no caso médio. Consideramos $\gamma \in [\frac{1}{2}, 1)$ como o fator existente na linha 5 do Algoritmo 2 (no pseudocódigo, é adotado $\gamma = \frac{1}{2}$). A ideia da prova envolve mostrar que para uma constate $\alpha = \max\{4, 2(\ln \frac{1}{\gamma})^{-1}\}$:

$$\text{depth}_{\text{avg}}(T) \leq \alpha \cdot \ln(P(O)w(o)) \cdot \text{depth}_{\text{avg}}(M^*), \tag{3-1}$$

onde $M^*$ é uma árvore multivia de altura $\text{depth}^*_{\text{avg}}$, ótima no caso médio para a instância $I$. A prova da Equação 3-1 se dá, grosso modo, através de indução no número de pares mal classificados $P(O)$ da instância.

Já que $\ln(P(O)w(O)) \leq \ln(n^2W) = O(\ln(nW))$, e sabendo-se da equivalência entre árvores ótimas multivia e binárias [1], a equação 3-1 resulta em:

$$\text{expl}_{\text{avg}}(T) \leq \text{depth}_{\text{avg}}(T) \leq O(\ln(nW)) \cdot \text{depth}^*_{\text{avg}} = O(\ln(nW)) \cdot \text{expl}^*_{\text{avg}}$$

Daí, temos que:

$$\text{depth}_{\text{avg}}(T) \leq O\left(\ln\left(n\frac{W}{w_{min}}\right)\right)\text{depth}^*_{\text{avg}}$$
$$\text{expl}_{\text{avg}}(T) \leq O\left(\ln\left(n\frac{W}{w_{min}}\right)\right)\text{expl}^*_{\text{avg}}$$

Finalmente, pelo passo de pré-processamento citado anteriormente, advindo de [24], temos que o fator de aproximação $O\left(\ln\left(n\frac{W}{w_{min}}\right)\right)$ se torna $O(\log n)$, isto é, podemos remover a dependência nos pesos e alcançar a aproximação logarítmica desejada, como enunciado no teorema.

[1] *Vide* a intuição da prova do Teorema 1. A equivalência entre árvores ótimas multivia e binária em relação ao *explanation size* e altura também é válida no caso médio, como exposto em [6, Lema 3, pág. 18].

PUC-Rio - Certificação Digital Nº 2112508/CA

Observamos que, segundo [4] e [5], `SER-DT` garante a melhor aproximação obtida por um algoritmo polinomial em relação à altura e *explanation size*, seja no pior caso ou no caso médio.

PUC-Rio - Certificação Digital Nº 2112508/CA

# 4
# Experimentos

Realizamos uma série de experimentos computacionais comparativos entre `SER-DT` e algoritmos conhecidos para Árvores de Decisão. Além disso, avaliamos os algoritmos através da alteração de hiperparâmetros, medidas de impureza e de da implementação de um algoritmo de pós-poda. As seções a seguir discorrem sobre os experimentos realizados.

## 4.1
## *Datasets* e Configurações

Foram utilizados 20 *datasets* descritos na Tabela 4.1. Como pré-processamento, atributos categóricos foram convertidos em binários através de *one-hot-encoding*. Isto foi necessário porque, segundo o nosso modelo, os testes contidos nas árvores são da forma "$a(o) \leq t$?", conforme mencionado na Seção 3.1.

Foi reservada uma fração de 70% do número total de exemplos (para cada dataset) para treino e os 30% restantes para teste. Todos os objetos foram considerados igualmente importantes, isto é, o mesmo peso de 1 / (total de exemplos no conjunto de treino) foi atribuído para cada objeto.

Os experimentos foram realizados no seguinte ambiente:

- Notebook Inspiron 14-7460;
- Processador: Intel 7.ª Geração Core i5;
- 16 GB RAM (DDR4 at 2400 MHz);
- Tipo de Armazenamento SSD;
- GPU NVIDIA GeForce 940MX.

As bibliotecas necessárias para a execução foram: Python 3.8.10; pandas 1.4.2; numpy 1.22.3; sklearn 1.0.2.

O código completo dos experimentos está disponível em: `https://github.com/victorfcsouza/explainable-dts`.

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.1: *Datasets utilizados nos experimentos.* `n` é o número de exemplos; `num.` é o número de atributos numéricos; `categ.` é o número de atributos categóricos, `d` é o número total de atributos (depois de realizar *one-hot-encoding*) e `classes` é o número de classes de saída.

| dataset | n | num. | categ. | *d* | classes | referência |
|---|---|---|---|---|---|---|
| Anuran | 7195 | 22 | 0 | 22 | 4 | [25] |
| Audit Risk | 773 | 26 | 0 | 26 | 2 | [26] |
| Avila | 20867 | 10 | 0 | 10 | 12 | [27] |
| Banknote | 1372 | 4 | 0 | 4 | 2 | [28] |
| Bankruptcy Polish | 4885 | 64 | 0 | 64 | 2 | [29] |
| Cardiotocography | 2126 | 21 | 0 | 21 | 10 | [30] |
| Collins | 1000 | 19 | 0 | 19 | 30 | [31] |
| Defaults Credit Card | 30000 | 20 | 3 | 33 | 2 | [32] |
| Dry Bean | 13611 | 16 | 0 | 16 | 7 | [33] |
| EEG Eye State | 14980 | 14 | 0 | 14 | 2 | [34] |
| HTRU2 | 17898 | 8 | 0 | 8 | 2 | [35] |
| Iris | 150 | 4 | 0 | 4 | 3 | [36] |
| Letter Recognition | 20000 | 16 | 0 | 16 | 26 | [37] |
| Mice | 552 | 77 | 3 | 83 | 8 | [38] |
| OBS Network | 1060 | 19 | 2 | 24 | 4 | [39] |
| Occupancy Room | 10129 | 16 | 0 | 16 | 4 | [40] |
| Online Shoppers Intention | 12330 | 12 | 5 | 54 | 2 | [41] |
| Pen Digits | 10992 | 16 | 0 | 16 | 10 | [42] |
| Poker Hand | 1025010 | 10 | 0 | 10 | 10 | [43] |
| Sensorless | 58509 | 48 | 0 | 48 | 11 | [44] |

## 4.2 Implementação

Note que na linha 5 do Algoritmo 2 (caso em que existe um *split* balanceado), qualquer *split* $\tau$ que divida um nó em dois subconjuntos com `wpm` pequenos o suficiente pode ser usado para prosseguir com a recursão. Escolheremos *splits* que possibilitem obter uma árvore com *explanation size* pequeno e boa acurácia, na prática. Para tal, considere a escolha da medida de impureza Gini utilizada pelo `CART`. Ela é definida por:

$$\texttt{Gini}(S) = 1 - \sum_{i=1}^{c} \left( \frac{|S_i|}{|S|} \right)^2,$$

onde $S$ é o conjunto de objetos que alcançam um nó $\nu$ da árvore, $c$ é o número de classes rotuladas possíveis dentre os elementos de $S$ e $S_i$ é o conjunto de objetos cuja classe é $i$. A medida `Gini` ponderada, $\texttt{Gini}(\tau, \nu)$, induzida por um teste $\tau$ que divide o conjunto de exemplos $S$ que alcançam um nó $\nu$ nos subconjuntos $S^L$ e $S^R$ é definida por:

PUC-Rio - Certificação Digital Nº 2112508/CA

$$\texttt{Gini}(\tau, \nu) = \frac{|S^L|}{|S|}\texttt{Gini}(S^L) + \frac{|S^R|}{|S|}\texttt{Gini}(S^R).$$

Definimos `FactorExpl` como um hiperparâmetro no intervalo $[0, 1]$. Em vez do `Gini` padrão, o algoritmo `SER-DT` escolherá o split $\tau$ que minimiza a medida de **Gini modificado** definida por:

$$\texttt{AdjustedGiniExpl}(\tau, \nu) := I(\tau, \nu) \times \texttt{Gini}(\tau, \nu),$$

onde $I(\tau, \nu) = \texttt{FactorExpl}$ se o atributo associado ao teste $\tau$ já tiver aparecido no caminho da raiz até $\nu$, e $I(\tau, \nu) = 1$ caso contrário. Intuitivamente, um `FactorExpl` pequeno favorece a escolha de atributos repetidos no caminho da raiz até um nó, o que leva a árvores com *explanation size* menores. Em contrapartida, como o efeito do `Gini` é reduzido, espera-se que tais árvores possuam acurácia menor quando comparadas ao `CART`.

Em relação ao critério de parada, o algoritmo `SER-DT` não expande um nó $\nu$ se ele possuir altura máxima previamente definida, ou se não existir split $\tau$ que possua $\texttt{Gini}(\tau, \nu)$ menor que o Gini para o nó $\nu$. Essa última condição foi utilizada para permitir uma comparação justa de altura com a implementação do `CART`, que utiliza o mesmo critério. Para o caso de não existir *split* balanceado, o algoritmo `SER-DT` segue a linha 8 do pseudo-código, em que `FactorExpl` não é considerado. Como esses critérios de pré-pruning são empregados, a implementação de `SER-DT` não necessariamente gera árvores de classificação exata (a acurácia no conjunto de treino pode diferir de 1).

Para simplificação das árvores, após a execução dos algoritmos, um procedimento de pós-poda é utilizado para unir folhas irmãs que possuam a mesma classe assinalada, como detalhado no Algoritmo 3. Neste procedimento, a cada nó é assinalado um inteiro indicando a classe dos objetos que o alcançam (caso sejam todos da mesma classe), ou $-1$ (caso os objetos sejam de classes distintas). A poda é realizada na subárvore gerada para cada nó que possua um valor assinalado diferente de $-1$. Note que este algoritmo de pós-poda não altera a classificação dos objetos; somente simplifica a árvore. Outro algoritmo de pós-poda é apresentado na Seção 4.10.

PUC-Rio - Certificação Digital Nº 2112508/CA

**Algoritmo 3:** Simple-Post-Pruning

**Input:** Um nó $\alpha$
**if** `Get-Class`$(\alpha) \neq -1$ **then**
  Remova a subárvore gerada pelos filhos de $\alpha$;
**end**
$\alpha_l \leftarrow$ filho à esquerda de $\alpha$;
$\alpha_r \leftarrow$ filho à direita de $\alpha$;
`Simple-Post-Pruning`$(\alpha_l)$;
`Simple-Post-Pruning`$(\alpha_r)$;
**Procedimento** `Get-Class`
  **Input:** Um nó $\nu$
  **if** $\nu$ *é uma folha* **then**
    **return** $c(\nu)$, a classe representativa de $\nu$;
  **end**
  $\nu_l \leftarrow$ filho à esquerda de $\nu$;
  $\nu_r \leftarrow$ filho à direita de $\nu$;
  **if** `Get-Class`$(\nu_l)$ = `Get-Class`$(\nu_r) = c_{ch}$ **then**
    **return** $c_{ch}$;
  **end**
  **else**
    **return** $-1$;
  **end**

## 4.3 Impacto de `FactorExpl`

O primeiro experimento consistiu na averiguação do impacto de `FactorExpl` na acurácia e nas medidas de explicabilidade. A Figura 4.1 apresenta as médias de acurácia no conjunto de teste e $\text{expl}_{\text{avg}}$ calculadas sobre os 20 datasets listadas na Tabela 4.1 em função de `FactorExpl`. Mais especificamente, para cada

$$\texttt{FactorExpl} \in \{0.1 \times i \mid i \in \{1, \ldots, 9\}\} \cup \{0.9 + 0.01 \times i \mid i \in \{1, \ldots, 10\}\}$$

foram calculadas as médias de acurácia no conjunto de teste e $\text{expl}_{\text{avg}}$ referentes às árvores geradas por `SER-DT` sobre os 20 *datasets* por meio de 10 iterações. Em cada iteração alterou-se a semente associada à divisão aleatória entre os exemplos de treino e teste.

Como esperado, nota-se um aumento das métricas de acurácia e $\text{expl}_{\text{avg}}$ ao se aumentar `FactorExpl`. O interessante, porém, é que para valores de `FactorExpl` próximos a 1 há um pequeno aumento na acurácia de teste

PUC-Rio - Certificação Digital Nº 2112508/CA

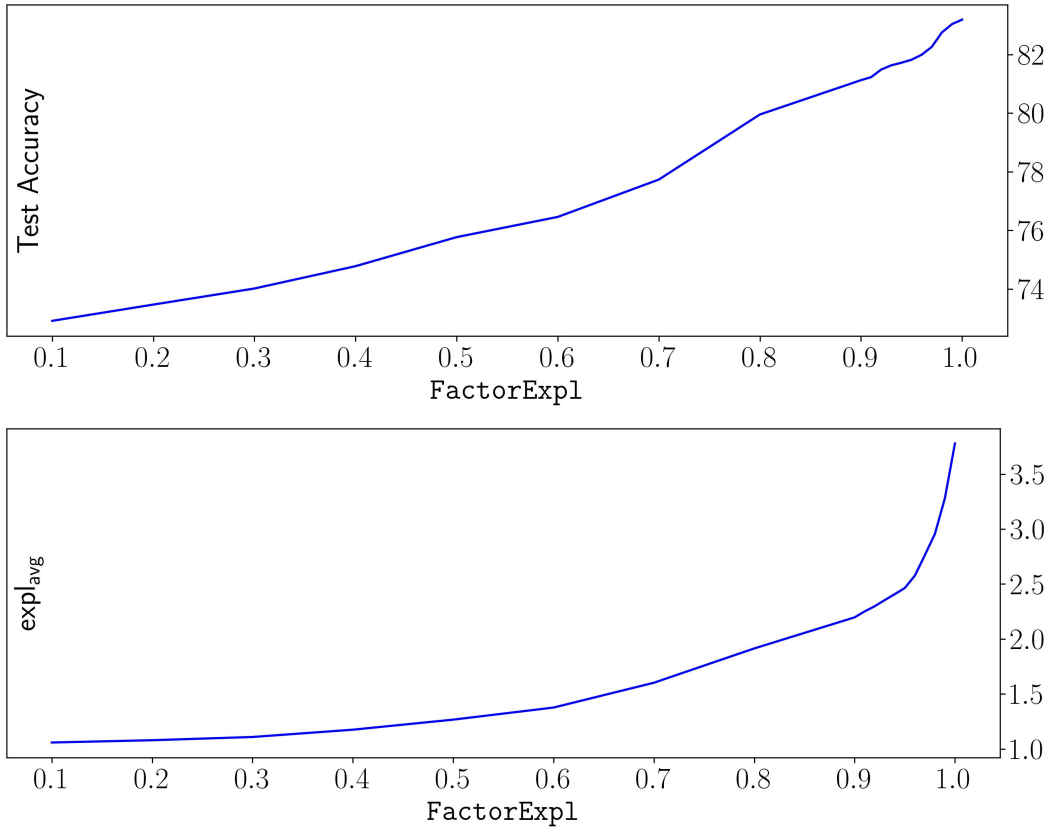


Figura 4.1: Acurácia no conjunto de teste e $\text{expl}_{\text{avg}}$ calculados em 20 datasets em função de `FactorExpl`.

quando comparado com o aumento de $\text{expl}_{\text{avg}}$. Isso sugere que a escolha de valores de `FactorExpl` próximas a 1 pode gerar árvores significativamente mais explicáveis, mas com pouca perda de acurácia.

## 4.4 Comparação com `CART`

Na seguinte série de experimentos comparamos `CART` com `SER-DT` fixando-se `FactorExpl` $= 0.97$. Esse valor foi escolhido por ser próximo de 1, o que poderia levar a resultados interessantes segundo a seção anterior. Para resultados utilizando outros valores de `FactorExpl` no intervalo $[0.90, 1]$, *vide* a Seção 4.5.

Utilizamos como critério de parada uma altura máxima de 6, em consonância com a necessidade de se limitar previamente a altura máxima, como exposto na Seção 4.2. O valor de 6 foi escolhido arbitrariamente por produzir árvores não muito complexas, o que é preferível para análise de explicabilidade. Para árvores produzidas sem limites de altura, *vide* a Seção 4.6.

A Tabela 4.2 apresenta os resultados de acurácia no conjunto de teste, *explanation size* no caso médio ($\text{expl}_{\text{avg}}$) e *explanation size* no pior caso ($\text{expl}_{\text{wc}}$)

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.2: Acurácia de teste, $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$ para `FactorExpl` = 0.97. Cada entrada é a média de 10 execuções utilizando diferentes sementes para definir aleatoriamente os exemplos de treino e teste. Valores em negrito indicam uma diferença de mais de 1% (colunas de posição 2 e 3) ou um ganho de mais de 25% em favor de `SER-DT` comparado a `CART` (colunas 4,5,6 e 7).

| *Dataset* | Acurácia de teste | | $\text{expl}_{\text{avg}}$ | | $\text{expl}_{\text{wc}}$ | |
|---|---|---|---|---|---|---|
| | `SER-DT` | `CART` | `SER-DT` | `CART` | `SER-DT` | `CART` |
| Anuran | 94.8% | 94.7% | 4.78 | 5.24 | 6.0 | 6.0 |
| Audit Risk | 99.9% | 99.9% | 1.00 | 1.00 | 1.0 | 1.0 |
| Avila | 61.5% | **63.2%** | **3.06** | 4.22 | 4.9 | 5.4 |
| Banknote | 97.6% | 98.1% | 2.44 | 2.55 | 3.8 | 3.4 |
| Bankruptcy Polish | 96.6% | 96.9% | **2.56** | 4.63 | 5.6 | 5.9 |
| Cardiotocography | 89.5% | 89.8% | 4.30 | 5.30 | 5.9 | 6.0 |
| Collins | 13.2% | **15.6%** | **2.13** | 4.76 | **4.4** | 5.9 |
| Default Credit Card | 82.0% | 81.9% | **1.45** | 4.29 | **4.5** | 6.0 |
| Dry Bean | 90.1% | 89.8% | **3.32** | 4.45 | 5.1 | 6.0 |
| EEG Eye State | 74.1% | 73.6% | 3.69 | 4.29 | 5.9 | 6.0 |
| HTRU2 | 97.7% | 97.7% | **1.20** | 2.03 | 4.3 | 4.9 |
| Iris | 94.2% | 93.6% | 1.75 | 1.76 | 3.1 | 3.4 |
| Letter Recognition | 44.9% | **47.9%** | **3.34** | 5.50 | 5.5 | 6.0 |
| Mice | 99.9% | 99.9% | 3.05 | 3.05 | 3.6 | 3.6 |
| OBS Network | **91.7%** | 89.5% | 3.48 | 4.26 | 5.3 | 5.9 |
| Occupancy Room | 99.4% | 99.3% | 4.18 | 4.54 | 5.3 | 5.7 |
| Online Shoppers Intention | 89.3% | 89.8% | 3.30 | 4.00 | 5.1 | 6.0 |
| Pen Digits | **88.6%** | 86.9% | 4.76 | 5.31 | 5.8 | 6.0 |
| Poker Hand | 52.9% | **55.0%** | **1.80** | 4.30 | **3.8** | 5.1 |
| Sensorless | **87.4%** | 80.1% | **2.94** | 4.03 | 4.9 | 5.5 |
| Média | 82.3% | 82.2% | 2.93 | 3.97 | 4.69 | 5.19 |

para todos os *datasets*. Cada entrada na tabela é uma média de 10 execuções alterando-se a semente associada à divisão aleatória entre os exemplos de treino e teste.

Notamos que a acurácia obtida pelo nosso algoritmo é muito próximo à do `CART`, enquanto o ganho nas métricas de explicabilidade é significativo. Em 7 datasets (assinalados em negrito nas colunas de posição 2 e 3) observamos uma diferença maior que 1% nas acurácias; em 3 deles, nosso algoritmo é melhor que `CART`, enquanto nos outros 4, `CART` é melhor. Considerando $\text{expl}_{\text{avg}}$, nosso algoritmo é tão bom quanto `CART` para todos os datasets, e para 9 deles (em negrito na coluna 4) há um ganho de pelo menos 25%. Para $\text{expl}_{\text{wc}}$ o ganho também é significativo: para todos os datasets, exceto *Banknote*, nosso algoritmo é tão bom quanto `CART`. Além disso, para 3 deles (em negrito na coluna 6), há uma melhoria de pelo menos 25%. Apresentamos no apêndice A os diagramas de caixa relacionados aos dados desta tabela, para uma visão mais completa.

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.3: $\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$ para `FactorExpl` = 0.97. Cada entrada é a média de 10 execuções utilizando diferentes sementes para definir aleatoriamente os exemplos de treino e teste. Valores em negrito indicam um ganho de mais de 25% em favor de `SER-DT` quando comparado ao `CART`.

| *Dataset* | $\text{depth}_{\text{avg}}$ | | $\text{depth}_{\text{wc}}$ | |
|---|---|---|---|---|
| | `SER-DT` | `CART` | `SER-DT` | `CART` |
| Anuran | 5.38 | 5.57 | 6.00 | 6.00 |
| Audit Risk | 1.00 | 1.00 | 1.00 | 1.00 |
| Avila | 5.14 | 5.29 | 6.00 | 6.00 |
| Banknote | 4.39 | 4.43 | 6.00 | 6.00 |
| Bankruptcy Polish | 4.45 | 4.94 | 6.00 | 6.00 |
| Cardiotocography | 4.98 | 5.51 | 6.00 | 6.00 |
| Collins | 5.89 | 5.91 | 6.00 | 6.00 |
| Default Credit Card | **2.15** | 4.33 | 6.00 | 6.00 |
| Dry Bean | 4.76 | 5.41 | 6.00 | 6.00 |
| EEG Eye State | 5.15 | 5.47 | 6.00 | 6.00 |
| HTRU2 | **2.80** | 5.30 | 6.00 | 6.00 |
| Iris | 2.50 | 2.52 | 4.90 | 4.80 |
| Letter Recognition | 5.96 | 5.94 | 6.00 | 6.00 |
| Mice | 3.05 | 3.05 | 3.60 | 3.60 |
| OBS Network | 4.47 | 4.39 | 6.00 | 6.00 |
| Occupancy Room | 4.72 | 4.83 | 6.00 | 6.00 |
| Online Shoppers Intention | 3.89 | 4.77 | 6.00 | 6.00 |
| Pen Digits | 5.73 | 5.80 | 6.00 | 6.00 |
| Poker Hand | 4.61 | 4.30 | 6.00 | 5.30 |
| Sensorless | 5.26 | 5.33 | 6.00 | 6.00 |
| Média | 4.31 | 4.70 | 5.58 | 5.54 |

A Tabela 4.3 apresenta os valores correspondentes de $\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$ para o `FactorExpl` escolhido de 0.97. O valor de $\text{depth}_{\text{avg}}$ para nosso algoritmo é melhor ou igual que o de `CART` para 17 e pior somente para 3 *datasets*. Além disso, para 2 *datasets* nosso algoritmo é melhor em pelo menos 25%. Considerando $\text{depth}_{\text{wc}}$, observa-se um comportamento similar entre os dois algoritmos. Isso é esperado já que limitamos a árvore em uma altura fixa.

A Tabela 4.4 apresenta os valores dos tempos de execução necessários para realizar os experimentos das Tabelas 4.2 e 4.3 em conjunto. A razão entre os tempos de execução é nó máximo 2 para todos os datasets e, na média, `SER-DT` possui tempo menor do que `CART`.

## 4.5 Sensibilidade a `FactorExpl`

Lembre-se que a Figura 4.1 sugere ser possível obter um balanço entre acurácia e métricas de explicabilidade através da escolha de `FactorExpl`.

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.4: Tempos de execução para `SER-DT` (`FactorExpl` = 0.97) e `CART`. Cada medida é uma média de tempo em segundos para 10 iterações.

| *Dataset* | `SER-DT` (sec) | `CART` (sec) |
|---|---|---|
| Anuran | 7.0 | 3.7 |
| Audit Risk | 0.1 | 0.1 |
| Avila | 9.0 | 9.2 |
| Banknote | 0.2 | 0.1 |
| Bankruptcy Polish | 9.8 | 5.2 |
| Cardiotocography | 1.4 | 1.7 |
| Collins | 2.2 | 1.7 |
| Default Credit Card | 18.1 | 19.3 |
| Dry Bean | 12.1 | 7.1 |
| EEG Eye State | 3.5 | 4.0 |
| HTRU2 | 5.0 | 3.0 |
| Iris | 0.0 | 0.0 |
| Letter Recognition | 17.1 | 26.1 |
| Mice | 1.7 | 1.0 |
| OBS Network | 0.5 | 0.5 |
| Occupancy Room | 2.7 | 3.7 |
| Online Shoppers Intention | 9.1 | 11.7 |
| Pen Digits | 5.4 | 7.1 |
| Poker Hand | 330.8 | 443.6 |
| Sensorless | 178.5 | 120.5 |
| Média | 28.8 | 30.1 |

A Seção 4.4 corrobora ser possível obter árvores significativamente mais explicáveis, mas com pouca perda de acurácia, escolhendo-se `FactorExpl` = 0.97, quando comparamos `SER-DT` com `CART`. Com o intuito de fornecer maiores evidências, realizamos experimentos com `FactorExpl` $\in \{0.9, 0.95, 0.99\}$. Os dados obtidos de acurácia e $\text{expl}_{\text{avg}}$ são mostrados na Tabela 4.5.

Notamos que $\text{expl}_{\text{avg}}$ tem um comportamento esperado: quanto menor `FactorExpl`, menor $\text{expl}_{\text{avg}}$ (últimas 3 colunas da tabela). Considerando acurácia, para 13 datasets a diferença entre `FactorExpl` = 0.99 e `FactorExpl` = 0.9 é menor que 1%. Esse número aumenta para 16 quando comparamos `FactorExpl` = 0.99 e `FactorExpl` = 0.95. Nesse sentido, não recomendamos utilizar valores menores que 0.9 para `FactorExpl`, na prática, porque a perda em acurácia pode ser acentuada. Nossa recomendação é utilizar `SER-DT` com `FactorExpl` $\in [0.95, 0.99]$.

## 4.6 Removendo a Restrição de Altura Máxima

Nos experimentos anteriores, limitamos a altura máxima das árvores em 6, para evitarmos a geração de árvores muito grandes. As Tabelas 4.6 e 4.7 apresentam os resultados quando implementamos `SER-DT` sem a restrição de

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.5: Dados de acurácia e $\text{expl}_{\text{avg}}$ para `SER-DT` em função de `FactorExpl` (`FE`).

| *Dataset* | Acurácia de teste | | | $\text{expl}_{\text{avg}}$ | | |
|---|---|---|---|---|---|---|
| | `FE`= 0.99 | `FE`=0.95 | `FE`=0.90 | `FE`=0.99 | `FE`=0.95 | `FE`=0.90 |
| Anuran | 94.8% | 94.8% | 94.6% | 5.11 | 4.50 | 3.85 |
| Audit Risk | 99.9% | 99.9% | 99.9% | 1.00 | 1.00 | 1.00 |
| Avila | 66.0% | 57.8% | 55.9% | 4.22 | 1.99 | 1.64 |
| Banknote | 97.8% | 97.6% | 97.5% | 2.51 | 2.38 | 2.36 |
| Bankruptcy Polish | 96.7% | 96.6% | 96.7% | 3.73 | 1.92 | 1.50 |
| Cardiotocography | 89.7% | 89.3% | 89.5% | 4.51 | 4.10 | 3.98 |
| Collins | 15.9% | 13.1% | 12.8% | 3.74 | 1.61 | 1.35 |
| Default Credit Card | 81.9% | 82.0% | 82.0% | 1.94 | 1.41 | 1.40 |
| Dry Bean | 89.7% | 90.0% | 89.8% | 3.60 | 3.25 | 3.05 |
| EEG Eye State | 74.6% | 72.9% | 67.1% | 4.32 | 3.15 | 1.82 |
| HTRU2 | 97.7% | 97.7% | 97.6% | 1.27 | 1.19 | 1.15 |
| Iris | 94.2% | 94.2% | 94.2% | 1.75 | 1.75 | 1.75 |
| Letter Recognition | 51.9% | 42.4% | 40.1% | 4.20 | 2.90 | 2.59 |
| Mice | 99.9% | 99.9% | 99.9% | 3.05 | 3.05 | 3.05 |
| OBS Network | 91.7% | 91.3% | 90.3% | 3.75 | 3.15 | 2.67 |
| Occupancy Room | 99.3% | 99.4% | 99.5% | 4.25 | 4.15 | 4.07 |
| Online Shoppers Intention | 89.9% | 89.1% | 88.7% | 3.82 | 3.12 | 2.94 |
| Pen Digits | 88.5% | 88.6% | 87.7% | 4.90 | 4.75 | 4.42 |
| Poker Hand | 53.0% | 52.8% | 52.7% | 1.89 | 1.79 | 1.76 |
| Sensorless | 87.5% | 87.0% | 85.9% | 3.01 | 2.85 | 2.67 |
| Média | 83.0% | 81.8% | 81.1% | 3.33 | 2.70 | 2.45 |

altura das árvores. O único critério de parada utilizado foi a não existência de um split $\tau$ que possua $\texttt{Gini}(\tau, \nu)$ menor que o `Gini` para o nó $\tau$ (*vide* parágrafo sobre critério de parada na Seção 4.2). Para fins de comparação, alteramos a implementação de `CART` para possuir esse mesmo critério de parada.

Algumas observações:

- `SER-DT` produz árvores com altura muito menor do que aquelas construídas por `CART`. Essa diferença não era possível de se observar quando havia o limite de altura máxima.
- Os ganhos de `SER-DT` sobre `CART` considerando todas as métricas de explicabilidade tornam-se evidentes quando a restrição de limite de altura é removida.

## 4.7 Entropia como Critério de Divisão

Nos experimentos da Seção 4.4 comparamos `SER-DT` com `CART`, que utiliza `Gini` como critério de avaliação de uma divisão de nós e como critério de parada. Outra possibilidade é utilizar a medida de `Entropia` para o mesmo fim.

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.6: Acurácia no conjunto de teste, $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$ quando não há restrição de altura máxima. Os valores em negrito representam as diferenças de acurácia maiores que 1% e os casos em que o ganho nas métricas de explicabilidade ($\text{expl}_{\text{avg}}$ ou $\text{expl}_{\text{wc}}$) são maiores do que 30% quando comparados ao `CART`.

| *Dataset* | Acurácia de teste | | $\text{expl}_{\text{avg}}$ | | $\text{expl}_{\text{wc}}$ | |
|---|---|---|---|---|---|---|
| | `SER-DT` | `CART` | `SER-DT` | `CART` | `SER-DT` | `CART` |
| Anuran | 95.7% | 95.6% | **6.00** | 8.74 | 9.4 | 11.1 |
| Audit Risk | 99.9% | 99.9% | 1.00 | 1.00 | 1 | 1 |
| Avila | 92.8% | **98.4%** | 5.47 | 5.81 | 9 | 9.6 |
| Banknote | 97.8% | 98.0% | 2.46 | 2.56 | 3.8 | 3.8 |
| Bankruptcy Polish | 95.9% | 95.8% | **3.14** | 7.25 | **8** | 13 |
| Cardiotocography | 87.6% | 88.0% | **5.73** | 8.54 | **8.7** | 14.6 |
| Collins | 12.0% | **14.5%** | **4.30** | 7.81 | **7.3** | 12.7 |
| Default Credit Card | 72.7% | 72.5% | **5.74** | 13.08 | **11.1** | 20.2 |
| Dry Bean | 89.5% | 89.3% | **4.60** | 7.41 | 8.6 | 11.4 |
| EEG Eye State | 82.2% | 82.9% | 6.48 | 8.31 | 11.1 | 13 |
| HTRU2 | 96.7% | 96.7% | **1.74** | 4.24 | 6.2 | 7.3 |
| Iris | 94.2% | 93.6% | 1.75 | 1.76 | 3.1 | 3.4 |
| Letter Recognition | 84.3% | **86.2%** | 6.84 | 8.83 | 11.5 | 14.8 |
| Mice | 99.9% | 99.9% | 3.05 | 3.05 | 3.6 | 3.6 |
| OBS Network | 100.0% | 100.0% | 3.94 | 4.83 | 6.5 | 8.2 |
| Occupancy Room | 99.6% | 99.5% | 4.26 | 4.98 | 6.9 | 7.9 |
| Online Shoppers Intention | 86.2% | 86.3% | **5.72** | 9.15 | **10.1** | 16.4 |
| Pen Digits | 95.7% | 96.0% | 6.64 | 7.87 | 10.4 | 12.7 |
| Poker Hand | **81.0%** | 62.5% | **6.05** | 8.74 | 10 | 10 |
| Sensorless | 98.4% | 98.1% | **4.93** | 9.60 | **10.3** | 20.3 |
| Média | 88.1% | 87.7% | 4.49 | 6.68 | 7.83 | 10.75 |

Tabela 4.7: $\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$ quando não há restrição de altura máxima. Os valores em negrito assinalam os casos em que o ganho de $\text{depth}_{\text{avg}}$ ou $\text{depth}_{\text{wc}}$ são maiores que 20% quando comparado ao `CART`.

| *Dataset* | $\text{depth}_{\text{avg}}$ | | $\text{depth}_{\text{wc}}$ | |
|---|---|---|---|---|
| | `SER-DT` | `CART` | `SER-DT` | `CART` |
| Anuran | **7.69** | 10.79 | **11.2** | 15.1 |
| Audit Risk | 1.00 | 1.00 | 1.0 | 1.0 |
| Avila | 11.25 | 11.49 | 18.9 | 21.1 |
| Banknote | 4.51 | 4.60 | 6.9 | 7.3 |
| Bankruptcy Polish | **5.37** | 8.22 | **9.8** | 16.6 |
| Cardiotocography | **7.56** | 10.20 | **11.5** | 17.5 |
| Collins | 9.44 | 10.55 | **16.2** | 21.1 |
| Default Credit Card | **12.46** | 19.89 | **21.0** | 42.3 |
| Dry Bean | **9.20** | 12.16 | **15.8** | 25.0 |
| EEG Eye State | 10.81 | 13.38 | **18.1** | 25.2 |
| HTRU2 | **8.26** | 10.86 | **14.4** | 22.2 |
| Iris | 2.50 | 2.52 | 4.9 | 4.8 |
| Letter Recognition | 11.39 | 12.92 | 22.4 | 27.8 |
| Mice | 3.05 | 3.05 | 3.6 | 3.6 |
| OBS Network | 5.36 | 5.73 | 8.9 | 9.8 |
| Occupancy Room | 5.09 | 6.08 | 9.6 | 10.5 |
| Online Shoppers Intention | **9.01** | 11.56 | **15.8** | 24.2 |
| Pen Digits | 8.91 | 9.73 | 14.1 | 17.6 |
| Poker Hand | 17.92 | 20.75 | **30.5** | 40.7 |
| Sensorless | **9.64** | 13.66 | **17.6** | 31.6 |
| Média | 8.02 | 9.96 | 13.6 | 19.3 |

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.8: Acurácia no conjunto de teste, $\text{expl}_{\text{avg}}$ and $\text{expl}_{\text{wc}}$ quando entropia (Entr.) é utilizada em vez de `Gini`. Os valores em negrito representam diferenças em acurácia maiores que 1% e os casos em que as reduções de $\text{expl}_{\text{avg}}$ ou $\text{expl}_{\text{wc}}$ são maiores que 20%.

| *Dataset* | Acurácia de teste | | $\text{expl}_{\text{avg}}$ | | $\text{expl}_{\text{wc}}$ | |
|---|---|---|---|---|---|---|
| | SER-DT $_{Ent}$ | Entr. | SER-DT $_{Ent}$ | Entr. | SER-DT $_{Ent}$ | Entr. |
| Anuran | 94.5% | 94.5% | 4.92 | 5.06 | 6.0 | 6.0 |
| Audit Risk | 99.9% | 99.9% | 1.00 | 1.00 | 1.0 | 1.0 |
| Avila | **66.5%** | 65.3% | 3.54 | 4.18 | 5.0 | 5.1 |
| Banknote | 97.9% | 97.9% | 2.41 | 2.51 | 3.8 | 3.8 |
| Bankruptcy Polish | 97.1% | 97.3% | 2.95 | 3.17 | 5.6 | 6.0 |
| Cardiotocography | 89.1% | 89.6% | 4.36 | 4.69 | 6.0 | 6.0 |
| Collins | 14.1% | 14.2% | **3.51** | 4.81 | 5.0 | 5.7 |
| Default Credit Card | 82.0% | 81.9% | **1.41** | 3.82 | **4.2** | 6.0 |
| Dry Bean | 90.1% | 90.0% | 3.66 | 4.34 | 5.8 | 6.0 |
| EEG Eye State | **73.2%** | 72.1% | 3.42 | 4.26 | 5.6 | 6.0 |
| HTRU2 | 97.8% | 97.8% | **1.47** | 1.97 | 4.6 | 5.0 |
| Iris | 94.5% | 94.2% | 1.70 | 1.73 | 3.2 | 3.3 |
| Letter Recognition | 58.5% | **59.6%** | 4.13 | 5.01 | 6.0 | 6.0 |
| Mice | 99.9% | 99.9% | 3.00 | 3.00 | 3.0 | 3.0 |
| OBS Network | **89.4%** | 87.5% | 3.54 | 4.17 | 5.2 | 6.0 |
| Occupancy Room | 99.5% | 99.5% | 3.27 | 3.37 | 4.6 | 5.0 |
| Online Shoppers Intention | 89.1% | 89.9% | **2.63** | 3.47 | **4.2** | 5.9 |
| Pen Digits | 89.6% | 89.7% | 5.12 | 5.44 | 6.0 | 6.0 |
| Poker Hand | 53.0% | **55.7%** | **1.83** | 4.53 | **4.0** | 5.0 |
| Sensorless | 90.0% | 90.0% | **3.08** | 4.53 | 4.9 | 5.8 |
| Média | 83.3% | 83.3% | 3.05 | 3.71 | 4.68 | 5.13 |

Mais especificamente, definimos a entropia $\texttt{Entr}(\text{S})$ de um conjunto de objetos S como

$$\texttt{Entr}(S) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \cdot \log_2 \frac{|S_i|}{|S|},$$

onde $S$ é o conjunto de objetos que alcançam um nó $\nu$ da árvore, $c$ é o número de classes rotuladas possíveis dentre os elementos de $S$ e $S_i$ é o conjunto de objetos cuja classe é $i$.

A medida de entropia ponderada $\texttt{Entr}(\tau, \nu)$ induzida por um teste $\tau$ que divide o conjunto de exemplos $S$ que alcançam um nó $\nu$ nos subconjuntos $S^L$ e $S^R$ é definida por:

$$\texttt{Entr}(\tau, \nu) = \frac{|S^L|}{|S|}\texttt{Entr}(S^L) + \frac{|S^R|}{|S|}\texttt{Entr}(S^R).$$

Neste novo experimento, $\texttt{Entr}(\tau, \nu)$ será a medida de impureza a se minimizar na busca de um nova divisão de um nó $\nu$.

Considerando que uma versão normalizada de entropia é utilizada pelo algoritmo amplamente difundido C4.5 [17], implementamos `SER-DT` trocando `Gini` pela medida de entropia. Para possibilitar uma comparação justa, alteramos da mesma forma a implementação de `CART` considerando também a

PUC-Rio - Certificação Digital Nº 2112508/CA

entropia. Os resultados encontram-se na Tabela 4.8 e são semelhantes aos expostos na Seção 4.4.

## 4.8 Outros Exemplos de Árvores

Nesta seção apresentaremos exemplos de outras árvores geradas por `CART` e `SER-DT`. Estes exemplos fornecem evidências adicionais de que nosso algoritmo pode gerar árvores com maior explicabilidade, mas acurácia similar quando comparado com `CART`. Para todas as árvores seguintes foi utilizado limite de altura 4 e `FactorExpl` $= 0.97$ (no caso de `SER-DT`).

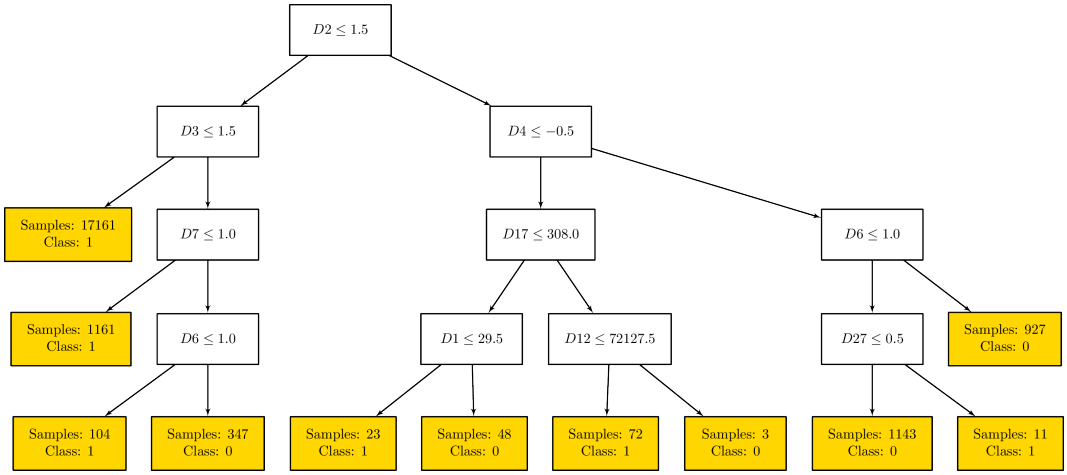


Figura 4.2: Árvore gerada por `CART` para o *dataset Default credit cart*. *Samples* e *Class* representam, respectivamente, o número total e a classe majoritária dos exemplos que alcançam uma folha.

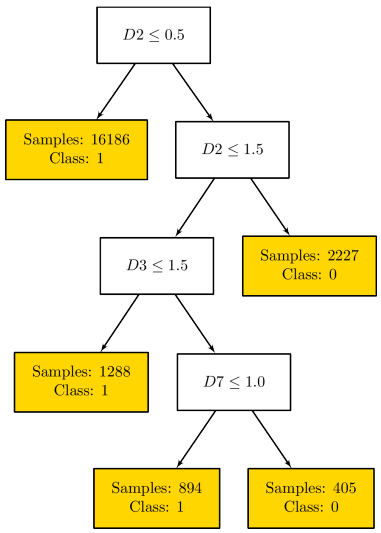


Figura 4.3: Árvore gerada por `SER-DT` para o *dataset Default credit cart*. *Samples* e *Class* representam, respectivamente, o número total e a classe majoritária dos exemplos que alcançam uma folha.

PUC-Rio - Certificação Digital Nº 2112508/CA

O primeiro exemplo apresenta duas árvores geradas a partir do dataset *Default Credit Card*. As Figuras 4.2 e 4.3 exibem árvores geradas por `CART` e `SER-DT`, respectivamente. A árvore gerada por `CART` possui uma acurácia de teste de 81.9%, enquanto `SER-DT` obtém 81.8%. No entanto, é aparente que a árvore gerada por `SER-DT` é muito mais simples do que a gerada por `CART`. Considerando *explanation size*, `CART` possui $\text{expl}_{\text{avg}}$= 2.27 enquanto $\text{expl}_{\text{avg}}$=1.19 para `SER-DT`.

O segundo exemplo emprega o *dataset Online Shoppers Intention*, representado pelas Figuras 4.4 e 4.5. `CART` gera uma árvore com acurácia de teste de 90.0% e `SER-DT`, de 89.1%. Neste caso também é claro que a árvore gerada por `SER-DT` é mais simples do que de `CART`, com $\text{expl}_{\text{avg}}$= 2.38 para `CART` e $\text{expl}_{\text{avg}}$=1.54 para `SER-DT`.

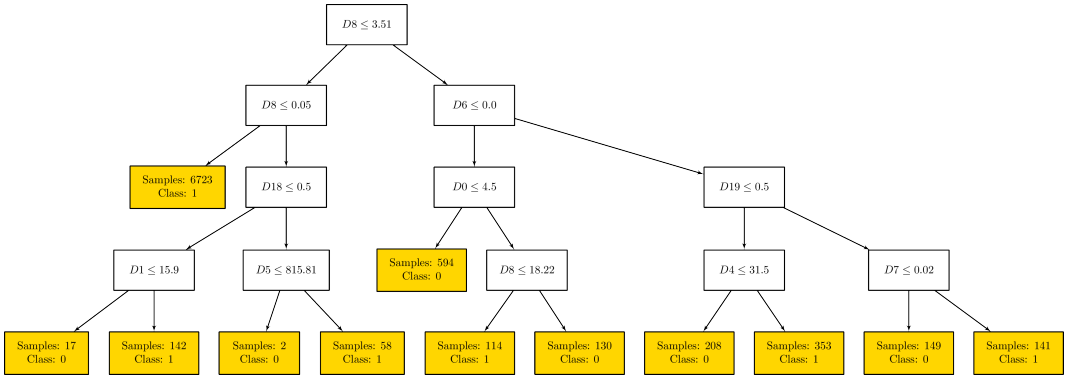


Figura 4.4: Árvore gerada por `CART` para o *dataset Online Shoppers Intention*. *Samples* e *Class* representam, respectivamente, o número total e a classe majoritária dos exemplos que alcançam uma folha.

O último exemplo é exibido nas Figuras 4.6 e 4.7, árvores geradas para o *dataset Dry Bean*. Embora possa parecer à primeira vista que não houve melhoria significativa de explicabilidade para o nosso algoritmo, `CART` gerou uma árvore com $\text{expl}_{\text{avg}}$ igual 3.49 e `SER-DT`, um $\text{expl}_{\text{avg}}$ de 2.60, o que representa uma redução de 25.6%. Considerando a acurácia de teste, `CART` fornece 82.8% e `SER-DT`, 82.4%.

## 4.9 Comparação com $\text{EC}^2$

Como observado na Seção 3.3, `SER-DT` possui a melhor garantia teórica possível e uma bom desempenho, na prática. A presente seção mostra que otimizar somente a média de altura não garante bons resultados em relação à acurácia e *explanation size*, comparando com o algoritmo `SER-DT` desenvolvido.

$\text{EC}^2$ [7] é um algoritmo guloso que obtém uma aproximação $O(\log\ n)$ para a minimização de altura média de árvores. Como mencionado na Seção 1.2,

PUC-Rio - Certificação Digital Nº 2112508/CA

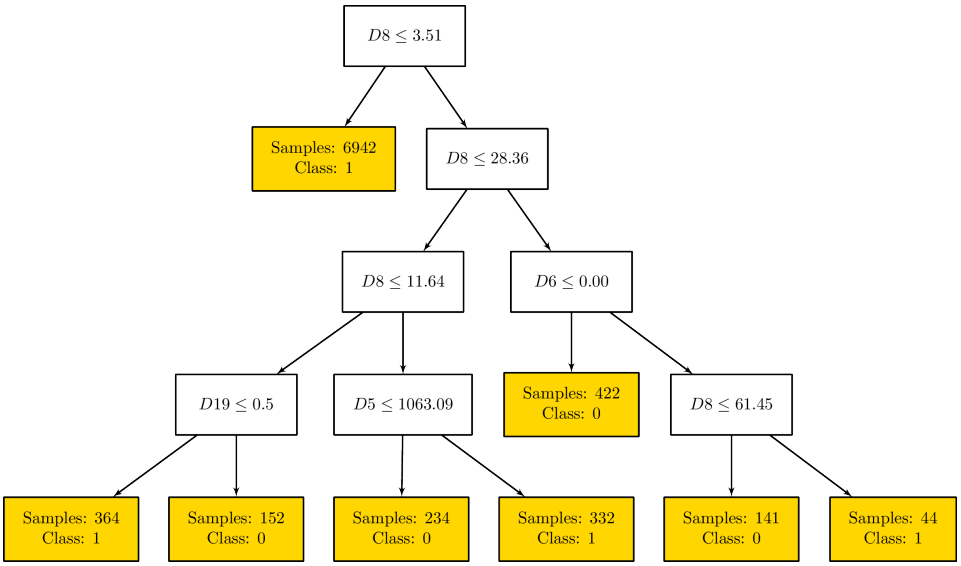


Figura 4.5: Árvore gerada por `SER-DT` para o *dataset Online Shoppers Intention*. *Samples* e *Class* representam, respectivamente, o número total e a classe majoritária dos exemplos que alcançam uma folha.

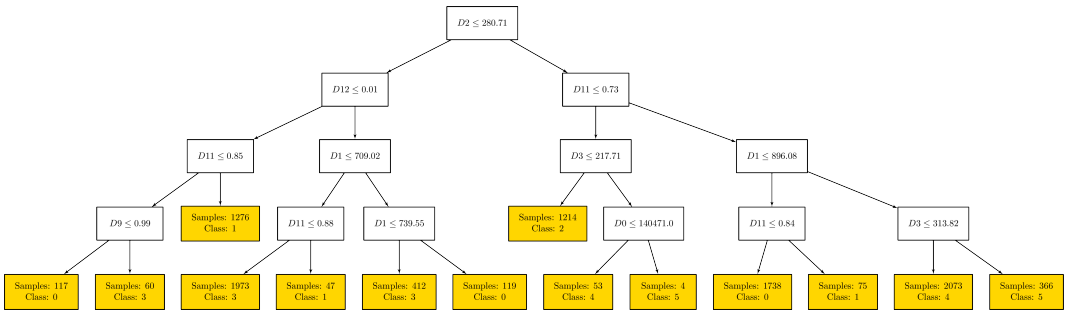


Figura 4.6: Árvore gerada por `CART` para o *dataset Dry Bean*. *Samples* e *Class* representam, respectivamente, o número total e a classe majoritária dos exemplos que alcançam uma folha.

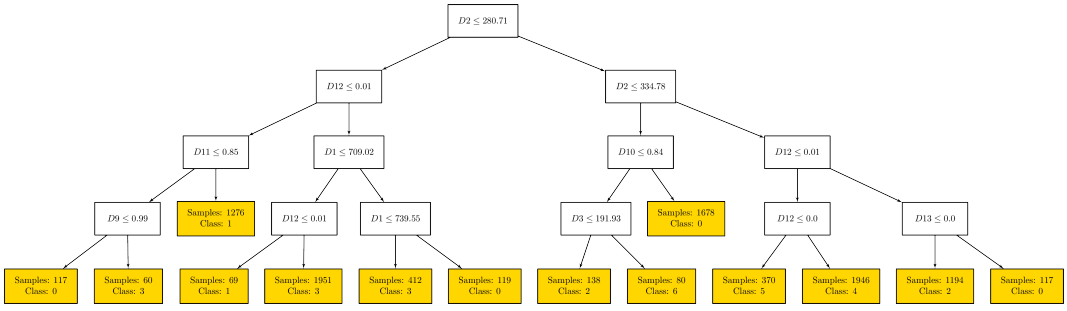


Figura 4.7: Árvore gerada por `SER-DT` para o *dataset Dry Bean*. *Samples* e *Class* representam, respectivamente, o número total e a classe majoritária dos exemplos que alcançam uma folha.

essa é a melhor garantia possível em tempo polinomial considerando $P \neq NP$.

Implementamos `EC`$^2$ e o comparamos com `SER-DT`, da mesma forma que realizamos com `CART` na Seção 4.4. A Tabela 4.9 apresenta a acurácia no

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.9: Acurácia no conjunto de teste, $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$ para `FactorExpl` = 0.97. Cada entrada é uma média de 10 execuções. O intervalo de confiança para as acurácias representa a variação de 1 desvio padrão (acima ou abaixo) sobre a amostra das 10 iterações. Valores em negrito representam diferenças maiores que 1% (em acurácia) ou 25% (nas métricas de explicabilidade).

| *Dataset* | Acurácia de Teste | | $\text{expl}_{\text{avg}}$ | | $\text{expl}_{\text{wc}}$ | |
|---|---|---|---|---|---|---|
| | SER-DT | $\text{EC}^2$ | SER-DT | $\text{EC}^2$ | SER-DT | $\text{EC}^2$ |
| Anuran | **94.8 ± 0.5%** | 89.4 ± 1.7% | 4.78 | 4.36 | 6.00 | 5.90 |
| Audit Risk | 99.9 ± 0.3% | 99.3 ± 0.8% | **1.00** | 3.07 | **1.00** | 3.90 |
| Avila | **61.5 ± 2.1%** | 57.7 ± 1.2% | 3.06 | 4.03 | 4.90 | 6.00 |
| Banknote | **97.6 ± 0.8%** | 92.3 ± 1.7% | 2.44 | 1.88 | 3.80 | 3.00 |
| Bankruptcy Polish | 96.6 ± 0.7% | 97.4 ± 0.4% | 2.56 | 2.19 | 5.60 | 5.50 |
| Cardiotocography | **89.5 ± 1.3%** | 80.1 ± 3.3% | 4.30 | 3.81 | 5.90 | 5.70 |
| Collins | 13.2 ± 1.8% | **16.0 ± 1.2%** | **2.13** | 4.95 | **4.40** | 6.00 |
| Default Credit Card | **82.0 ± 0.4%** | 80.5 ± 0.3% | **1.45** | 3.65 | **4.50** | 6.00 |
| Dry Bean | **90.1 ± 0.4%** | 83.9 ± 0.5% | 3.32 | 4.40 | 5.10 | 6.00 |
| EEG Eye State | **74.1 ± 0.5%** | 71.1 ± 0.5% | 3.69 | 3.78 | 5.90 | 5.60 |
| HTRU2 | 97.7 ± 0.2% | 97.7 ± 0.1% | 1.20 | 1.52 | 4.30 | 4.60 |
| Iris | **94.2 ± 2.6%** | 86.7 ± 7.0% | **1.75** | 2.57 | 3.10 | 3.60 |
| Letter Recognition | **44.9 ± 1.2%** | 37.8 ± 0.5% | **3.34** | 5.32 | 5.50 | 6.00 |
| Mice | **99.9 ± 0.2%** | 71.0 ± 4.1% | **3.05** | 4.79 | **3.60** | 6.00 |
| OBS Network | **91.7 ± 2.1%** | 81.2 ± 1.5% | 3.48 | 3.95 | 5.30 | 5.30 |
| Occupancy Room | 99.4 ± 0.2% | 98.6 ± 0.4% | 4.18 | 4.77 | 5.30 | 5.60 |
| Online Shoppers Intention | 89.3 ± 0.5% | 89.5 ± 0.6% | 3.30 | 1.84 | 5.10 | 5.60 |
| Pen Digits | **88.6 ± 2.0%** | 75.3 ± 1.8% | 4.76 | 4.74 | 5.80 | 6.00 |
| Poker Hand | 52.9 ± 1.0% | 52.0 ± 0.7% | **1.80** | 3.72 | **3.80** | 5.70 |
| Sensorless | **87.4 ± 1.0%** | 74.4 ± 0.4% | **2.94** | 4.19 | 4.90 | 6.00 |
| Média | 82.3% | 76.6% | 2.93 | 3.68 | 4.69 | 5.40 |

Tabela 4.10: Acurácia no conjunto de teste, $\text{depth}_{\text{avg}}$e $\text{depth}_{\text{wc}}$para `FactorExpl` = 0.97. Cada entrada é uma média de 10 execuções. Valores em negrito representam uma redução de pelo menos 25% em favor de `SER-DT`.

| *Dataset* | $\text{depth}_{\text{avg}}$ | | $\text{depth}_{\text{wc}}$ | |
|---|---|---|---|---|
| | SER-DT | $\text{EC}^2$ | SER-DT | $\text{EC}^2$ |
| Anuran | 5.38 | 4.83 | 6.00 | 6.00 |
| Audit Risk | **1.00** | 4.34 | **1.00** | 5.90 |
| Avila | 5.14 | 5.04 | 6.00 | 6.00 |
| Banknote | 4.39 | 3.85 | 6.00 | 6.00 |
| Bankruptcy Polish | 4.45 | 2.40 | 6.00 | 6.00 |
| Cardiotocography | 4.98 | 5.15 | 6.00 | 6.00 |
| Collins | 5.89 | 5.89 | 6.00 | 6.00 |
| Default Credit Card | **2.15** | 3.79 | 6.00 | 6.00 |
| Dry Bean | 4.76 | 5.03 | 6.00 | 6.00 |
| EEG Eye State | 5.15 | 5.19 | 6.00 | 6.00 |
| HTRU2 | 2.80 | 3.26 | 6.00 | 6.00 |
| Iris | **2.50** | 3.34 | 4.90 | 5.70 |
| Letter Recognition | 5.96 | 5.87 | 6.00 | 6.00 |
| Mice | **3.05** | 4.93 | **3.60** | 6.00 |
| OBS Network | 4.47 | 4.62 | 6.00 | 6.00 |
| Occupancy Room | 4.72 | 5.64 | 6.00 | 6.00 |
| Online Shoppers Intention | 3.89 | 3.49 | 6.00 | 6.00 |
| Pen Digits | 5.73 | 5.65 | 6.00 | 6.00 |
| Poker Hand | 4.61 | 4.71 | 6.00 | 6.00 |
| Sensorless | 5.26 | 5.47 | 6.00 | 6.00 |
| Média | 4.31 | 4.62 | 5.58 | 5.98 |

PUC-Rio - Certificação Digital Nº 2112508/CA

conjunto de teste, $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$ para os dois algoritmos. Notamos que `SER-DT` possui acurácia significativamente melhor que `EC`$^2$ em quase todos os datasets e medidas de explicabilidade igualmente melhores. Mais especificamente, em 14 *datasets* observa-se uma diferença maior que 1% nas acurácias entre os dois algoritmos; em 13 deles, nosso algoritmo é melhor que `EC`$^2$, enquanto em apenas em 1 deles, `EC`$^2$ é melhor. Considerando $\text{expl}_{\text{avg}}$, nosso algoritmo é melhor que `EC`$^2$ em 14 datasets e para 8 deles a redução da métrica é de pelo menos 25% (assinalados em negrito). A diferença de $\text{expl}_{\text{wc}}$ também é significativa: para 15 *datasets* `SER-DT` é tão bom quanto `EC`$^2$ e, para 5 destes, há uma redução de 25% nessa métrica.

A Tabela 4.10 apresenta resultados semelhantes relativos às métricas $\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$. Os valores presentes nas colunas de posição 2 e 4 são marcados em negrito quando a diferença em relação a `EC`$^2$ é maior que 25%. Para 12 *datasets* nosso algoritmo é melhor que `EC`$^2$, e para os outros 8 que são piores, para apenas um deles a diferença é maior que 25% (*Bankruptcy Polish*). Os resultados para $\text{depth}_{\text{wc}}$ são similares porque o limite de altura 6 foi mantido para ambos os algoritmos. Ainda assim, para 2 datasets o desempenho de `SER-DT` foi superior (*Audit Risk* e *Mice*)

Tabela 4.11: Acurácia de teste, $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$ para `SER-DT` com `FactorExpl` = 0.97, com e sem poda. Cada entrada é a média de 10 execuções. Valores em negrito indicam uma redução de mais de 20% nas métricas de explicabilidade.

| *Dataset* | Acurácia de Teste | | $\text{expl}_{\text{avg}}$ | | $\text{expl}_{\text{wc}}$ | |
|---|---|---|---|---|---|---|
| | sem poda | com poda | sem poda | com poda | sem poda | com poda |
| Anuran | 94.7% | 94.5% | 4.78 | 4.07 | 6.00 | 6.00 |
| Audit Risk | 99.9% | 99.9% | 1.00 | 1.00 | 1.00 | 1.00 |
| Avila | 61.2% | 60.9% | 3.06 | 2.98 | 4.90 | 4.90 |
| Banknote | 97.8% | 97.8% | 2.44 | 2.37 | 3.80 | 3.40 |
| Bankruptcy Polish | 96.5% | 97.3% | 2.56 | **1.45** | 5.60 | 4.50 |
| Cardiotocography | 89.1% | 89.4% | 4.30 | 3.76 | 5.90 | 5.60 |
| Collins | 13.1% | 13.1% | 2.13 | **1.40** | 4.40 | **3.30** |
| Default Credit Card | 82.0% | 82.0% | 1.45 | 1.29 | 4.50 | 3.90 |
| Dry Bean | 90.1% | 89.9% | 3.32 | 3.17 | 5.10 | 4.70 |
| EEG Eye State | 73.9% | 73.6% | 3.69 | 3.56 | 5.90 | 5.80 |
| HTRU2 | 97.7% | 97.8% | 1.20 | 1.09 | 4.30 | 3.70 |
| Iris | 93.7% | 93.2% | 1.75 | 1.44 | 3.10 | **1.70** |
| Letter Recognition | 44.9% | 44.4% | 3.34 | 3.21 | 5.50 | 5.30 |
| Mice | 100.0% | 100.0% | 3.05 | 3.05 | 3.60 | 3.60 |
| OBS Network | 92.0% | 91.6% | 3.48 | 3.35 | 5.30 | 5.30 |
| Occupancy Room | 99.3% | 99.3% | 4.18 | **3.22** | 5.30 | 4.70 |
| Online Shoppers Intention | 89.3% | 89.8% | 3.30 | **1.81** | 5.10 | **4.00** |
| Pen Digits | 88.7% | 88.3% | 4.76 | 4.56 | 5.80 | 5.40 |
| Poker Hand | 52.9% | 52.9% | 1.80 | 1.78 | 3.80 | 3.80 |
| Sensorless | 87.4% | 87.4% | 2.94 | 2.81 | 4.90 | 4.80 |
| Média | 82.2% | 82.1% | 2.93 | 2.57 | 4.69 | 4.27 |

PUC-Rio - Certificação Digital Nº 2112508/CA

## 4.10 Pós-poda

Algoritmos de pós-poda são interessantes para reduzir a complexidade de uma árvore e podem ser úteis, portanto, na melhoria das métricas de explicabilidade. Além do Algoritmo 3, descrito na Seção 4.2, desenvolvemos outro algoritmo de pós-poda com o seguinte comportamento: checamos se a junção de duas folhas irmãs não diminuiria a acurácia medida em um conjunto de validação formado pela metade do conjunto de teste original. Caso afirmativo, unimos essas duas folhas, aplicando esse processo recursivamente até a raiz da árvore. Este processo está descrito no Algoritmo 4.

**Algoritmo 4:** Post-Pruning

**Input:** Um nó $\alpha$
**if** `PodaOuNão(`$\alpha$`)` $=$ `Verdadeiro` **then**
  Remova as subárvores geradas pelos filhos de $\alpha$ (se houver);
**end**
$\alpha_l \leftarrow$ filho à esquerda de $\alpha$;
$\alpha_r \leftarrow$ filho à direita de $\alpha$;
`Post-Pruning(`$\alpha_l$`)`;
`Post-Pruning(`$\alpha_r$`)`;
**Procedimento** `PodaOuNão`
  **Input:** Um nó $\nu$
  **Output:** `Verdadeiro`, se as subárvores geradas pelos filhos de $\nu$ devem ser podadas; `Falso`, caso contrário
  **if** *$\nu$ é uma folha* **then**
    **return** `Verdadeiro`;
  **end**
  $\nu_l \leftarrow$ filho à esquerda de $\nu$;
  $\nu_r \leftarrow$ filho à direita de $\nu$;
  **if** `PodaOuNão(`$\nu_l$`)` $=$ `Verdadeiro` *e* `PodaOuNão(`$\nu_r$`)` $=$ `Verdadeiro` *e Acurácia da árvore no conjunto de validação ao se juntar os nós $\nu_l$ e $\nu_r$ não é menor que a acurácia atual da árvore* **then**
    **return** `Verdadeiro`;
  **end**
  **else**
    **return** `Falso`;
  **end**

A Tabela 4.11 apresenta os resultados para `SER-DT` sem e com pós-poda, limitando a altura em 6 e `FactorExpl` $= 0.97$. Observamos que a utilização de pós-poda não apresentou efeito significativo na acurácia, levando a diferenças

PUC-Rio - Certificação Digital Nº 2112508/CA

não maiores que 0.5% para todos os datasets. Em relação a $\text{expl}_{\text{avg}}$, a estratégia de poda produziu resultados melhores para todos os *datasets* — o que é esperado, pois estamos unindo folhas. Para 4 datasets, a diminuição dessa métrica foi de pelo menos 20% (em negrito na tabela). Da mesma forma, a algoritmo de pós-poda levou a valores menores ou iguais de $\text{expl}_{\text{wc}}$ para todos os datasets e, para 3 destes, a diferença foi de pelo menos 20%.

A Tabela 4.12 mostra os resultados equivalentes à tabela anterior, mas relacionados à altura das árvores ($\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$). Para todos os *datasets*, não há aumento das métricas aplicando-se pós-poda. Em 6 destes, há uma melhoria de mais de 20% para $\text{depth}_{\text{avg}}$; para 6 *datasets*, há uma melhoria de mais de 20% para $\text{depth}_{\text{wc}}$. Tais valores também são assinalados em negrito na tabela.

Tabela 4.12: $\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$ para `SER-DT` com `FactorExpl` = 0.97, com e sem poda. Cada entrada é a média de 10 execuções. Valores em negrito indicam uma redução de mais de 20% nas métricas de explicabilidade.

| *Dataset* | $\text{depth}_{\text{avg}}$ | | $\text{depth}_{\text{wc}}$ | |
|---|---|---|---|---|
| | sem poda | com poda | sem poda | com poda |
| Anuran | 5.38 | 4.44 | 6.00 | 6.00 |
| Audit Risk | 1.00 | 1.00 | 1.00 | 1.00 |
| Avila | 5.14 | 4.99 | 6.00 | 6.00 |
| Banknote | 4.39 | 3.69 | 6.00 | 5.70 |
| Bankruptcy Polish | 4.45 | **2.01** | 6.00 | 5.30 |
| Cardiotocography | 4.98 | 4.12 | 6.00 | 6.00 |
| Collins | 5.89 | **4.38** | 6.00 | 5.90 |
| Default Credit Card | 2.15 | 1.76 | 6.00 | 5.90 |
| Dry Bean | 4.76 | 4.40 | 6.00 | 6.00 |
| EEG Eye State | 5.15 | 4.85 | 6.00 | 6.00 |
| HTRU2 | 2.80 | **2.12** | 6.00 | 5.80 |
| Iris | 2.50 | **1.74** | 4.90 | **2.30** |
| Letter Recognition | 5.96 | 5.77 | 6.00 | 6.00 |
| Mice | 3.05 | 3.05 | 3.60 | 3.60 |
| OBS Network | 4.47 | 4.31 | 6.00 | 6.00 |
| Occupancy Room | 4.72 | **3.52** | 6.00 | 6.00 |
| Online Shoppers Intention | 3.89 | **2.19** | 6.00 | 5.90 |
| Pen Digits | 5.73 | 5.46 | 6.00 | 6.00 |
| Poker Hand | 4.61 | 4.56 | 6.00 | 6.00 |
| Sensorless | 5.26 | 5.08 | 6.00 | 6.00 |
| Média | 4.31 | 3.67 | 5.58 | 5.37 |

É interessante analisar também o efeito que o algoritmo de pós-poda possui no `CART` quando comparado ao nosso algoritmo. As Tabelas 4.13 e 4.14 relacionam as métricas de acurácia e explicabilidade obtidas pelos dois algoritmos utilizando-se pós-poda. Observamos que não houve diferença

PUC-Rio - Certificação Digital Nº 2112508/CA

significativa em relação à acurácia: em 7 dos *datasets* houve diferença maior que 1% entre `CART` e `SER-DT`; em 3 deles `SER-DT` é melhor que `CART`, e nos outros 4, `CART` é melhor. No entanto, considerando $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$, é clara a superioridade de `SER-DT`. Somente para 2 *datasets* `CART` possui $\text{expl}_{\text{avg}}$ menor que `SER-DT` e para 7 *datasets* nosso algoritmo é melhor que `CART` em pelo menos 20%. Em relação a $\text{expl}_{\text{wc}}$, nosso algoritmo perde para `CART` em somente 3 datasets e para 4 a melhoria foi de pelo menos 20% quando comparado ao `CART`.

As métricas relacionadas à altura são mostradas na Tabela 4.14. Nota-se valores mais semelhantes entre `CART` e `SER-DT` do que aqueles apresentados na Tabela 4.3. Considerando $\text{depth}_{\text{avg}}$, nosso algoritmo é melhor em 12, pior em 6 e igual em 2 *datasets* em relação a `CART`. Somente para 3 *datasets* a diferença é mais que 20% e para 2 deles `SER-DT` é melhor. Em relação a $\text{depth}_{\text{wc}}$, observamos valores próximos ao se comparar os dois algoritmos; somente para o dataset *Bankruptcy* a diferença é maior que 20%. Em geral, aliás, nota-se que *Bankruptcy* destaca-se como um *outlier* em relação às métricas de explicabilidade neste experimento.

PUC-Rio - Certificação Digital Nº 2112508/CA

Tabela 4.13: Acurácia no conjunto de teste, $\text{expl}_{\text{avg}}$ e $\text{expl}_{\text{wc}}$para `SER-DT` (`FactorExpl` = 0.97) e `CART`, ambos utilizando pós-poda. Cada entrada é a média de 10 execuções. Os valores em negrito representar melhorias de pelo menos 1% em acurácias (colunas 2 e 3) ou 20% nas métricas de explicabilidade (colunas 4, 5, 6 e 7).

| *Dataset* | Acurácia de Teste | | $\text{expl}_{\text{avg}}$ | | $\text{expl}_{\text{wc}}$ | |
|---|---|---|---|---|---|---|
| | `CART` | `SER-DT` | `CART` | `SER-DT` | `CART` | `SER-DT` |
| Anuran | 94.3% | 94.5% | 4.58 | 4.07 | 6.00 | 6.00 |
| Audit Risk | 99.9% | 99.9% | 1.00 | 1.00 | 1.00 | 1.00 |
| Avila | **62.7%** | 60.9% | 4.17 | **2.98** | 5.40 | 4.90 |
| Banknote | 98.2% | 97.8% | 2.51 | 2.37 | 3.20 | 3.40 |
| Bankruptcy Polish | 97.5% | 97.3% | **1.15** | 1.45 | **2.50** | 4.50 |
| Cardiotocography | 89.4% | 89.4% | 4.02 | 3.76 | 5.40 | 5.60 |
| Collins | **14.5%** | 13.1% | 3.94 | **1.40** | 5.40 | **3.30** |
| Default Credit Card | 82.1% | 82.0% | 2.38 | **1.29** | 5.10 | **3.90** |
| Dry Bean | 89.7% | 89.9% | 3.89 | 3.17 | 5.80 | 4.70 |
| EEG Eye State | 72.9% | 73.6% | 4.18 | 3.56 | 5.80 | 5.80 |
| HTRU2 | 97.8% | 97.8% | 1.15 | 1.09 | 3.70 | 3.70 |
| Iris | 93.7% | 93.2% | 1.54 | 1.44 | 1.80 | 1.70 |
| Letter Recognition | **47.9%** | 44.4% | 5.37 | **3.21** | 6.00 | 5.30 |
| Mice | 100.0% | 100.0% | 3.05 | 3.05 | 3.60 | 3.60 |
| OBS Network | 88.4% | **91.6%** | 4.09 | 3.35 | 5.80 | 5.30 |
| Occupancy Room | 99.3% | 99.3% | 3.17 | 3.22 | 4.90 | 4.70 |
| Online Shoppers Intention | 89.8% | 89.8% | 2.49 | **1.81** | 5.50 | **4.00** |
| Pen Digits | 86.1% | **88.3%** | 4.98 | 4.56 | 6.00 | 5.40 |
| Poker Hand | **55.0%** | 52.9% | 4.28 | **1.78** | 5.10 | **3.80** |
| Sensorless | 80.1% | **87.4%** | 3.85 | **2.81** | 5.40 | 4.80 |
| Média | 82.0% | 82.1% | 3.29 | 2.57 | 4.67 | 4.27 |

Tabela 4.14: $\text{depth}_{\text{avg}}$ e $\text{depth}_{\text{wc}}$ para `SER-DT` (`FactorExpl` = 0.97) e `CART`, ambos com pós-poda. Os valores em negrito representam uma melhoria de pelo menos 20% nas métricas de explicabilidade quando `SER-DT` é comparado com `CART`.

| *Dataset* | $\text{depth}_{\text{avg}}$ | | $\text{depth}_{\text{wc}}$ | |
|---|---|---|---|---|
| | `CART` | `SER-DT` | `CART` | `SER-DT` |
| Anuran | 4.72 | 4.44 | 6.00 | 6.00 |
| Audit Risk | 1.00 | 1.00 | 1.00 | 1.00 |
| Avila | 5.19 | 4.99 | 6.00 | 6.00 |
| Banknote | 3.74 | 3.69 | 6.00 | 5.70 |
| Bankruptcy Polish | **1.17** | 2.01 | **2.60** | 5.30 |
| Cardiotocography | 4.19 | 4.12 | 5.70 | 6.00 |
| Collins | 4.95 | 4.38 | 6.00 | 5.90 |
| Default Credit Card | 2.39 | **1.76** | 5.10 | 5.90 |
| Dry Bean | 4.55 | 4.40 | 6.00 | 6.00 |
| EEG Eye State | 5.22 | 4.85 | 6.00 | 6.00 |
| HTRU2 | 2.52 | 2.12 | 5.80 | 5.80 |
| Iris | 1.71 | 1.74 | 2.10 | 2.30 |
| Letter Recognition | 5.80 | 5.77 | 6.00 | 6.00 |
| Mice | 3.05 | 3.05 | 3.60 | 3.60 |
| OBS Network | 4.19 | 4.31 | 6.00 | 6.00 |
| Occupancy Room | 3.29 | 3.52 | 5.70 | 6.00 |
| Online Shoppers Intention | 2.75 | **2.19** | 6.00 | 5.90 |
| Pen Digits | 5.41 | 5.46 | 6.00 | 6.00 |
| Poker Hand | 4.28 | 4.56 | 5.20 | 6.00 |
| Sensorless | 5.12 | 5.08 | 6.00 | 6.00 |
| Média | 3.76 | 3.67 | 5.14 | 5.37 |

PUC-Rio - Certificação Digital Nº 2112508/CA

# 5
# Conclusões

Neste trabalho, apresentamos uma nova métrica de explicabilidade para árvores de decisões, chamada *explanation size*. Tal métrica reflete o número de atributos necessários para explicar a classificação de uma folha, em contraste às métricas de altura e número de nós usualmente utilizadas. Um algoritmo, intitulado `SER-DT`, foi implementado com a finalidade de construir árvores com *explanation size* pequenos. O algoritmo possui uma aproximação $O(\log(n))$ não somente para a minimização no pior caso e caso médio do *explanation size*, mas também para a otimização de altura da árvore (também no pior caso e caso médio). Tal aproximação é ótima considerando $P \neq NP$.

Uma série de experimentos foi realizada com o intuito de comparar `SER-DT` com algoritmos conhecidos da área, como `CART` e `EC`$^2$. Em geral, `SER-DT` produz árvores com acurácias próximas, mas muito mais explicáveis (considerando *explanation size*) em relação aos outros algoritmos considerados. Além disso, a análise de estratégias de poda e averiguação de hiperparâmetros forneceram evidências adicionais de que nosso algoritmo pode ser utilizado em situações nas quais árvores explicáveis são desejadas.

PUC-Rio - Certificação Digital Nº 2112508/CA

## Referências bibliográficas


[1] GILPIN, L. H.; BAU, D.; YUAN, B. Z.; BAJWA, A.; SPECTER, M. A. ; KAGAL, L.. **Explaining explanations: An overview of interpretability of machine learning**. 2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA), p. 80–89, 2018.

[2] GOOGLE. **Google trends**. `https://www.google.com/trends`, 2023. Acesso: em 5 fev. 2023.

[3] BREIMAN, L.; FRIEDMAN, J.; STONE, C. J. ; OLSHEN, R.. **Classification and Regression Trees**. Chapman and Hall/CRC, 1984.

[4] LABER, E. S.; NOGUEIRA, L. T.. **On the hardness of the minimum height decision tree problem**. Discret. Appl. Math., 144(1-2):209–212, 2004.

[5] CHAKARAVARTHY, V. T.; PANDIT, V.; ROY, S.; AWASTHI, P. ; MOHANIA, M. K.. **Decision trees for entity identification: Approximation algorithms and hardness results**. ACM Trans. Algorithms, 7(2):15:1–15:22, 2011.

[6] SOUZA, V. F.; CICALESE, F.; LABER, E. S. ; MOLINARO, M.. **Decision trees with short explainable rules**. In: Oh, A. H.; Agarwal, A.; Belgrave, D. ; Cho, K., editors, ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 2022.

[7] GOLOVIN, D.; KRAUSE, A. ; RAY, D.. **Near-optimal bayesian active learning with noisy observations**. In: Lafferty, J. D.; Williams, C. K. I.; Shawe-Taylor, J.; Zemel, R. S. ; Culotta, A., editors, ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 23: 24TH ANNUAL CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS 2010. PROCEEDINGS OF A MEETING HELD 6-9 DECEMBER 2010, VANCOUVER, BRITISH COLUMBIA, CANADA, p. 766–774. Curran Associates, Inc., 2010.

[8] GUILLORY, A.; BILMES, J. A.. **Average-case active learning with costs**. In: Gavaldà, R.; Lugosi, G.; Zeugmann, T. ; Zilles, S., editors, ALGORITHMIC LEARNING THEORY, 20TH INTERNATIONAL CONFERENCE,


PUC-Rio - Certificação Digital Nº 2112508/CA

(ALT) 2009, PORTO, PORTUGAL, OCTOBER 3-5, 2009. PROCEEDINGS, volumen 5809 de **Lecture Notes in Computer Science**, p. 141–155. Springer, 2009.

[9] CICALESE, F.; LABER, E. S. ; SAETTLER, A. M.. **Diagnosis determination: decision trees optimizing simultaneously worst and expected testing cost**. In: PROCEEDINGS OF THE 31TH INTERNATIONAL CONFERENCE ON MACHINE LEARNING, ICML 2014, BEIJING, CHINA, 21-26 JUNE 2014, volumen 32 de **JMLR Workshop and Conference Proceedings**, p. 414–422. JMLR.org, 2014.

[10] BELLALA, G.; BHAVNANI, S. K. ; SCOTT, C.. **Group-based active query selection for rapid diagnosis in time-critical situations**. IEEE Transactions on Information Theory, 58(1):459–478, 2012.

[11] GUPTA, A.; KRISHNASWAMY, R.; NAGARAJAN, V. ; RAVI, R.. **Approximation algorithms for optimal decision trees and adaptive (tsp) problems**. CoRR, abs/1003.0722, 2010.

[12] HWANG, S.; YEO, H. G. ; HONG, J.. **A new splitting criterion for better interpretable trees**. IEEE Access, 8:62762–62774, 2020.

[13] BERTSIMAS, D.; DUNN, J.. **Optimal classification trees**. Mach. Learn., 106(7):1039–1082, 2017.

[14] VERWER, S.; ZHANG, Y.. **Learning optimal classification trees using a binary linear program formulation**. In: THE THIRTY-THIRD AAAI CONFERENCE ON ARTIFICIAL INTELLIGENCE, (AAAI), 2019, THE THIRTY-FIRST INNOVATIVE APPLICATIONS OF ARTIFICIAL INTELLIGENCE CONFERENCE, IAAI 2019, THE NINTH AAAI SYMPOSIUM ON EDUCATIONAL ADVANCES IN ARTIFICIAL INTELLIGENCE, EAAI 2019, HONOLULU, HAWAII, USA, JANUARY 27 - FEBRUARY 1, 2019, p. 1625–1632. AAAI Press, 2019.

[15] LIN, J.; ZHONG, C.; HU, D.; RUDIN, C. ; SELTZER, M. I.. **Generalized and scalable optimal sparse decision trees**. In: PROCEEDINGS OF THE 37TH INTERNATIONAL CONFERENCE ON MACHINE LEARNING, ICML 2020, 13-18 JULY 2020, VIRTUAL EVENT, volumen 119 de **Proceedings of Machine Learning Research**, p. 6150–6160. PMLR, 2020.

[16] BUHRMAN, H.; DE WOLF, R.. **Complexity measures and decision tree complexity: a survey**. Theor. Comput. Sci., 288(1):21–43, 2002.

PUC-Rio - Certificação Digital Nº 2112508/CA

[17] QUINLAN, J. R.. **C4.5: programs for machine learning**. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[18] ZHENG, A.; CASARI, A.. **Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists**. O'Reilly, 2018.

[19] QUINLAN, J. R.. **Induction of decision trees**. Machine Learning, 1:81–106, 1986.

[20] HUNT, E. B.; MARIN, J. ; STONE, P. J.. **Experiments in Induction**. Academic Press, New York, 1966.

[21] ADLER, M.; HEERINGA, B.. **Approximating optimal binary decision trees**. Algorithmica, 62(3-4):1112–1121, 2012.

[22] PILTAVER, R.; LUŠTREK, M.; GAMS, M. ; MARTINČIĆ-IPŠIĆ, S.. **What makes classification trees comprehensible?** Expert Syst. Appl., 62(C):333–346, nov 2016.

[23] AHLSWEDE, R.; WEGENER, I.. **Search problems**. John Wiley & Sons, Inc., 1987.

[24] KOSARAJU, S. R.; PRZYTYCKA, T. M. ; BORGSTROM, R. S.. **On an optimal split tree problem**. In: PROCEEDINGS OF THE 6TH INTERNATIONAL WORKSHOP ON ALGORITHMS AND DATA STRUCTURES, WADS '99, p. 157–168, London, UK, 1999. Springer-Verlag.

[25] JUAN, C.; NAKAMURA, E.; CRISTO, M. ; GORDO, M.. **Anuran Calls (MFCCs)**. UCI Machine Learning Repository, 2017.

[26] NISHTHA, H.. **Audit Data**. UCI Machine Learning Repository, 2018.

[27] STEFANO, C. D.; MANIACI, M.; FONTANELLA, F. ; DI FRECA, A. S.. **Reliable writer identification in medieval manuscripts through page layout features: The "avila" bible case**. Engineering Applications of Artificial Intelligence, 72:99–110, 2018.

[28] **Banknote authentication**. UCI Machine Learning Repository, 2013.

[29] TOMCZAK, S.. **Polish companies bankruptcy data**. UCI Machine Learning Repository, 2016.

[30] CAMPOS, D.; BERNARDES, J.. **Cardiotocography**. UCI Machine Learning Repository, 2010.

[31] VANSCHOREN, J.. **collins dataset**. OpenML.

PUC-Rio - Certificação Digital Nº 2112508/CA

[32] YEH, I.-C.; HUI LIEN, C.. **The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients**. Expert Systems with Applications, 36(2, Part 1):2473–2480, 2009.

[33] KOKLU, M.; OZKAN, I. A.. **Multiclass classification of dry beans using computer vision and machine learning techniques**. Computers and Electronics in Agriculture, 174:105507, 2020.

[34] ROESLER, O.. **EEG Eye State**. UCI Machine Learning Repository, 2013.

[35] LYON, R. J.; STAPPERS, B. W.; COOPER, S.; BROOKE, J. M. ; KNOWLES, J. D.. **Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach**. Monthly Notices of the Royal Astronomical Society, 459(1):1104–1123, 04 2016.

[36] FISHER, R.. **Iris**. UCI Machine Learning Repository, 1988.

[37] **Letter recognition**. UCI Machine Learning Repository, 1990.

[38] HIGUERA, C.; GARDINER, K. ; CIOS, K.. **Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome**. PloS one, 10:e0129126, 06 2015.

[39] RAJAB, A.. **Burst Header Packet (BHP) flooding attack on Optical Burst Switching (OBS) Network**. UCI Machine Learning Repository, 2017.

[40] R, A.. **Room occupancy estimation data set**. Kaggle.

[41] SAKAR, C. O.; POLAT, S.; KATIRCIOGLU, M. ; KASTRO, Y.. **Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and lstm recurrent neural networks**. Neural Computing and Applications, 31, 10 2019.

[42] ALPAYDIN, E.; ALIMOGLU, F.. **Pen-Based Recognition of Handwritten Digits**. UCI Machine Learning Repository, 1998.

[43] CATTRAL, R.; OPPACHER, F.. **Poker Hand**. UCI Machine Learning Repository, 2006.

[44] BATOR, M.. **Dataset for Sensorless Drive Diagnosis**. UCI Machine Learning Repository, 2015.

PUC-Rio - Certificação Digital Nº 2112508/CA

# A
# Diagramas de caixa para CART e SER-DT

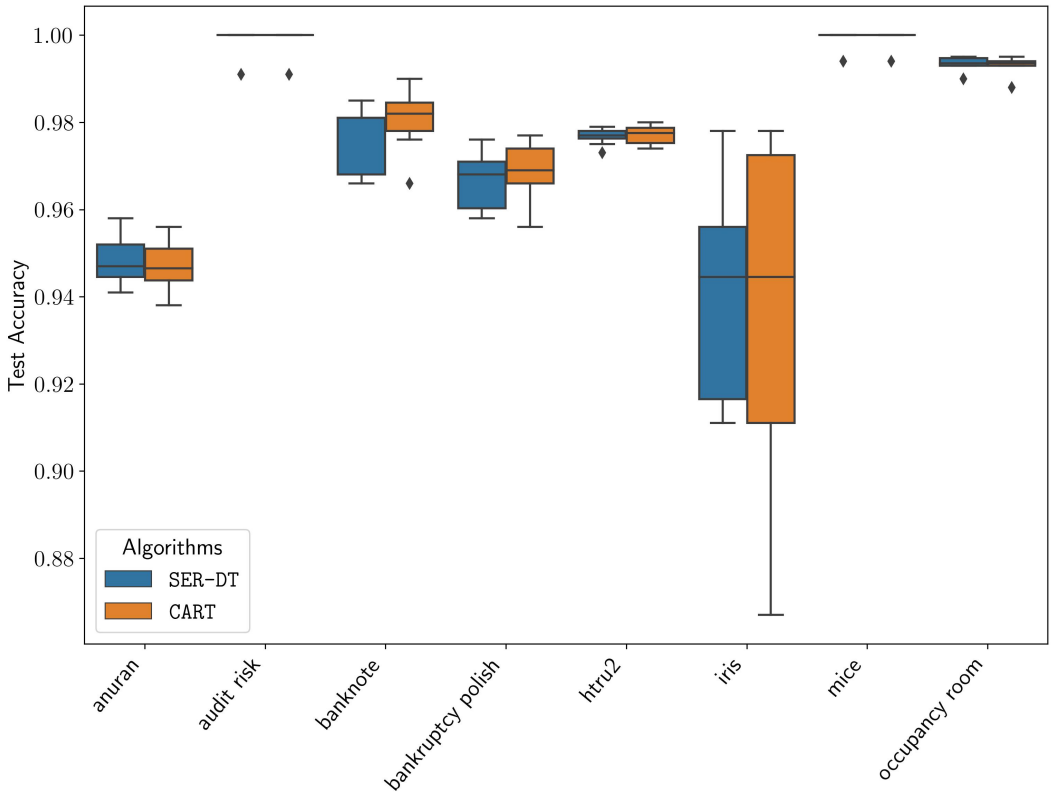


Figura A.1: Acurácia no conjunto de teste para CART e SER-DT para alguns *datasets*.

PUC-Rio - Certificação Digital Nº 2112508/CA

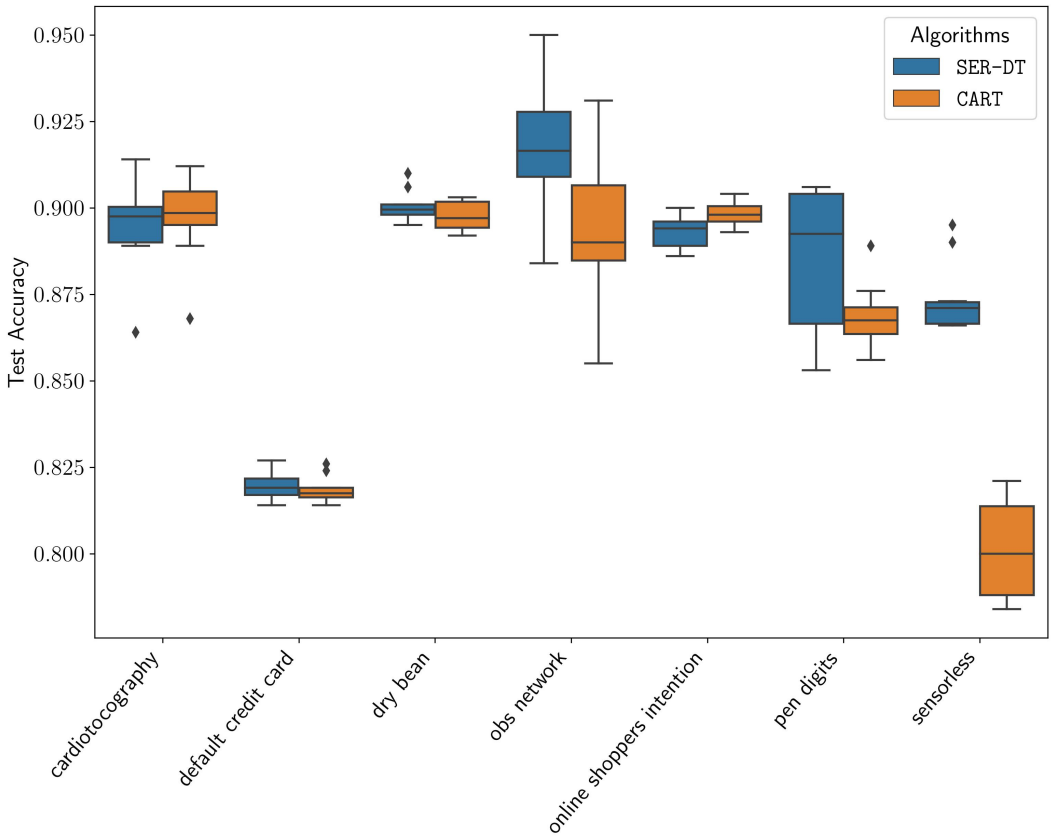


Figura A.2: Acurácia no conjunto de teste para `CART` e `SER-DT` para alguns *datasets*.

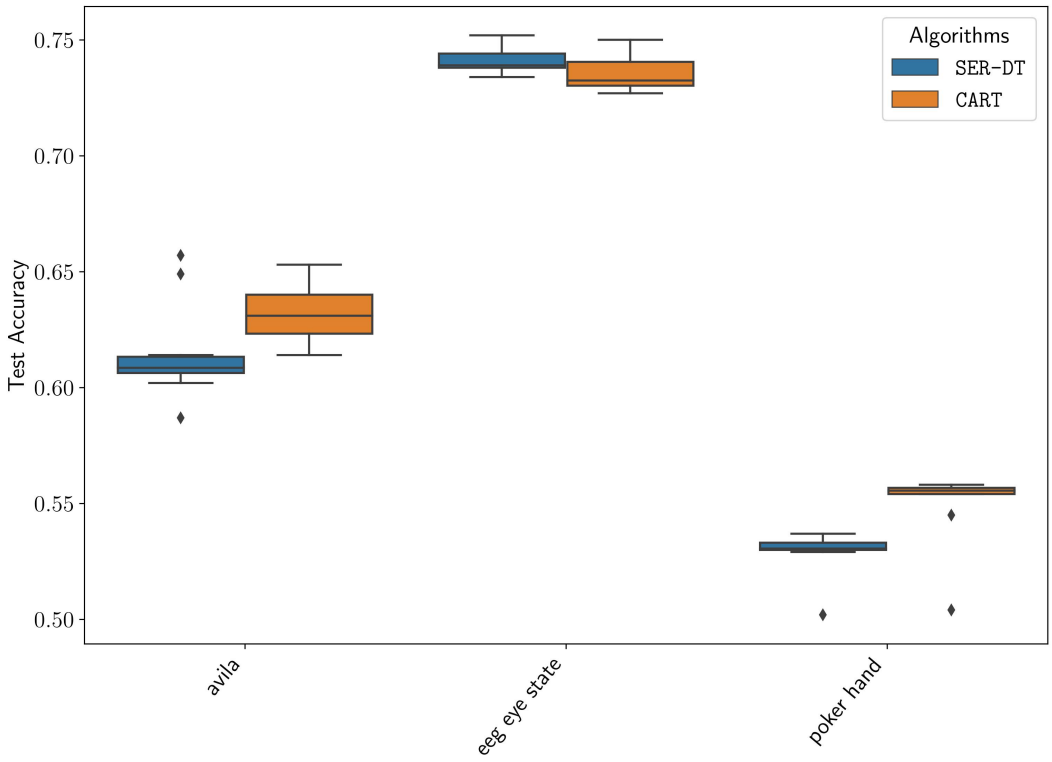


Figura A.3: Acurácia no conjunto de teste para `CART` e `SER-DT` para alguns *datasets*.

PUC-Rio - Certificação Digital Nº 2112508/CA

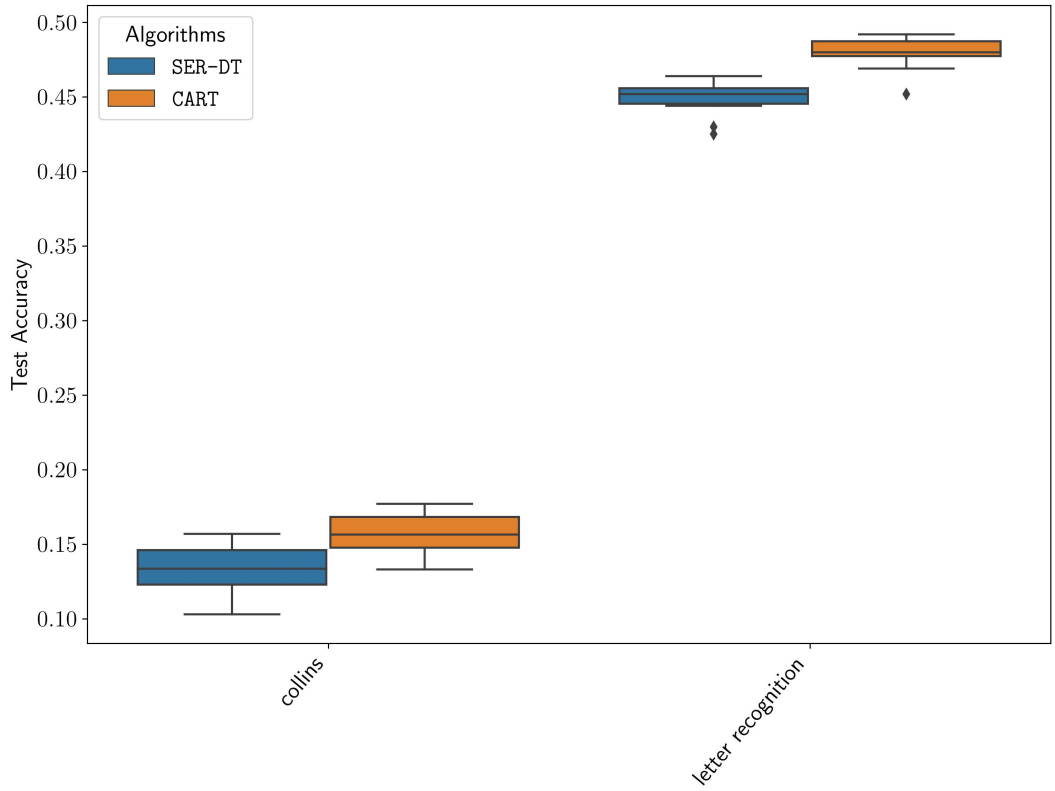


Figura A.4: Acurácia no conjunto de teste para `CART` e `SER-DT` para alguns *datasets*.

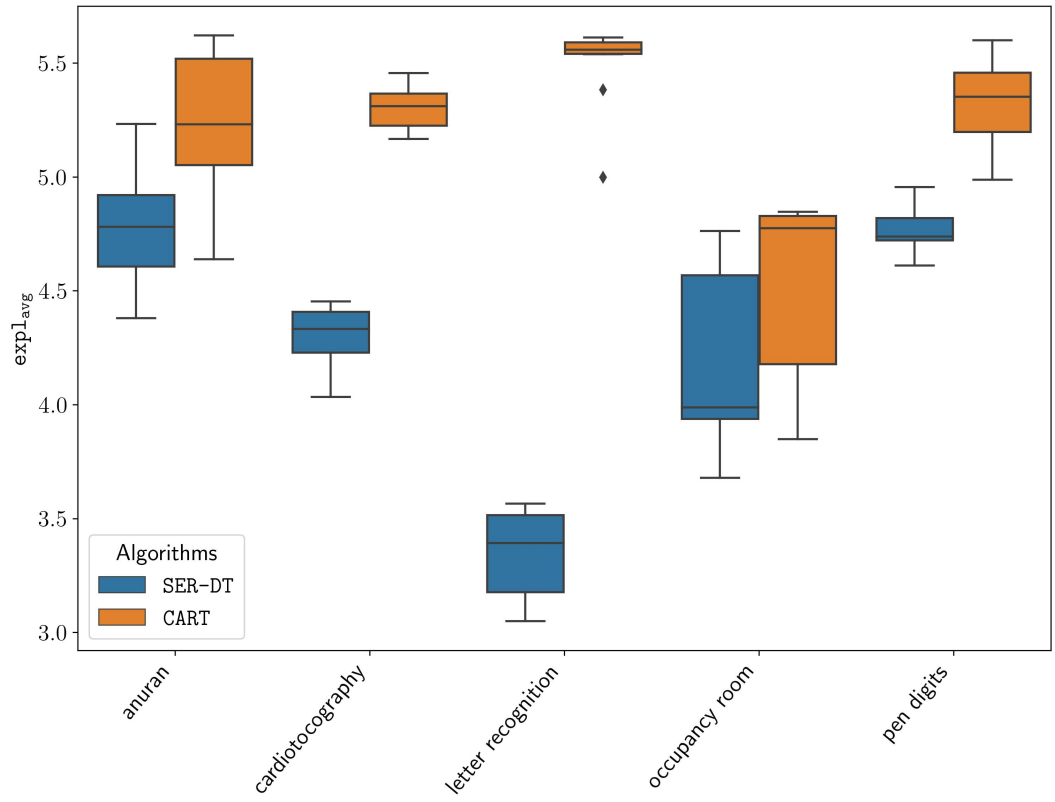


Figura A.5: $expl_{avg}$ para `CART` e `SER-DT` para alguns *datasets*.

PUC-Rio - Certificação Digital Nº 2112508/CA

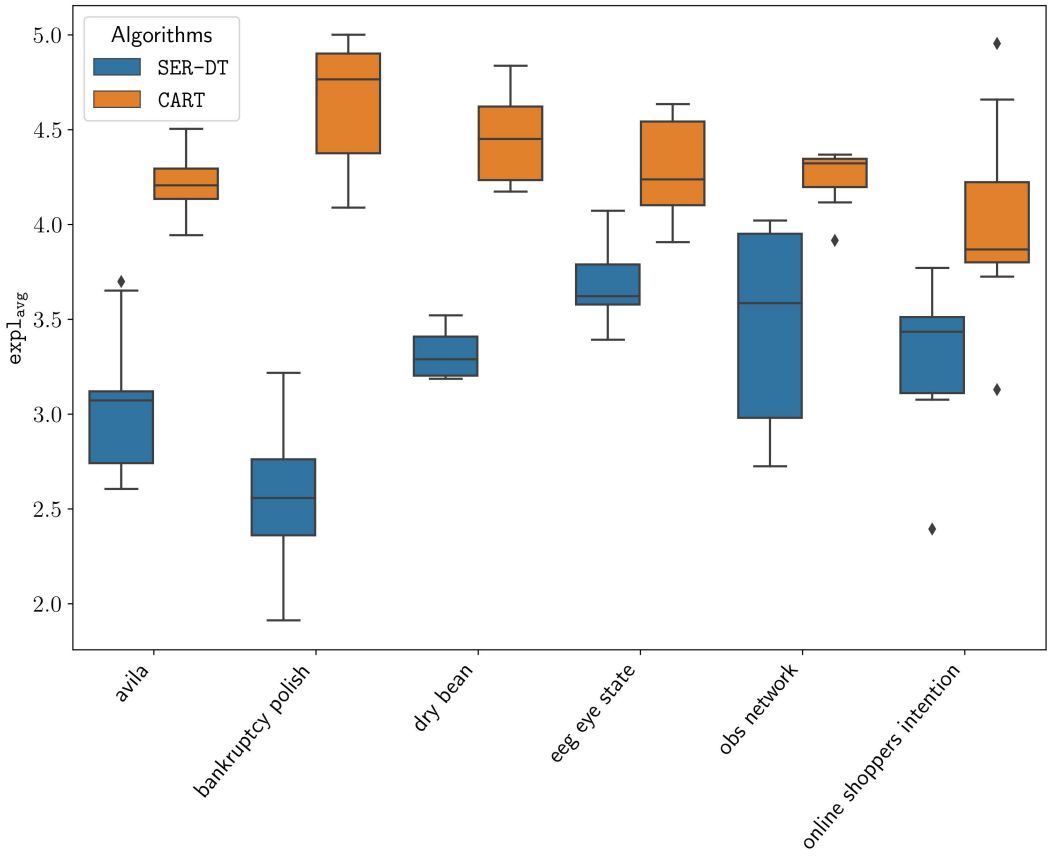


Figura A.6: $expl_{avg}$ para CART e SER-DT para alguns *datasets*.

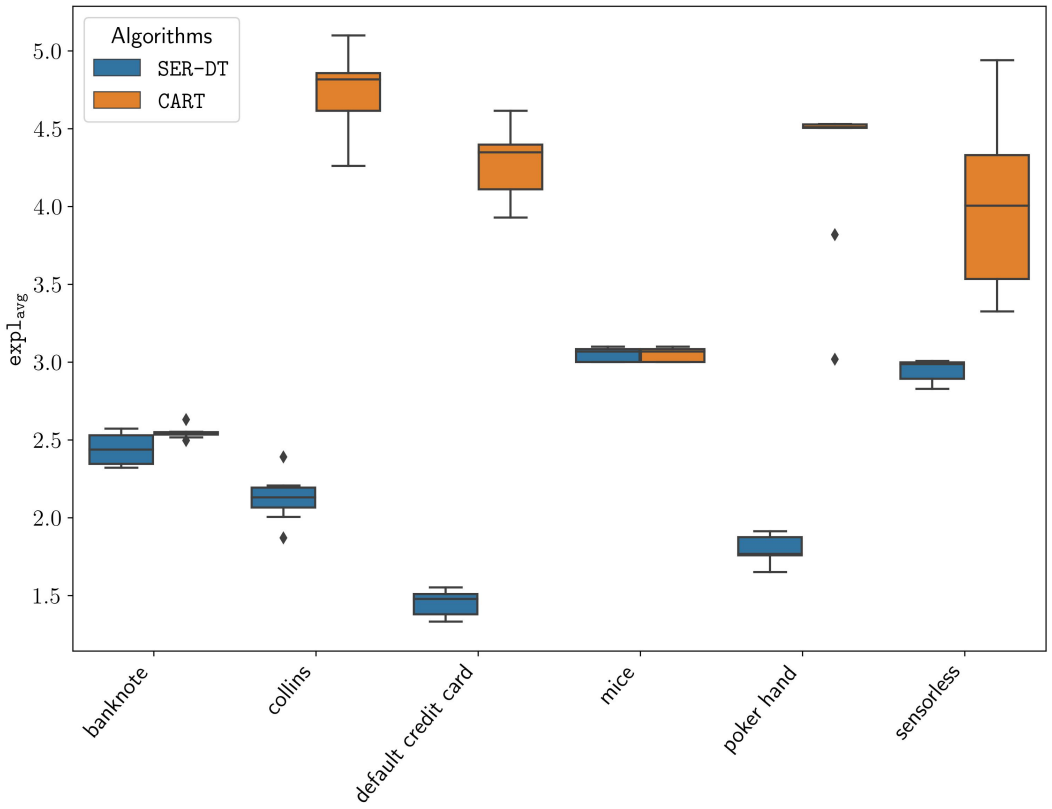


Figura A.7: $expl_{avg}$ para CART e SER-DT para alguns *datasets*.

PUC-Rio - Certificação Digital Nº 2112508/CA

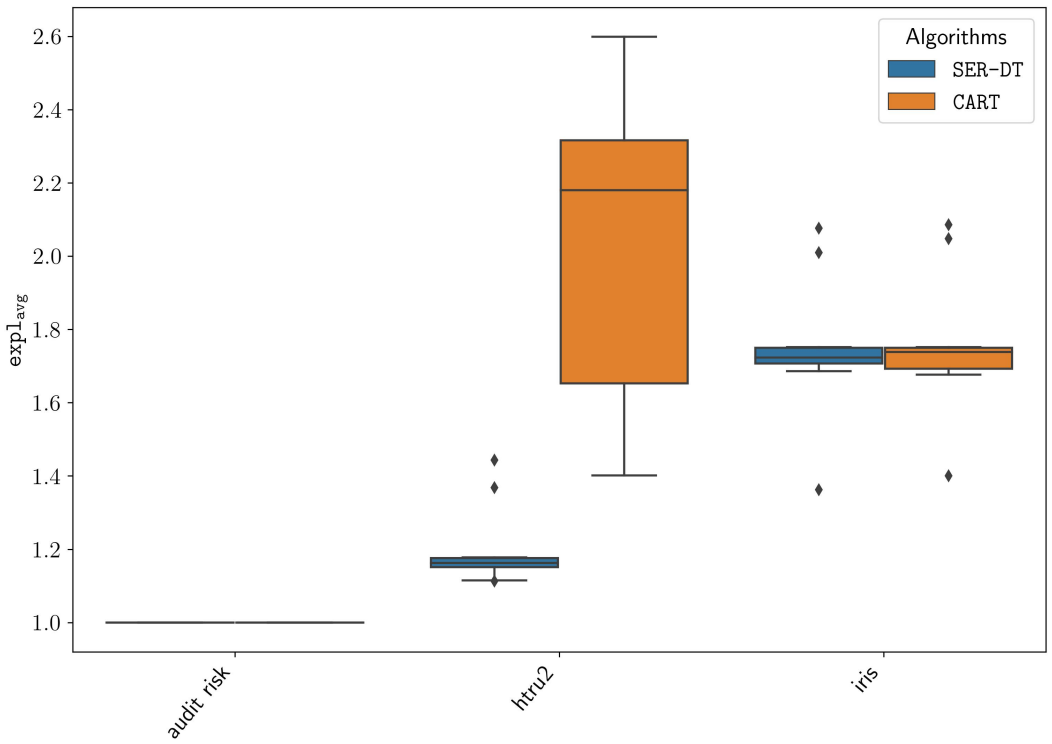


Figura A.8: $expl_{avg}$ para CART e SER-DT para alguns *datasets*.

PUC-Rio - Certificação Digital Nº 2112508/CA