Análise Comparativa de Implementações de Algoritmos de Árvores de Decisão para Aplicações no Serviço Público

Vinicius Rogério da Silva[1], Eduardo Corrêa Gonçalves[1]

[1]Escola Nacional de Ciências Estatísticas (ENCE/IBGE)
Rio de Janeiro – RJ – Brasil

vinicius-rogerio@hotmail.com, eduardo.correa@ibge.gov.br

Abstract. *Decision Trees are widely used in the context of public administration where predictions of algorithms are employed to support managers in making decisions that can have a profound effect on people's lives. In this paper, we performed a comparative analysis of three open source implementations in Python and R for two popular decision tree algorithms (CART and C4.5). We compared the models generated by these implementations with respect to predictive performance, training and classification time, and interpretability. The results of this study are intended to contribute to the use of the evaluated implementations in public service as well as in other areas in which the use of interpretable classification models is desirable.*

Resumo. *Árvores de Decisão (ADs) possuem larga utilização na administração pública, setor no qual as previsões baseadas em dados são utilizadas para apoiar os gestores no processo de tomada de decisões que podem ter um efeito profundo nas pessoas. Este trabalho realiza uma análise comparativa de três diferentes implementações de código aberto em Python e R para dois populares algoritmos de aprendizado de ADs (C4.5 e CART). Os modelos gerados foram comparados quanto ao desempenho preditivo, tempo para treinamento e classificação, e interpretabilidade. Objetiva-se que os resultados do estudo forneçam importantes contribuições para a utilização das implementações no serviço público, bem como nas demais áreas nas quais o uso de modelos de classificação interpretáveis seja desejável.*

1. Introdução

O desenvolvimento de estratégias automatizadas para classificação representa um dos tópicos de pesquisa mais ativos na área de ciência de dados [Han et al. 2011]. Neste problema, o objetivo é realizar a associação automática de objetos a classes discretas (também chamadas de "rótulos de classe" ou simplesmente "rótulos") com base nas características dos objetos. Existem muitas aplicações importantes para a tarefa de classificação, variando desde a análise de crédito (classificar clientes que solicitaram um empréstimo bancário de acordo com o risco de crédito "baixo", "médio" ou "alto") até a genômica funcional (determinar as funções biológicas de genes e proteínas).

Nos anos recentes, a maior parte das abordagens para classificação propostas na literatura baseia-se na construção de modelos caixa-preta, através do emprego de técnicas como SVM, *ensembles* e *deep learning* [Rudin 2019]. Estas técnicas foram projetadas para maximizar a acurácia dos classificadores, mas possuem uma desvantagem importante: não podem ser empregadas quando o problema requer a construção de

modelos de classificação interpretáveis. Um classificador interpretável possui a habilidade de "explicar" as suas classificações para os usuários através, por exemplo, da apresentação de regras de classificação no formato: SE <condição> ENTÃO <rótulo(s) de classe> [Freitas 2014, Parmentier and Vidal 2021]. Em alguns domínios de aplicação importantes da classificação, como diagnose médica [Luo et al. 2015], bioinformática [Fabris et al. 2017] e problemas no âmbito jurídico-criminal [Zeng et al. 2017], a habilidade de interpretar o resultado de uma classificação representa algo tão importante quanto o desempenho preditivo do modelo. Isto é válido também no contexto da administração pública, setor no qual as previsões baseadas em dados são normalmente utilizadas para apoiar os gestores no processo de tomada de decisões que podem ter um efeito profundo nas pessoas [Varshney 2015].

Dentre os diferentes algoritmos para o aprendizado de classificadores interpretáveis [Freitas 2014], destacam-se, por sua popularidade e aplicabilidade, os algoritmos de árvores de decisão – objeto de estudo deste artigo. Há um número significativo de trabalhos recentes que abordam o uso de classificadores baseados em árvores de decisão para resolver problemas do âmbito do serviço público, tais como gestão de projetos [Prado et al. 2015], estudos sobre mobilidade urbana e migração populacional [Nelson and Kennedy 2015, Pérez et al. 2019] e análise de dados de pesquisas sociais e econômicas [Kern et al. 2019]. Sendo assim, o presente trabalho realiza uma análise comparativa de três diferentes implementações de código aberto em Python e R para dois algoritmos de aprendizado de árvores de decisão: C4.5 e CART. As implementações foram comparadas segundo os critérios de desempenho preditivo, desempenho de execução (tempo de treinamento e classificação) e interpretabilidade dos modelos gerados. Acredita-se que este comparativo forneça importantes contribuições para a utilização dessas implementações no âmbito do serviço público e em outras áreas nas quais o uso de modelos de classificação interpretáveis seja um requisito necessário.

O restante do trabalho está dividido da seguinte forma. A Seção 2 apresenta o referencial teórico, cobrindo as características básicas das árvores de decisão e de seus algoritmos de aprendizado. Em seguida, a Seção 3 revisa brevemente trabalhos recentes sobre o uso de árvores de decisão no serviço público. A Seção 4 é a principal do artigo, onde apresenta-se a metodologia do estudo e os resultados do experimento que comparou as diferentes implementações de árvores de decisão. Por fim, na Seção 5 são apresentadas as conclusões do estudo e ideias para trabalhos futuros.

2. Referencial Teórico

A técnica de árvores de decisão (ADs) é uma das mais utilizadas para o aprendizado de classificadores interpretáveis. Isto é justificado pelo fato de as ADs possuírem uma estrutura gráfica e intuitiva, semelhante a um fluxograma, que torna natural a interpretação do modelo de classificação [Pérez et al. 2019]. A Figura 1 mostra um exemplo de AD que classifica um cliente como possível comprador de carro importado com base em sua renda e idade (atributos preditivos). Observe que a AD possui as seguintes propriedades:

- É desenhada com a raiz no topo e as folhas na parte inferior.
- Cada nó interno representa um teste sobre um atributo preditivo (neste caso, os atributos "Renda" e "Idade").

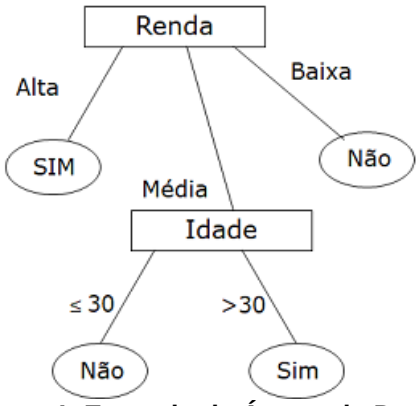

Figura 1. Exemplo de Árvore de Decisão

- Uma ramificação, partindo de um nó interno, representa um resultado para o teste (por exemplo, Idade ≤ 30)
- Cada nó folha representa um rótulo classe (neste caso, Comprador = "Sim" ou Comprador = "Não").
- Um novo objeto é classificado seguindo um caminho na árvore, do nó raiz (neste exemplo, o nó "Renda") até um nó folha.

Uma AD é formada por um conjunto de regras de classificação, uma vez que existe sempre um único caminho da raiz para cada folha e este caminho representa uma expressão da regra utilizada para classificar um objeto. Folhas diferentes podem produzir a mesma classificação, mas cada uma por uma razão diferente (pois cada caminho representa uma regra diferente). Por exemplo, a AD da Figura 1 é composta por quatro regras:

- SE (Renda = 'Alta') ENTÃO (Comprador = 'Sim')
- SE (Renda = 'Média') e (Idade ≤ 30) ENTÃO (Comprador = 'Não')
- SE (Renda = 'Média') e (Idade > 30) ENTÃO (Comprador = 'Sim')
- SE (Renda = 'Baixa') ENTÃO (Comprador = 'Não')

Os algoritmos CART [Breiman et al. 1984] e C4.5 [Quinlan 1993] são os mais conhecidos e utilizados para o aprendizado de ADs, possuindo implementações em código aberto em diferentes linguagens de programação, como R e Python [Kuhn 2020, Pedregosa et al. 2011, Therneau et al. 2019]. Ambos utilizam estratégias parecidas: são recursivos, constroem a árvore utilizando uma abordagem *top-down* e possuem como meta a construção de árvores que possuam o menor tamanho e a maior acurácia possíveis. O algoritmo CART produz apenas árvores binárias (cada nó pode ter no máximo dois ramos), enquanto o C4.5 é capaz de gerar árvores n-árias, como a da Figura 1.

As ADs geradas pelo C4.5, CART e outros algoritmos são compostas por uma série de questões. A resposta da primeira questão determina a questão seguinte, e assim sucessivamente até que um nó folha seja alcançado. Caso as questões sejam bem formuladas (na melhor ordem possível), um pequeno número delas poderá ser suficiente para classificar corretamente um objeto. Portanto, um aspecto chave para a construção de uma AD consiste na estratégia para a escolha dos atributos preditivos que estarão mais próximos da raiz da árvore (ou seja, os atributos que são inicialmente avaliados para determinar a classe à qual um objeto pertence). Medidas de diversidade como a entropia (adotada pelo C4.5) e o índice de Gini (adotada pelo CART) são utilizadas tratar este

problema. Detalhes sobre essas medidas e sobre como elas são utilizadas pelos algoritmos C4.5 e CART podem ser obtidos em [Han et al. 2011].

3. Trabalhos Relacionados

Esta seção realiza uma revisão de trabalhos recentes que empregam ADs em problemas relacionados ao serviço público. Nestes trabalhos, o modelo de classificação de AD foi empregado não apenas com a finalidade preditiva (previsão do rótulo de classe de novos objetos), mas também com a finalidade descritiva (explicar as características dos objetos que pertencem a cada classe).

Em [Prado et al. 2015], os autores exploraram o uso de ADs geradas pelo algoritmo C4.5 para avaliar o sucesso ou fracasso de projetos no âmbito da administração pública. Foram analisados os orçamentos e os prazos, segundo critérios e boas práticas de gerenciamento de projetos. A base de dados utilizada foi um recorte de dados disponíveis na prefeitura de uma cidade turística do interior do Brasil. Os resultados obtidos, embora preliminares, demonstraram um grande potencial de utilização de ADs para descoberta de conhecimento a partir das características dos projetos e suas respectivas execuções.

O estudo de [Pérez et al. 2019] foca na influência de variáveis socioeconômicas na mobilidade urbana. Utilizando dados de pesquisas realizadas pela Companhia do Metropolitano de São Paulo e explorando características como idade, renda, sexo e grau de instrução, os autores utilizaram ADs geradas com o algoritmo C4.5 com o intuito de identificar se essas variáveis conseguem explicar padrões em viagens por motivo de trabalho e educação. Dessa forma, o trabalho fornece uma contribuição à administração pública no que tange ao planejamento das dinâmicas da mobilidade.

[Nelson and Kennedy 2015] discutem uma reprodução simulada de padrões de migração doméstica a partir de dados do Censo colombiano. Utilizando ADs produzidas com o uso do algoritmo CART, os autores procuram entender quais são os principais motivos para a migração, além dos fatores que contribuem para a escolha do destino. Alguns padrões foram identificados, como o de que a população de migrantes demonstra maior escolaridade e mais poder aquisitivo em comparação com os não-migrantes, além de migrações por motivos de estudo, trabalho e saúde estarem mais concentradas nas elites econômicas.

[Kern et al. 2019] discute a aplicação do CART e outros algoritmos baseados em ADs no contexto das metodologias adotadas por institutos de pesquisa como forma de identificar e mitigar alguns problemas comuns. A aplicação mais tradicional, segundo os autores, se dá na predição de não-resposta, na qual, em geral, a propensão à resposta pode ser estimada baseada na quantidade de respondentes em um determinado nó da árvore. O trabalho explora, ainda, a utilização de *ensembles* (métodos que combinam as decisões de várias ADs), como as Random Forests [Vidal and Shiffer 2020], que também se mostraram efetivas na predição da não-resposta.

O relatório de [Thapa 2019] trata de aspectos relacionados a Governança Eletrônica no uso de Inteligência Artificial. Dentre outros pontos importantes, destaca os desafios na utilização de algoritmos de IA no contexto das democracias liberais. O trabalho pontua que, em comparação com algoritmos mais complexos, as árvores de decisão podem ser traduzidas em estruturas compreensíveis para os seres humanos, o que representa uma vantagem em termos de confiança e transparência.

4. Experimento

4.1. Metodologia

O experimento reportado neste artigo fez uso de doze bases de dados com número de atributos e objetos variado, sendo quatro delas provenientes de pesquisas realizadas no âmbito do serviço público (adult, cnae, pme e pof). Embora as demais bases não envolvam especificamente problemas da administração pública, elas representam outros tipos de aplicação em que a interpretabilidade do modelo também é um requisito importante, tais como análise de crédito, marketing, biologia e medicina. As características das bases são resumidas na Tabela 1. Nesta tabela, as colunas "N", "atribs" e "classes" indicam, respectivamente, o número de objetos, atributos preditivos e rótulos de classe de cada base. As bases 1 a 3 são provenientes de [Kaggle 2021], 4 a 10 de [Dua and Graff 2021] e 11 e 12 foram cedidas, respectivamente, pelo IBGE e FGV.

Tabela 1. Descrição das bases de dados utilizadas no experimento

	base de dados	descrição do problema de classificação	N	atribs.	classes
1	credit_data	Classificar clientes quanto à obtenção de crédito usando o seu histórico financeiro.	2.000	4	2
2	diabetes	Identificar portadores de diabetes a partir de dados de seus exames clínicos.	2.000	8	2
3	online shoppers intention	Identificar intenções de compra em uma loja online usando os dados de navegação dos clientes.	12.330	17	2
4	spambase	Classificação de e-mail como spam ou normal com base no texto de seu corpo.	4.601	57	2
5	audit_risk	Previsão do risco de fraude em empresas com base em informações de fatores de risco.	776	25	2
6	bank	Identificar clientes com potencial para aplicações financeiras em um banco.	45.211	16	2
7	caesarean	Classificar o tipo de parto como cesariana ou normal usando dados da gestante.	80	5	2
8	mushrooms	Classificar cogumelos como venenosos ou comestíveis de acordo com as suas características físicas	8.124	21	2
9	adult	Identificar a classe de renda de um indivíduo a partir de dados censitários (Censo de Washington)	48.842	14	2
10	cnae_9	Classificar a atividade econômica de uma empresa a partir do texto de seu contrato social (CNAE/IBGE).	1.080	856	9
11	pme	Classificar o tipo de ocupação de um indivíduo a partir de seus dados demográficos (PME/IBGE)	7.741	4	5
12	pof_16	Inferir a cidade de residência de uma família a partir de sua cesta de compras mensais (POF/FGV)	904	16	5

A avaliação do desempenho preditivo dos modelos foi feita com o uso do método *holdout* [Japkowicz and Shah 2011], que consiste em dividir a base de dados em duas partições: treino e teste. A partição de treino é utilizada para gerar a AD, enquanto a partição de teste fica reservada para estimar o seu desempenho preditivo. No presente trabalho, a divisão foi feita respeitando a fração de dois terços dos dados para treinamento e um terço para teste. Algumas das bases já se encontravam particionadas quando coletadas nos repositórios e, neste caso, estas partições pré-determinadas foram utilizadas.

Quatro diferentes medidas de desemprenho preditivo foram empregadas: Precisão (Pr), Recall (Re), F-Measure (FM) e Acurácia (Ac), respectivamente apresentadas nas Equações (1), (2), (3) e (4) [Han et al. 2011]. Nas fórmulas apresentadas, VP (Verdadeiros Positivos) consiste no número de objetos positivos (objetos da classe de interesse na partição de teste) que foram corretamente classificados pelo algoritmo, FP (Falsos Positivos) é o número de objetos negativos que foram incorretamente classificados como

positivos, FN (Falsos Negativos) é o número de objetos positivos incorretamente classificados como negativos, e VN (Verdadeiro Negativos) é o número de objetos negativos que foram corretamente classificados.

$$Pr = \frac{VP}{VP + FP} \tag{1}$$

$$Re = \frac{VP}{VP + FN} \tag{2}$$

$$FM = 2 \times \frac{Pr \times Re}{Pr + Re} \tag{3}$$

$$Ac = \frac{VP + VN}{VP + FP + FN + VN} \tag{4}$$

Para verificar a significância estatística dos resultados, utilizou-se o Teste de Wilcoxon [Japkowicz and Shah 2011], um teste não paramétrico que pode ser utilizado para comparar pares de classificadores em múltiplos domínios (bases de dados). Os resultados foram verificados ao nível de 5% de significância.

4.2. Experimento

As implementações avaliadas no experimento são as disponibilizadas pelos pacotes C5.0 [Kuhn 2020] e rpart [Therneau et al. 2019], da linguagem R, e o pacote scikit-learn [Pedregosa et al. 2011], da linguagem Python. Todas consistem em implementações de código aberto (*open source*), sendo que o pacote C5.0 implementa uma adaptação do algoritmo C4.5, enquanto o rpart oferece uma implementação do CART. Na scikit-learn, as ADs são também geradas com o uso do algoritmo CART, mas permitem a utilização tanto do índice de Gini como da entropia como medida de diversidade.

Os experimentos foram executados utilizando-se os parâmetros *default* dos pacotes, exceto pelo fato de que na scikit-learn o parâmetro "random_state=0" foi empregado para possibilitar um comportamento determinístico a cada execução e "criterion= 'entropy'" e "criterion= 'gini'" foram utilizados para a realizar a geração do modelo de classificação com o emprego das medidas da entropia e do índice de Gini, respectivamente. A seguir, são reportados os resultados das avaliações de desempenho preditivo, desempenho de execução e interpretabilidade dos modelos.

Análise 1: Desempenho Preditivo

A Tabela 2 apresenta os resultados obtidos para as medidas de Precisão (Pr) e Recall (Re). Os melhores resultados para cada base estão sublinhados. É possível observar que o pacote C5.0 obteve o melhor desempenho médio para a medida de Recall enquanto a scikit (gini) obteve o melhor desempenho médio para a Precision. De maneira oposta, o pacote rpart gerou os modelos com o pior desempenho médio para ambas as medidas. Com relação à significância estatística dos resultados, o teste de Wilcoxon indicou que: (i) para a medida de Recall, nenhuma diferença significativa foi encontrada entre os pacotes; (ii) para a medida de Precisão, o C5.0 apresentou diferença estatisticamente significativa em relação ao rpart (com $p = 0{,}049$), porém o mesmo não se verificou na

comparação feita com a implementação do algoritmo CART oferecida pela scikit-learn (seja utilizando a entropia ou o índice de Gini como medida de diversidade).

A Tabela 3 apresenta os resultados obtidos para as medidas de F-Measure (FM) e Acurácia (Ac). Neste caso, o pacote C5.0 obteve o melhor resultado para a maioria das bases, além do melhor desempenho médio para as duas medidas. A avaliação da significância estatística indicou que o pacote C5.0 se mostrou estatisticamente superior ao pacote rpart tanto para a F-Measure como para a Acurácia ($p = 0,027$ e $p = 0,037$, respectivamente). Entretanto, não foi encontrada diferença significativa entre C5.0 e a scikit-learn com o uso da entropia ou do índice de Gini.

Em resumo, os resultados do teste de Wilcoxon indicam que não há evidência de que haja diferença de desempenho preditivo entre os pacotes C5.0 e scikit-learn considerando as quatro medidas de desempenho utilizadas e que o C5.0 possui desempenho superior ao do pacote rpart para três das quatro medidas (Precision, F-Measure e Acurácia).

Tabela 2. Desempenho dos algoritmos para as medidas Precision e Recall

base de dados	C5.0		rpart		scikit (entropia)		scikit (gini)	
	Pr	**Re**	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**
credit_data	0,9879	0,9948	0,9773	0,9773	0,9879	0,9948	0,9878	0,9861
diabetes	0,8333	0,8577	0,7402	0,6318	0,9016	0,9205	0,9139	0,9331
online shoppers	0,9393	0,9500	0,9331	0,9535	0,9252	0,9093	0,9261	0,9099
spambase	0,8950	0,9133	0,8743	0,8401	0,8939	0,9030	0,8671	0,8878
audit_risk	0,9907	1,0000	0,9907	1,0000	0,9817	1,0000	0,9817	1,0000
bank	0,9305	0,9609	0,9249	0,9660	0,9282	0,9269	0,9284	0,9290
caesarean	0,5000	0,8333	0,3636	0,3333	0,3889	0,5833	0,4375	0,5833
mushrooms	1,0000	1,0000	0,9878	1,0000	1,0000	1,0000	1,0000	1,0000
adult	0,8990	0,9325	0,8613	0,9493	0,8778	0,8773	0,8782	0,8753
cnae_9	0,8824	0,8521	0,8462	0,7295	0,8550	0,8457	0,8831	0,8742
pme	0,2274	0,2917	0,2274	0,2917	0,3050	0,2789	0,3046	0,2788
pof_16	0,4032	0,4204	0,4837	0,4437	0,4418	0,4522	0,4485	0,4597
Média	0,7907	0,8339	0,7676	0,7597	0,7906	0,8077	0,7964	0,8098

Tabela 3. Desempenho dos algoritmos para a medidas F-Measure e Acurácia

base de dados	C5.0		rpart		scikit (entropia)		scikit (gini)	
	FM	**Ac**	**FM**	**Ac**	**FM**	**Ac**	**FM**	**Ac**
credit_data	0,9913	0,9850	0,9774	0,9610	0,9913	0,9850	0,9869	0,9775
diabetes	0,8454	0,8876	0,6817	0,7886	0,9110	0,9355	0,9234	0,9445
online shoppers	0,9446	0,9056	0,9432	0,9026	0,9172	0,8608	0,9179	0,8620
spambase	0,9040	0,9257	0,8569	0,8924	0,8985	0,9218	0,8773	0,9048
audit_risk	0,9953	0,9961	0,9953	0,9961	0,9907	0,9922	0,9907	0,9922
bank	0,9455	0,9023	0,9450	0,9009	0,9275	0,8724	0,9287	0,8743
caesarean	0,6250	0,5556	0,3478	0,4444	0,4667	0,4074	0,5000	0,4815
mushrooms	1,0000	1,0000	0,9938	0,9933	1,0000	1,0000	1,0000	1,0000
adult	0,9154	0,8684	0,9031	0,8445	0,8776	0,8130	0,8767	0,8120
cnae_9	0,8579	0,8500	0,7482	0,7278	0,8481	0,8444	0,8766	0,8722
pme	0,2551	0,6575	0,2551	0,6575	0,2783	0,6470	0,2781	0,6466
pof_16	0,4045	0,4180	0,4235	0,4515	0,4420	0,4515	0,4465	0,4548
Média	0,8070	0,8293	0,7559	0,7967	0,7957	0,8109	0,8002	0,8185

Análise 2: Desempenho de Execução

O desempenho de execução das implementações foi avaliado no que diz respeito ao tempo para o treinamento e teste dos modelos de cada base de dados. O experimento foi realizado em um PC com 8GB de RAM, S.O. Windows Home 10 e CPU Intel i5-8265U. Os *boxplots* da Figura 2 apresentam a distribuição dos tempos para os pacotes analisados.

Percebe-se que a implementação do algoritmo C5.0 no R foi, de modo geral, a mais lenta, tanto no treinamento (gráfico à esquerda na Figura 2) como na classificação/teste (gráfico à direita). Já a scikit-learn mostrou-se mais eficiente. Esse resultado possivelmente está relacionado ao fato de que a scikit-learn trabalha apenas com atributos numéricos, podendo assim organizar bases de dados em matrizes NumPy [Harris et al. 2020] – estrutura de dados mais eficiente do que o *data frame* adotado pelo pacote C5.0. Apesar desse notável ganho no desempenho de execução, a scikit-learn impõe dificuldades na interpretação dos modelos gerados, como será discutido a seguir.

Análise 3: Interpretabilidade do Modelo

Uma desvantagem do pacote scikit-learn em relação aos demais encontra-se no fato de que ele não suporta atributos preditivos categóricos, exigindo com que o usuário os converta para o tipo numérico antes que o modelo possa ser treinado. Isso é necessário para que seja possível estruturar a base a ser analisada em uma matriz numérica NumPy. O processo de conversão requer que cada categoria seja associada a um inteiro (por exemplo, os valores 'F' e 'M' do atributo "sexo" devem ser transformados em 0 e 1).

Pode-se destacar dois pontos negativos que resultam dessa conversão. O primeiro está na maior dificuldade de utilização, já que a conversão é uma etapa extra que não é exigida pelos pacotes rpart e C5.0. O segundo e principal ponto diz respeito ao prejuízo para a interpretabilidade do modelo, visto que a AD gerada perde a representação das categorias dos atributos preditivos, como vê-se na subárvore da base de dados da "pme" apresentada na Figura 3. Observe que esta AD gerada pela scikit-learn obriga o usuário a conhecer os códigos numéricos que foram associados às categorias dos atributos preditivos para que seja possível compreender, por exemplo, qual é o significado de um teste como "sexo <= 0.50" (neste caso, se as categorias 'F' e 'M' foram associadas aos valores 0 e 1, respectivamente, "sexo <= 0.50" na verdade significa "sexo = F").

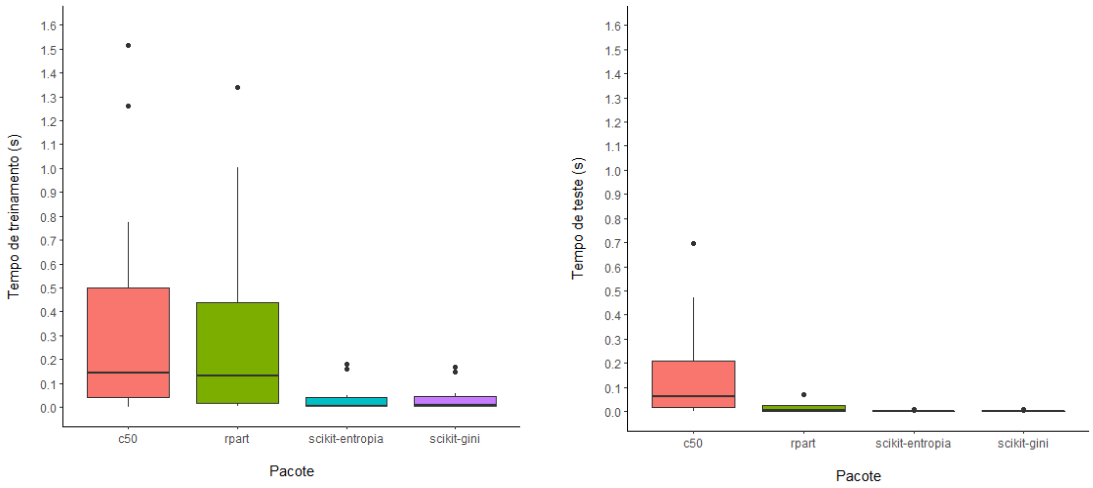

Figura 2. Distribuição dos tempos de treinamento e teste, por pacote

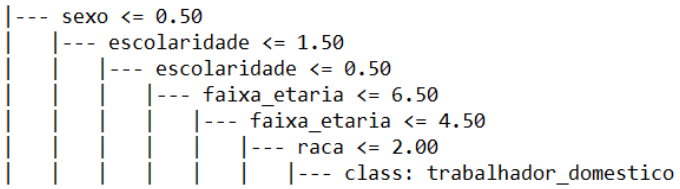

Figura 3. Subárvore da base da "pme" gerada pela scikit-learn (entropia)

5. Conclusão

Este trabalho reportou os resultados de um experimento em que foi realizada a análise comparativa de três diferentes implementações de código aberto para os algoritmos de árvores de decisão C4.5 e CART. Estes algoritmos são largamente utilizados em problemas no âmbito do serviço público, pois neste tipo de problema a possibilidade de interpretar o resultado de uma classificação costuma representar algo tão importante quanto o desempenho preditivo do modelo.

As implementações avaliadas neste trabalho são disponibilizadas pelos pacotes C5.0 e rpart da linguagem R e pelo pacote scikit-learn da linguagem Python. O pacote C5.0 oferece uma implementação baseada no algoritmo C4.5, enquanto os pacotes rpart e scikit-learn oferecem implementações que têm por base o algoritmo CART. Os três pacotes foram comparados no que diz respeito ao desempenho preditivo, desempenho de execução (tempo para treinamento e classificação) e interpretabilidade dos modelos gerados. Os resultados obtidos indicaram que: (i) não foi encontrada evidência estatística de que os pacotes C5.0 do ambiente R e scikit-learn do Python possuam diferença de performance preditiva; (ii) embora a scikit-learn tenha apresentado tempos mais baixos para treinamento e classificação, o pacote C5.0 oferece benefícios em termos da interpretabilidade dos modelos de AD produzidos, requisito que é considerado de grande relevância para aplicações no âmbito da administração pública.

Espera-se que os resultados do estudo forneçam importantes contribuições para a utilização das implementações não apenas em problemas relacionados ao serviço público, como também em outras áreas em que o uso de modelos de classificação interpretáveis seja desejável. Como trabalho futuro, pretende-se explorar os parâmetros das implementações com o uso de estratégias de AutoML [Feurer et al. 2015]. Adicionalmente, deseja-se avaliar a técnica recentemente proposta em [Vidal and Shiffer 2020], que torna possível transformar uma Random Forest em uma única árvore de decisão com o mesmo poder preditivo.

Referências

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984), Classification and regression trees, Taylor & Francis, 1st edition.

Dua, D. and Graff, C. (2021). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Acessado em 15/09/2021.

Fabris, F., Magalhães, J. P. and Freitas, A. A. (2017). A review of supervised machine learning applied to ageing research. In *Biogerontology*, 18, n. 2, pages. 171–188.

Feurer et al. (2015). "Efficient and Robust Automated Machine Learning", In: 28th Int'l Conf. on Neural Information Processing Systems (NIPS 2015), p.2755–2763.

Freitas, A. A. (2014). Comprehensible classification models: a position paper. In *SIGKDD Explor. Newsl*. 15, pages 1–10. ACM.

Han, J., Kamber, M., and Pei, J. (2011), Data mining: Concepts and techniques, Morgan Kaufmann Publishers, 3rd edition.

Harris et al. (2020). Array programming with NumPy. In *Nature* 585, pages 357–362.

Japkowicz, N. and Shah, M., Evaluating Learning Algorithms: A Classification Perspective, Cambridge University Press, 2011.

Kaggle. https://www.kaggle.com. Acessado em 15/09/2021.

Kern, C., Klausch, T., and Kreuter, F. (2019). Tree-based machine learning methods for survey research. In *Survey Research Methods*, 13, Issue 1, pages 73–93. ESRA.

Kuhn, M. (2020) "C5.0: C5.0 Decision Trees and Rule-Based Models", https://CRAN.R-project.org/package=C5.0.

Luo, G. et al. (2015). A systematic review of predictive models for asthma development in children. In *BMC Med Inform Decis Mak*,. 15(1):99.

Nelson, J. B., Kennedy, W. G., and Greenberg, A. M. (2015). "Agents and Decision Trees for Microdata". In: 24th BRiMS.

Parmentier, A. and Vidal, T. (2021). "OCEAN: Optimal Counterfactual Explanations in Tree Ensembles", In: 38th Int'l Conf. on Machine Learning (ICML 2021).

Pedregosa et al. (2011). Scikit-learn: Machine learning in python. In *JMLR* 12, pages 2825–2830.

Pérez et al. (2019). "Análise de Mudanças em Fatores Socioeconômicos Baseado em Árvore de Decisão para o Estudo de Viagens por Motivos Trabalho e Estudo na Região Metropolitana de São Paulo", In: 51º SBPO, SOBRAPO, p.399–406.

Prado, C. R., Peres, S. M., and Fantinato, M. (2015). "Tomada de Decisão na Administração Pública Apoiada pela Descoberta de Conhecimento: Um Estudo de Caso em Gestão de Projetos", In: XI SBSI, SBC, p.399–406.

Quinlan, J. (1993), C4.5: Programs for machine learning, Morgan Kaufmann.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. In *Nat Mach Intell,* 1, pages 206–215.

Thapa, B. E. P. (2019). Artificial intelligence, big data and algorithmic decision-making in government from a liberal perspective, ELF.

Therneau, T., Atkinson, B., and Ripley, B. (2019) "rpart: Recursive Partitioning and Regression Trees". https://CRAN.R-project.org/package=rpart.

Varshney, K. R. (2015). "Data Science of the People, for the People, by the People: A Viewpoint on an Emerging Dichotomy", In: D4GX 2015, Bloomberg, p. 1–6.

Vidal, T. and Schiffer, M. (2020). "Born-Again Tree Ensembles", In: 37th Int'l Conf. on Machine Learning (ICML 2020), p.9743–9753.

Zeng, J., Ustun, B. and Rudin, C. (2017). Interpretable classification models for recidivism prediction. In *J. R. Stat. Soc. A*, 180, pages 689–722. Royal Statistical Society.