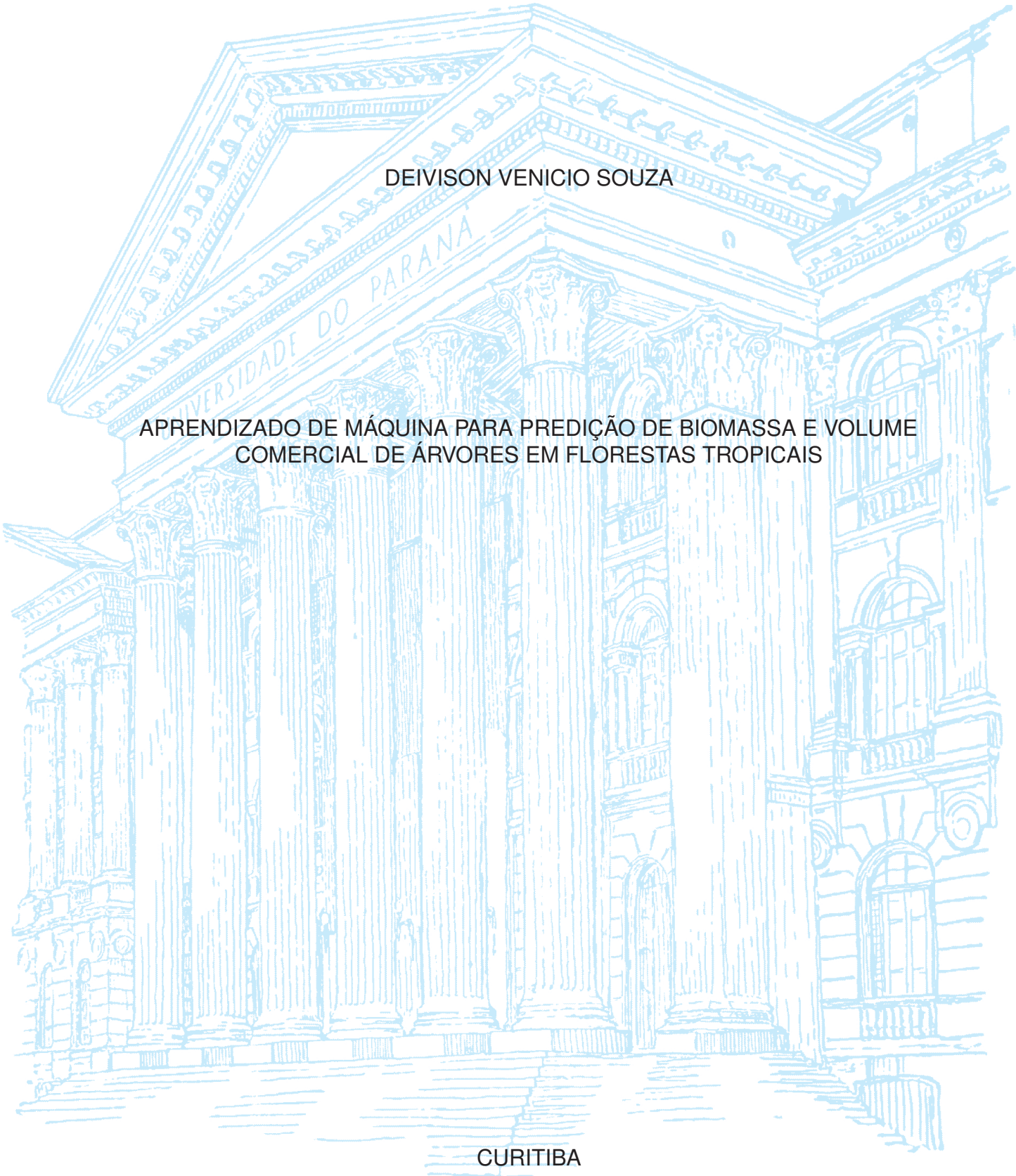
UNIVERSIDADE FEDERAL DO PARANÁ

DEIVISON VENICIO SOUZA



APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DE BIOMASSA E VOLUME COMERCIAL DE ÁRVORES EM FLORESTAS TROPICAIS

CURITIBA

2020

DEIVISON VENICIO SOUZA

APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DE BIOMASSA E VOLUME COMERCIAL DE ÁRVORES EM FLORESTAS TROPICAIS

Tese apresentada ao Programa de Pós-Graduação em Engenharia Florestal, do Setor de Ciências Agrárias, da Universidade Federal do Paraná, como requisito parcial à obtenção do título de Doutor em Engenharia Florestal.

Orientador: Prof. Carlos Roberto Sanquetta, Dr.
Coorientador: Prof. Júlio César Nievola, Dr.
Coorientadora: Profª Ana Paula Dalla Corte, Drª.

CURITIBA

2020

Ficha catalográfica elaborada pela
Biblioteca de Ciências Florestais e da Madeira - UFPR

Souza, Deivison Venicio

Aprendizado de máquina para predição de biomassa e volume comercial de árvores em florestas tropicais / Deivison Venicio Souza. – Curitiba, 2020.

171 f. : il.

Orientador: Prof. Dr. Carlos Roberto Sanquetta
Coorientadores: Prof. Dr. Júlio César Nievola
Profa. Dra. Ana Paula Dalla Corte

Tese (Doutorado) - Universidade Federal do Paraná, Setor de Ciências Agrárias, Programa de Pós-Graduação em Engenharia Florestal. Defesa: Curitiba, 24/01/2020.
Área de concentração: Manejo Florestal.

1. Florestas - Medição. 2. Biomassa - Medição. 3. Árvores - Medição. 4. Mineração de dados (Computação). 5. Algoritmos. 6. Teses. I. Sanquetta, Carlos Roberto. II. Nievola, Júlio César. III. Dalla Corte, Ana Paula. IV. Universidade Federal do Paraná, Setor de Ciências Agrárias. V. Título.

CDD – 634.9
CDU – 634.0.51

Bibliotecária: Berenice Rodrigues Ferreira – CRB 9/1160

MINISTÉRIO DA EDUCAÇÃO
SETOR DE CIENCIAS AGRARIAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO ENGENHARIA
FLORESTAL - 40001016015P0

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em ENGENHARIA FLORESTAL da Universidade Federal do Paraná foram convocados para realizar a arguição da tese de Doutorado de **DEIVISON VENICIO SOUZA** intitulada: **APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DE BIOMASSA E VOLUME COMERCIAL DE ÁRVORES EM FLORESTAS TROPICAIS**, sob orientação do Prof. Dr. CARLOS ROBERTO SANQUETTA, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de doutor está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 24 de Janeiro de 2020.

CARLOS ROBERTO SANQUETTA
Presidente da Banca Examinadora

RAZER ANTHOM NIZER ROJAS MONTAÑO
Avaliador Externo (UNIVERSIDADE FEDERAL DO PARANÁ)

JULIO CÉSAR NIEVOLA
Avaliador Externo (PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ)

ALEXANDRE BEHLING
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

JAIME WOJCIECHOWSKI
Avaliador Externo (UNIVERSIDADE FEDERAL DO PARANÁ)

À minha mãe Maria do Carmo Souza (In memoriam).
À minha mãe de criação Ana Lobato Monteiro (In memoriam).
À minha tia Floracy Nascimento (In memoriam).

Dedico.

AGRADECIMENTOS

À *Universidade Federal do Paraná - UFPR*, pelo apoio institucional e pela oportunidade de ampliação dos meus conhecimentos. Em especial, agradeço aos professores do *Programa de Pós-graduação em Engenharia Florestal* que muito contribuíram para o conhecimento adquirido.

Ao *Conselho de Desenvolvimento Científico e Tecnológico* – CNPq, pela concessão de bolsas de estudos.

Ao meu orientador, *Professor Dr. Carlos Roberto Sanquetta*, pelo total apoio, liberdade, confiança e contribuições ao desenvolvimento desta pesquisa.

À minha coorientadora, *Professora Drª. Ana Paula Dalla Corte*, pela orientação e discussões e, principalmente, por acreditar no meu potencial e oportunizar um grande crescimento profissional.

Ao meu coorientador, *Professor Dr. Júlio César Nievola*, da Pontifícia Universidade Católica do Paraná - PUCPR, pela oportunidade de aprendizado, conversas e conhecimentos transferidos, sobretudo acerca da temática principal desta Tese.

À *Joielan Xipaia dos Santos*, minha amada, por todo o amor e companheirismo ao longo de toda essa jornada.

Ao casal de amigos *Maria Emília Martins Ferreira* e *Paulo Afonso Bracarense Costa* pela grande amizade, conversas, trocas de experiências e pelas sensacionais e inesquecíveis aventuras gastronômicas. Ao Paulo Afonso, em especial, destaco o grande estatístico, e agradeço muitíssimo as notáveis e engrandecedoras discussões sobre teoria estatística clássica, e orientações em tomadas de decisões sobre a minha pesquisa científica.

Aos amigos cientistas do *Centro de Excelência em Pesquisas sobre Fixação de Biomassa* - BIOFIX. Em especial, ao *Vinicius Morais Coutinho*, *Myrcia Minatti*, *Aurélio Lourenço Rodrigues*, *Rafael Schmitz*, *Thiago Wendling Gonçalves de Oliveira* e *Luani Rosa de Oliveira Piva*.

À amiga *Linamara Smaniotto Ferrari*, pessoa maravilhosa e transparente, pela grande amizade, convívio, conversas e conhecimentos compartilhados.

À amiga *Helena Cristina Vieira* pela amizade, parceria e pelos conhecimentos e momentos compartilhados.

À amiga *Tawani Lorena Naide*, minha tradutora oficial, pela amizade e momentos compartilhados.

À amiga *Ângela Maria Stüpp* pela amizade e momentos compartilhados.

Aos demais amigos conquistados do Laboratório de Anatomia e Qualidade da Madeira (LANAQM). Em especial, agradeço a amizade de *Eliane Lopes da Silva* e *Cibelle Amaral Reis*.

À *Professora Drª. Silvana Nisgoski* pela oportunidade e disponibilidade de base de imagens macroscópicas de madeiras do Laboratório de Anatomia e Qualidade da Madeira (LANAQM), que permitiram a compreensão e o aprofundamento do conheci-

mento sobre a aplicação de técnicas de aprendizado de máquina no reconhecimento de espécies florestais.

Aos casal de amigos *Mara Rúbia Monteiro* e *Edir Dumaszak* pela amizade, acolhimento, e todas as aventuras e momentos maravilhosos compartilhados.

Ao amigo *Alexandre Doral Marques*, estatístico, e colega do curso de Especialização em Data Science & Big Data da Universidade Federal do Paraná, pelas conversas e discussões estatísticas, que muito agregaram à minha pesquisa científica.

“The model ("empirical" or "theoretical") is only a hypothetical conjecture that might or might not summarize and explain important features of the data. But while it is true that all models are wrong and some are useful, it is equally true that no model is universally useful.“

(George E. P. Box (1978) - Statistics for experimenters, p. 208)

RESUMO

No campo da mensuração florestal, encontrar modelos acurados para predizer variáveis biométricas difíceis de determinar diretamente em campo e de alto custo operacional em inventários florestais, sempre constituiu um grande interesse das pesquisas. Em florestas naturais inequiâneas, em particular, a elevada heterogeneidade das variáveis biométricas é uma condição marcante e intrínseca, e que torna a modelagem preditiva um grande desafio. Devido a isso, por vezes, métodos tradicionais, como a regressão linear clássica, não são capazes de modelar adequadamente a natureza. Assim, o uso de métodos mais flexíveis e com boa capacidade de descrever a realidade, como as técnicas de aprendizado de máquina, ganham forte apelo quando o intuito é melhorar a acurácia dos modelos de biomassa e volume de árvores em florestas naturais inequiâneas. O principal objetivo desta pesquisa foi estudar e comparar o potencial de técnicas de aprendizado de máquina na melhoria das estimativas da biomassa aérea total (BAT) e volume comercial com casca de árvores individuais, frente à abordagem de modelagem por regressão tradicional, a partir de dados coletados em diversos sítios de florestas naturais inequiâneas. Para além disso, objetivou-se usar abordagens recentes para prover algum nível de interpretação para os modelos algorítmicos, desmistificando a metáfora da "caixa preta", e também desenvolver aplicações web para disponibilizar os modelos de aprendizado de máquina mais acurados. Dois estudos de casos foram conduzidos. No primeiro, foi usada uma base global compilada contendo dados de biomassa aérea total de 4004 árvores-amostras (diâmetro $\geq$ 5cm) colhidas e distribuídas em 58 sítios de diferentes países. No segundo, foi usado um conjunto de dados com volume comercial com casca de 13831 árvores (diâmetro $\geq$ 50cm) pertencentes à 38 espécies florestais manejadas na Amazônia brasileira. Para fins de modelagem preditiva, cada conjunto de dados original foi dividido em dados de treinamento (80%) e teste (20%). O método *k*-fold cross-validation foi usado para obter estimativas imparciais de desempenho para os modelos de aprendizado de máquina (MAM). O conjunto de teste foi reservado para uma comparação mais apropriada da precisão das abordagens de regressão tradicional e aprendizado de máquina. Em ambos os estudos de caso, foram consideradas nove técnicas de aprendizado de máquina. Para a modelagem tradicional da BAT, apenas a forma funcional do "Modelo Pantropical-MP" foi admitida e ajustada no conjunto de treinamento, e designada "Modelo Alternativo - MA". Para a modelagem volumétrica tradicional foram admitidas dez formas funcionais usuais na Mensuração Florestal. Na modelagem da biomassa aérea total, os melhores MAM construídos apresentaram rRMSE variando entre 7,30% e 7,98% na validação cruzada, sendo uma rede neural artificial do tipo MLP (size = 9; decay = 0,2) a mais acurada. No conjunto de teste, os MAM e MA apresentaram semelhança na distribuição residual e no desempenho médio para predizer a variável resposta em amostras independentes. Na modelagem do volume comercial com casca, os melhores MAM foram obtidos usando um espaço de recursos dependente do diâmetro e altura, com rRMSE variando entre 22,83% e 24,47%, sendo um modelo de regressão por vetores de suporte com função kernel radial (sigma = 0,003; C = 128) o mais acurado. Os modelos tradicionais genéricos de dupla entrada com resposta logarítmica (Spurr e Schumacher-Hall) apresentaram menor erro padrão residual, mas forte heterocedasticidade. A heterocedasticidade nos MAM aprendidos usando um espaço de recursos dependente de diâmetro e altura da árvore, parece ser menos severa do que aquela constatada nos modelos de dupla entrada com resposta logarítmica. A modelagem preditiva usando técnicas de

aprendizado de máquina não proveu a melhoria esperada nas estimativas de volume comercial com casca e biomassa área total em florestas naturais inequiâneas. Apesar disso, o uso de técnicas de aprendizado de máquina parece ser bastante promissor para modelagem de variáveis biométricas, uma vez que conseguiu equiparar-se aos modelos tradicionais que a décadas são usados na Ciência Florestal.

Palavras-chaves: rede neural artificial, aplicação web, engenharia de variáveis, interpretabilidade de modelos de aprendizado de máquina, florestas naturais inequiâneas.

ABSTRACT

In the scope of forest measurement, finding accurate models to predict biometric variables that are difficult to determine directly in the field and costly to operate in forest inventories has always been a major research concern. In natural uneven-aged forests the high heterogeneity of biometric variables is a striking and intrinsic condition that makes predictive modeling a major challenge. Because of this, sometimes the traditional methods such as classical linear regression, are not able to adequately model the nature. Thus, the use of more flexible methods with a good ability to describe reality, such as machine learning techniques, is strongly sought when the aim is to improve the accuracy of biomass and tree volume models in unequal natural forests. The main objective of this research was to study and compare the potential of machine learning techniques to improve estimates of above-ground biomass (AGB) and commercial volume of individual bark trees, compared to the traditional regression modeling approach, based on data collected from various sites of natural uneven-aged forests. In addition, it aimed to use recent approaches to provide some level of interpretation for algorithmic models, demystifying the "black box" metaphor, and also develop web applications to provide the most accurate machine learning models. Two case studies were conducted. In the first, a compiled global database containing total aerial biomass data from 4004 sample trees (diameter $\geq$ 5cm) collected and distributed in 58 sites from different countries was used. In the second, a data set with commercial volume of 13831 bark trees (diameter $\geq$ 50cm) belonging to 38 forest species managed in the Brazilian Amazon rainforest was used. For the purpose of predictive modeling, each original data set was divided into training (80%) and test (20%) data. The k-fold cross-validation method was used to obtain unbiased performance estimates for machine learning models (MLM). The test set has been reserved for a more appropriate comparison of the accuracy of traditional regression and machine learning approaches. In both case studies, nine machine learning algorithms were considered. For traditional AGB modeling, only the functional form of the "Pantropical Model – PM" was admitted and adjusted in the training set, and designated "Alternative Model - AM". Ten usual functional forms were allowed for traditional volumetric modeling in Forest Measurement. In the modeling of the above-ground biomass, the best MLM built presented rRMSE ranging from 7.30% to 7.98% in the cross validation, being an artificial neural network of the MLP type (size = 9; decay = 0.2) the most accurate. In the test set, MLM and AM showed similarity in residual distribution and mean performance to predict the response variable in independent samples. In the bark tree commercial volume modeling, the best MLM were obtained using a diameter and height dependent resource space, with rRMSE ranging from 22.83% to 24.47%, being a support vector regression model with radial kernel function (sigma = 0.003, C = 128) the most accurate. The traditional generic logarithmic double-input models (Spurr and Schumacher-Hall) showed lower residual standard error but strong heteroscedasticity. The heteroscedasticity in MLM learned using a tree diameter and height dependent resource space appears to be less severe than that found in logarithmic response double entry models. Predictive modeling using machine learning techniques did not provide the expected improvement in the accuracy of commercial volume estimates of bark trees and above-ground biomass in natural uneven-aged forests. Nevertheless, the use of machine learning techniques seems to be quite promising for modeling biometric variables, as it has managed to match the traditional models that have been used in forest science for decades.

Key-words: artificial neural network, web application, feature engineering, interpretability of machine learning models, natural uneven-age forests.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E DE SIGLAS

AIC *Akaike Information Criterion*

BAT Biomassa Aérea Total (*Above-Ground Biomass*)

BT *Bagged Trees*

CARET *Classification And Regression Training*

CV Coeficiente de Variação (*Coefficient of Variation*)

DF *Degree of Freedom*

LOESS *Local Polynomial Regression*

M5' Model Tree

MA Modelo Alternativo (*Alternative model*)

MAE *Mean Absolute Error*

MAM Modelos de Aprendizado de Máquina (*Machine Learning Models*)

MELNV Melhores Estimadores Lineares Não-Viesados (*Best Linear Unbiased Estimator*)

MLP *Multilayer Perceptron*

MP Modelo Pantropical (*Pantropical Model*)

MQO Mínimos Quadrados Ordinários (*Ordinary Least Squares*)

MVS Máquinas de Vetores de Suporte (*Support Vector Machines*)

PRESS *Prediction Residual Error Sum of Squares*

RF *Random Forest*

RMSE *Root Mean Square Error*

RNA Redes Neurais Artificiais (*Artificial neural networks*)

RT *Regression Trees*

SGB *Stochastic Gradient Boosting*

SVR *Support Vector Regression* (Regressão por Vetores de Suporte)

VIF *Variance Inflation Factor*

XGBoost *eXtreme Gradient Boosting*

d Diâmetro a 1,30m do solo (*Diameter at Breast Height*)

h Altura total (*Total Height*)

k-NN *k-Nearest-Neighbor*

rRMSE *Relative Root Mean Square Error*

v Volume comercial (*Commercial volume*)

wkNN *Weighted k-Nearest Neighbors* (Vizinho mais Próximo Ponderado)

LISTA DE SÍMBOLOS

$\in$	Simbolo matemático pertence
Δ	Letra grega Delta
Σ	Letra grega Sigma
λ	Letra grega lambda
ρ	Letra grega rho
α	Letra grega alfa
β	Letra grega beta
σ	Letra grega sigma
ϵ	Letra grega épsilon

SUMÁRIO

1 INTRODUÇÃO

No campo da mensuração florestal, estimar com acurácia variáveis biométricas que, em geral, são difíceis de determinar diretamente em campo e de alto custos em inventários florestais, sempre constituiu um dos principais interesse de pesquisadores. Neste contexto, a técnica de Regressão Linear (RL) convencional, simples e múltipla, tem sido mais extensivamente usada para a modelagem preditiva de variáveis florestais, como volume, altura, afilamento, biomassa, carbono, e diversas outras.

A preferência pela RL, dentre outros fatores, está principalmente associada à simplicidade da modelagem preditiva, à interpretabilidade do(s) parâmetro(s) estimados associado(s) à(s) covariável(is) e, à possibilidade de estimar a incerteza relacionada à(s) estimativa(s) do(s) parâmetro(s) do modelo estatístico (intervalos de confiança e predição). Além da RL, outros métodos paramétricos como regressão não-linear (HIGUCHI *et al.*, 1998), modelos lineares generalizados e modelos de efeito misto (FEHRMANN *et al.*, 2008), também têm conquistado espaço nas pesquisas sobre modelagem florestal.

O sucesso do emprego da RL tem sido reportado em muitos estudos, seja para predizer variáveis em povoamentos equiâneos (PELISSARI; LANSSANOVA; DRESCHER, 2011; OUNBAN; PUANGCHIT; DILOKSUMPUN, 2016), seja em florestas naturais inequiâneas (BROWN; GILLESPIE; LUGO, 1989; HIGUCHI *et al.*, 1998; CHAVE *et al.*, 2005, 2014), como a Amazônia brasileira. Apesar disso, não é incomum encontrar pesquisas que reportam erros-padrões de estimativas (S_{yx}) expressivos com uso de RL, sobretudo quando a intenção é obter modelos genéricos para florestas nativas, cuja elevada heterogeneidade das variáveis biométricas é uma condição marcante e intrínseca, e que torna a modelagem preditiva um desafio.

Os estudos de Cysneiros *et al.* (2017) e Araújo *et al.* (2018), por exemplo, são evidências da dificuldade de realizar modelagem multiespécies do volume e biomassa, respectivamente, usando da técnica de regressão linear. Em Cysneiros *et al.* (2017), Schumacher-Hall foi o melhor modelo genérico para predizer o volume de 32 espécies comerciais (n=5.231; S_{yx}=35,97%), em área de manejo sob concessão na Floresta Amazônica. Em Araújo *et al.* (2018), os modelos genéricos de biomassa seca total (n=111; 50 espécies), em áreas de restauração na Mata Atlântica, apresentaram erros-padrões de estimativa ainda mais elevados. Quando o diâmetro foi a única covariável usada para explicar a biomassa, o S_{yx} esteve entre 70% e 124%, já para as equações com a variável altura incluída S_{yx} variou entre 60% e 84%. Goussanou *et al.* (2016) ajustaram modelos genéricos para estimar a biomassa de 18 espécies arbóreas em uma floresta semi-decídua da África Ocidental e, encontraram rRMSE=39% para o melhor modelo.

Na busca por modelos mais acurados, a modelagem em nível de espécie por meio de RL constitui uma abordagem alternativa para prover melhorias na acurácia em relação aos modelos genéricos (GOUSSANOU *et al.*, 2016), porém, melhorias nem sempre são alcançadas e, por vezes, o ajuste de modelos específicos pode não promover aumento significativo da acurácia quando comparado aos modelos genéricos (VIBRANS *et al.*, 2015). A estratificação como função de classes de diâmetro, razão altura-diâmetro e grupo ecológico de espécies, também são estratégias usuais para

conseguir modelos mais acurados em florestas heterogêneas (CYSNEIROS *et al.*, 2017; ARAÚJO *et al.*, 2018).

Nas últimas duas décadas uma nova cultura de modelagem estatística progrediu extensivamente: o Aprendizado de Máquina - AM (do inglês, *Machine Learning* - ML) (JORDAN; MITCHELL, 2015), que tem sido amplamente usado para solucionar problemas complexos em diversas áreas do conhecimento. Essa nova cultura de modelagem estatística foi denominada por Breiman (2001b) de "A Cultura da Modelagem Algorítmica" (do inglês, *"The Algorithmic Modeling Culture"*), cuja abordagem considera que a forma funcional que descreve os dados é complexa e desconhecida.

O Aprendizado de Máquina é comumente classificado em duas categorias: Aprendizado Supervisionado (AS) e Aprendizado Não-Supervisionado (ANS). No Aprendizado Supervisionado, para cada observação de preditora(s), x_i ($i = 1, ..., n$), existe uma resposta associada y_i (JAMES *et al.*, 2013). A idéia é que cada y_i exerça um papel de "Professor", responsável por supervisionar o processo de aprendizagem de uma função $f(x)$ (HASTIE; TIBSHIRANI; FRIEDMAN, 2016). Em geral, o AS inclui tarefas de regressão e classificação, enquanto que o ANS engloba problemas de agrupamento e associação.

Na mensuração florestal, em particular no Brasil, as pesquisas com AS foram impulsionadas no início da década de 2010, especialmente direcionadas ao uso de Redes Neurais Artificiais - RNA (do inglês, *Artificial Neural Networks* - ANN) em povoamentos florestais. Apesar do interesse recente no Brasil, estudos empíricos remotos (HAARA; MALTAMO; TOKOLA, 1997; SIRONEN *et al.*, 2003; MALTAMO; EERIKAINEN, 2001; MALTAMO *et al.*, 2003) já relatavam o potencial de técnicas de aprendizagem de máquina, no caso específico do *k*-Nearest Neighbor (*k*-NN), na predição de variáveis florestais.

Nas ciências florestais, as aplicações mais recentes usando técnicas de "Aprendizado Supervisionado" incluem: 1) modelagem de variáveis biométricas, tais como altura (BINOTI; BINOTI; LEITE, 2013), volume de árvores (SOARES *et al.*, 2011; BINOTI; SILVA BINOTI; LEITE, 2014), afilamento do fuste (SCHIKOWSKI; CORTE; SANQUETTA, 2015; SCHIKOWSKI *et al.*, 2018), biomassa (FEHRMANN *et al.*, 2008; SANQUETTA *et al.*, 2015; VAHEDI, 2016), carbono (SANQUETTA *et al.*, 2013), volume e espessura de casca (DIAMANTOPOULOU; MILIOS, 2010; NIETO *et al.*, 2012), e densidade básica da madeira (LEITE *et al.*, 2016); 2) modelagem de crescimento individual e produção florestal (ASHRAF *et al.*, 2015); 3) modelagem da mortalidade, sobrevivência e recrutamento (REIS *et al.*, 2018; ROCHA *et al.*, 2018; REIS *et al.*, 2019); 4) modelagem de atributos florestais a partir de dados de sensoriamento remoto (AGUIRRE-SALADO *et al.*, 2009; MCROBERTS, 2012); 5) mapeamento e classificação de florestas (KUPLICH, 2006); 6) reconhecimento de espécies florestais (MARTINS *et al.*, 2013; PAULA FILHO *et al.*, 2014; MARTINS *et al.*, 2015; MARUYAMA *et al.*, 2018); 7) reconhecimento de doenças em plantas (SINGH; MISRA, 2017); 8) previsão de incêndios florestais (SAKR *et al.*, 2010).

Diversos algoritmos de AM têm sido desenvolvidos. Porém, as RNAs têm sido mais empregadas nas pesquisas de modelagem florestal. O algoritmo Máquinas de Vetores de Suporte (MVS) (do inglês, *Support Vector Machine* - SVM) também tem ganhado destaque pelos bons resultados (NIETO *et al.*, 2012; MONTAÑO *et al.*, 2017). Mais recentemente o emprego do método do *k*-NN tem ganhado maior destaque pelo sucesso na previsão de variáveis, como estoque de carbono, biomassa e volume

(FEHRMANN *et al.*, 2008; SANQUETTA *et al.*, 2013; SANQUETTA *et al.*, 2018).

Pesquisas recentes, a exemplo de Reis *et al.* (2018), Rocha *et al.* (2018) e Reis *et al.* (2019) tem contribuído para o aumento do conhecimento sobre a potencialidade de técnicas de aprendizado de máquina na predição de variáveis na Amazônia brasileira e Mata Atlântica, e também sobre a habilidade de modelar dados oriundos de diversos sítios tropicais, como em Montaño *et al.* (2017). Apesar disso, técnicas de aprendizado de máquina para predição de biomassa aérea total e volume comercial com casca em florestas naturais inequiâneas ainda são pouco estudados.

Apesar do sucesso do emprego das técnicas de AM em diversas áreas do conhecimento, comumente a metáfora da "caixa preta" (do inglês, *Black-Box*) é usada para questionar a interpretabilidade dos modelos aprendidos, em outras palavras, a dificuldade de interpretar/compreender a função f(x) modelada pelos algoritmos. Portanto, questões como: Qual a importância relativa de preditoras para o modelo preditivo? Qual a relação (ou efeito) marginal entre resposta-preditor no modelo? são recorrentes quando técnicas de AM são usadas.

Felizmente, recentemente algumas iniciativas têm demonstrado que é possível obter algum nível de interpretabilidade para modelos de aprendizado supervisionado, fato que incentivou a criação de bibliotecas especializadas, como DALEX (*Descriptive mAchine Learning Explanations*) (BIECEK, 2018), LIME (*Local Interpretable Model-Agnostic Explanations*) (PEDERSEN; BENESTY, 2019) e iml (*Interpretable Machine Learning*) (MOLNAR; BISCHL; CASALICCHIO, 2018), disponíveis no ambiente estatístico R. A biblioteca DALEX possui uma coleção consistente de "explicadores" (ou funções), que incorporam as abordagens mais conhecidas para explicação de modelos preditivos (BIECEK, 2018). Duas abordagens conhecidas são: 1) gráficos de importância de variáveis preditoras; e 2) gráficos de dependência parcial (do inglês, *Partial Dependence Plots* - PDP) (FRIEDMAN, 2001).

Os modelos de aprendizado de máquina (MAM) construídos, em sua maioria, possuem estrutura bastante complexa e, portanto, o seu uso prático e intuitivo pode ser dificultoso, em especial para não programadores ou pessoas com pouco conhecimento em tecnologias. Portanto, uma maneira amigável de disponibilizar os modelos aprendidos é por meio de aplicações web. Em particular, a linguagem R dispõe do framework "Shiny" (CHANG *et al.*, 2019) idealizado para o desenvolvimento de aplicações web. A aplicação pode ser compartilhada como uma página da web, usando serviços de hospedagens gratuitos.

No aprendizado de máquina uma das vantagens frente à regressão tradicional é não requerer a especificação a priori de uma forma funcional que descreva a relação entre preditor(es) e a variável resposta (FEHRMANN *et al.*, 2008; SANQUETTA *et al.*, 2015). Ainda, são flexíveis (SANQUETTA *et al.*, 2015) e, portanto, podem ser mais eficazes para modelar relações não-lineares mais complexas, além da versatilidade de aplicações. Ademais, as técnicas de AM são flexíveis à inclusão de variáveis qualitativas no espaço de preditores, estratégia que pode contribuir para construção de modelos biométricos mais acurados, seja em povoamentos florestais, seja em florestas naturais inequiâneas.

Finalmente, este trabalho constitui uma contribuição ao conhecimento sobre modelagem preditiva de variáveis em florestas naturais inequiâneas por meio de técnicas de aprendizado de máquina. Particularmente, tradicionais e modernos algoritmos

de AS foram usados para modelar a biomassa aérea total e volume comercial de árvores individuais de diferentes espécies em Florestas Tropicais. Para tanto, dois estudos de casos foram conduzidos. No primeiro, foi usada uma base global compilada por Chave *et al.* (2014), contendo dados de Biomassa Aérea Total (BAT) de 4004 árvores-amostras (diâmetro $\geq$ 5cm) colhidas e distribuídas em 58 sítios de diferentes países. No segundo, foi usado um conjunto de dados com volume comercial com casca (13.831 árvores; 38 espécies) obtido em cubagens de romaneio, em área de Manejo Florestal Sustentável.

1.1 HIPÓTESES

- Técnicas de aprendizado de máquina podem apresentar melhor acurácia na estimativa da biomassa e volume comercial de árvores individuais em florestas naturais inequiâneas, em detrimento as abordagens de regressão linear tradicionalmente usada nas ciências florestais.
- A construção de novas covariáveis (*feature engineering*) é uma tarefa crucial para o sucesso da modelagem preditiva de variáveis em florestas naturais inequiâneas por meio de técnicas de aprendizado de máquina, uma vez que os preditores podem ter desigual importância para diferentes algoritmos.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Estudar e comparar o potencial de técnicas de aprendizado de máquina na melhoria das estimativas da biomassa aérea total e volume comercial de árvores individuais, frente à abordagem de modelagem por regressão tradicional, a partir de dados coletados em diversos sítios de florestas tropicais.

1.2.2 Objetivos Específicos

- Ajustar e avaliar o desempenho preditivo de modelos de aprendizado de máquina (MAM) para predição de biomassa e volume comercial em florestas naturais inequiâneas;
- Realizar “engenharia de variáveis”, em cada base de dados, na busca de melhores representações do espaço de covariáveis, para construir modelos de aprendizado de máquina mais acurados;
- Implementar abordagens atuais para compreender a importância de variáveis preditoras e a relação preditor-resposta em modelos de aprendizado de máquina, desmistificando a metáfora da “caixa preta";
- Desenvolver uma aplicação web para disponibilizar os modelos de aprendizado de máquina mais precisos para estimava de biomassa aérea total e volume comercial com casca em florestas naturais inequiâneas.

2 REVISÃO TEÓRICO-EMPÍRICA

2.1 BIOMASSA FLORESTAL

2.1.1 Quantificação de biomassa em ecossistemas florestais

O termo “biomassa” deve ser distinguido do termo “massa”, isto porque, a massa de um objeto ou indivíduo expressa seu peso obtido diretamente numa balança. Portanto, a biomassa se refere à massa dos componentes de uma árvore excluindo-se a água, isto é, a “massa seca” da árvore (BATISTA; COUTO; SILVA FILHO, 2014). Sanquetta *et al.* (2002) definiram a biomassa florestal como a massa de matéria biológica vegetal, viva ou morta, existente na floresta ou apenas na fração arbórea. Quando o objetivo é estimar a biomassa das plantas, como é o caso dos inventários de carbono, então o termo correto a ser usado é a fitomassa (SOMOGYI *et al.*, 2007).

Ao detalharem um protocolo de medição e estimativa de biomassa e carbono florestal Higa *et al.* (2014) descreveram os seguintes constituintes para os compartimentos da fitomassa:

- **Biomassa acima do solo** (*above-ground biomass*): troncos, tocos, galhos, copa, sementes e folhas;
- **Biomassa abaixo do solo** (*below-ground biomass*): raízes vivas (excluindo aquelas com diâmetro < 2 mm, pois não podem ser distinguidas da matéria orgânica do solo);
- **Serrapilheira**: toda a biomassa morta acima do solo, inclusive madeira morta com diâmetro < 2 cm, em vários estágios de decomposição;
- **Necromassa**: toda a biomassa lenhosa morta caída no chão da floresta, que não faz parte da serrapilheira. Inclui o que já está caído no solo e também preso às árvores ou em pé, com diâmetro > 2 cm.

Os métodos utilizados para quantificação de biomassa florestal dividem-se em diretos (destrutivos) e indiretos (não destrutivos) (VASHUM; JAYAKUMAR, 2012). Os métodos diretos implicam em determinações, enquanto que os indiretos geram estimativas de biomassa (SANQUETTA; BALBINOT, 2004). Em função das medidas tomadas diretamente em campo, podem-se distinguir duas técnicas de determinação direta da biomassa: i) gravimétrica; e ii) volumétrica (BATISTA; COUTO; SILVA FILHO, 2014).

Na técnica gravimétrica a biomassa é obtida a partir da pesagem da massa dos diversos compartimentos da árvore imediatamente após o corte da árvore (massa verde) e, do teor de umidade médio determinado em laboratório, com base em amostras de discos da madeira dos componentes da árvore. Posteriormente, a biomassa seca de cada componente da árvore é obtida através do produto da massa verde pelo teor de umidade (BATISTA; COUTO; SILVA FILHO, 2014). A técnica volumétrica é aplicada para facilitar as atividades no campo (SANQUETTA *et al.*, 2002). Por esta técnica a biomassa é determinada pelo produto do volume real (v_i), obtido por cubagem rigorosa,

e a densidade básica da madeira definida em laboratório, com base em amostras de discos tomadas do lenho da árvore (BATISTA; COUTO; SILVA FILHO, 2014) (Equação 2.1). Em que: *i* = *i*-ésima árvore da espécie comercial; b_{vi} = biomassa da *i*-ésima espécie, por meio da técnica volumétrica, em kg; v_i = volume da *i*-ésima espécie, obtido por cubagem rigorosa (m^3); e d_{bi} = densidade básica da *i*-ésima espécie, em $kg.m^{-3}$.

$$b_{vi} = v_i.d_{bi} \tag{2.1}$$

Em relação às técnicas de amostragem de biomassa pelo método destrutivo, Sanquetta *et al.* (2002) reportam dois métodos: i) método da árvore individual; e ii) método da parcela.

O emprego de métodos diretos é inviável para aplicação em grandes áreas, devido ao tempo e ao custo de execução (SANQUETTA *et al.*, 2014), que podem variar conforme os objetivos e as restrições técnicas (SANQUETTA; BALBINOT, 2004). Diante disso, inúmeras pesquisas (BROWN; GILLESPIE; LUGO, 1989; HIGUCHI *et al.*, 1998; CHAVE *et al.*, 2005, 2014; MATE; JOHANSSON; SITOE, 2014) têm sido conduzidas no sentido de testar a acuracidade de métodos indiretos para a estimativa de biomassa e carbono em florestas.

Os métodos indiretos de quantificação da biomassa baseiam-se principalmente na modelagem preditiva da biomassa por meio do emprego de técnicas de regressão (SANQUETTA *et al.*, 2014; BROWN; GILLESPIE; LUGO, 1989; CHAVE *et al.*, 2005, 2014; NATH *et al.*, 2019) ou no uso de técnicas de sensoriamento remoto (HE, 2012; MCROBERTS; NÆSSET; GOBAKKEN, 2015; NAESSET *et al.*, 2016; WU *et al.*, 2016; TANAGO *et al.*, 2018; LI *et al.*, 2018). O uso de Fatores de Expansão de Biomassa (FEB) e razão de raízes (R) também constituem abordagens indiretas para estimar a fitomassa (CORTE; SILVA; SANQUETTA, 2012; SKOVSGAARD; NORD-LARSEN, 2012; SOARES; TOMÉ, 2012; LIM *et al.*, 2013; NORD-LARSEN; NIELSEN, 2015; NJANA, 2017), bastante úteis quando a biomassa através do volume e densidade da madeira está disponível (CORTE; SILVA; SANQUETTA, 2012).

As equações alométricas constituem o método indireto mais utilizado para estimar a biomassa da floresta (VASHUM; JAYAKUMAR, 2012), as quais consistem na construção de uma relação funcional entre a biomassa e outras variáveis biométricas da árvore (BROWN; GILLESPIE; LUGO, 1989). Em geral, as equações alométricas de biomassa são desenvolvidas em função do diâmetro da árvore a 1,30 m do solo (DAP) e suas transformações (BROWN; GILLESPIE; LUGO, 1989; CHAMBERS *et al.*, 2001; NOGUEIRA *et al.*, 2008), ou da combinação das variáveis DAP, altura e/ou densidade básica da madeira (CHAVE *et al.*, 2005; FELDPAUSCH *et al.*, 2012; CHAVE *et al.*, 2014), e dependem de dados obtidos por meio de métodos destrutivos como os usados nas pesquisas de Overman, Witte e Saldarriaga (1994), Chambers *et al.* (2001), Chave *et al.* (2005), Nogueira *et al.* (2008), Feldpausch *et al.* (2012) e Chave *et al.* (2014).

Em se tratando de equações alométricas para estimar a biomassa acima do solo (AGB) ainda persistem inúmeras incertezas (CHAVE *et al.*, 2004; FELDPAUSCH *et al.*, 2012). Em relação a isso, Chave *et al.* (2004) destacam o pouco uso da variável independente densidade da madeira, cuja negligência pode comprometer a acurácia das estimativas de biomassa e, ainda, Goodman, Phillips e Baker (2014) alertam para o uso de um número ínfimo ou mesmo inexistente de dados de árvores de grande porte para a modelagem de relações alométricas. Iqbal et al. (2014) advertem que o uso de

equações alométricas para árvores de grande porte deve ser cuidadosamente avaliado, sobretudo se o indivíduo estiver fora do intervalo de variação diamétrica considerado para o modelo desenvolvido.

2.2 APRENDIZADO DE MÁQUINA: ESTADO DA ARTE E APLICAÇÕES NAS CIÊNCIAS FLORESTAIS

Os relatos históricos apontam que o termo “Inteligência Artificial” (IA) (do inglês, *Artificial Intelligence*) foi usado pela primeira vez no início da década de 50, pelo Cientista da Computação John McCarthy, e que seu trabalho científico intitulado “*Programs with common sense*” (MCCARTHY, 1959) foi provavelmente o primeiro artigo sobre lógica de IA. Em 1959, o termo “Aprendizagem de Máquina” (do inglês, *Machine Learning* - ML) foi cunhado pela primeira vez por Arthur Lee Samuel na pesquisa intitulada “*Some Studies in Machine Learning Using the Game of Checkers*” (SAMUEL, 1959). Baseado nesta publicação uma definição de ML foi parafraseada de Samuel (1959): “um subconjunto da inteligência artificial que frequentemente usa técnicas estatísticas para dar aos computadores a capacidade de “aprender” com dados, sem serem explicitamente programados.” Mais recentemente surgiu o termo “Aprendizagem Profundo” (do inglês, *Deep Learning*), que constitui um subconjunto do aprendizado de máquina, que baseia-se na aprendizagem de representações de dados.

Atualmente, a Inteligência Artificial é uma das áreas de estudo mais amplamente discutidas e de grandes avanços nos mais diversos campos de conhecimento. Nas ciências florestais, as aplicações mais recentes de IA incluem: 1) modelagem de variáveis biométricas, tais como altura (BINOTI; BINOTI; LEITE, 2013), volume de árvores (SOARES *et al.*, 2011; BINOTI; SILVA BINOTI; LEITE, 2014), afilamento do fuste (SCHIKOWSKI; CORTE; SANQUETTA, 2015; SCHIKOWSKI *et al.*, 2018), biomassa (FEHRMANN *et al.*, 2008; SANQUETTA *et al.*, 2015; VAHEDI, 2016), carbono (SANQUETTA *et al.*, 2013), volume e espessura de casca (DIAMANTOPOULOU; MILIOS, 2010; NIETO *et al.*, 2012), e densidade básica da madeira (LEITE *et al.*, 2016); 2) modelagem de crescimento individual e produção florestal (ASHRAF *et al.*, 2015); 3) modelagem da mortalidade, sobrevivência e recrutamento (REIS *et al.*, 2018; ROCHA *et al.*, 2018; REIS *et al.*, 2019); 4) modelagem de atributos florestais a partir de dados de sensoriamento remoto (AGUIRRE-SALADO *et al.*, 2009; MCROBERTS, 2012); 5) mapeamento e classificação de florestas (KUPLICH, 2006); 6) reconhecimento de espécies florestais (MARTINS *et al.*, 2013; PAULA FILHO *et al.*, 2014; MARTINS *et al.*, 2015; MARUYAMA *et al.*, 2018); 7) reconhecimento de doenças em plantas (SINGH; MISRA, 2017); 8) previsão de incêndios florestais (SAKR *et al.*, 2010).

Poderosos algoritmos de aprendizado supervisionado, como Redes Neurais Artificiais (RNA) e Máquinas de Vetores de Suporte (MVS), têm sido preferidos nas pesquisas de modelagem de variáveis biométricas. Mais recentemente o emprego do método conhecido como *k*-Nearest-Neighbor (*k*-NN) tem ganhado maior destaque pelo sucesso na previsão de variáveis florestais (FEHRMANN *et al.*, 2008; SANQUETTA *et al.*, 2013; SANQUETTA *et al.*, 2018). A seguir alguns estudos com IA nas ciências florestais são apresentados, com relato da variável resposta e a técnica de aprendizado supervisionado de sucesso.

Diamantopoulou (2005) interessado em estimar o volume de casca de árvores de *Pinus brutia*, avaliou o uso de redes neurais artificiais como uma alternativa aos

modelos de regressão não-linear e, encontrou melhor desempenho para o modelo RNA com valor de RMSE = 6,02%. Diamantopoulou e Milios (2010) também estudaram a espécie *Pinus brutia*, porém objetivando estimar o volume total reportaram que as redes neurais forneceram previsões imparciais e com melhor precisão em comparação com modelos de regressão convencional. Os autores expuseram que os modelos de RNA são superiores devido à sua capacidade de superar diversos problemas em dados florestais, tais como relações não-lineares, distribuições não gaussianas, multicolinearidade, outliers e ruído nos dados.

Özçelik *et al.* (2010) usaram RNA para predizer o volume de fustes de árvores de espécies da Turquia, e expuseram que os modelos de redes neurais artificiais em cascata (CCANN) foram confiáveis para estimar o volume de fuste arbóreo das quatro espécies (*Scots pine*, *Cilicica fir*, *Brutian pine* e *Cedar of Lebanon*). A eficiência da técnica RNA para predizer a altura de povoamentos equiâneos de eucalipto também foi reportada, com valores de coeficiente de correlação superiores a 0,99 (BINOTI; BINOTI; LEITE, 2013). Schikowski, Corte e Sanquetta (2015) avaliaram o emprego de RNA em estudos da forma do fuste de *Eucalyptus* sp. e reportaram que o desempenho das RNAs foi semelhante ao das funções de afilamento na estimava do diâmetro relativo, porém foram mais precisas para estimativa do volume total e comercial do que modelos de afilamento tradicionais, como *Hradetzky* e *Garay*. Nunes e Görgens (2016) usando técnicas de IA recomendaram o uso de redes neurais para previsões de afilamento e volume em florestas tropicais, especialmente quando a forma do caule e a variação na arquitetura das árvores são complexas.

O uso do algoritmo de máquina de vetores de suporte também tem mostrado bons resultados em estudos de modelagem biométrica. Nieto *et al.* (2012) modelaram o volume de casca de árvores de Eucalyptus globulus, e encontraram superioridade do algoritmo MVS com núcleo radial sobre os modelos de regressão alométrica e o redes neurais. Montaño *et al.* (2017) usaram diversas técnicas de inteligência artificial para predizer a biomassa de árvores em florestais tropicais e compararam à modelagem tradicional, e expuseram que o MVS mostrou melhor precisão para estimativa da biomassa.

Pesquisas recentes têm também despertado para o potencial do algoritmo *k*-nearest neighbor (*k*-NN) na predição de variáveis biométricas, como estoque de carbono, biomassa, altura e volume de árvores individuais (FEHRMANN *et al.*, 2008; SANQUETTA *et al.*, 2013; SANQUETTA *et al.*, 2018). Fehrmann *et al.* (2008) relataram que *k*-NN foi superior aos modelos de regressão linear e efeito misto para estimar a biomassa individual de árvores de *Picea abies* (L.) Karst e *Pinus sylvestris* L. Sanquetta *et al.* (2015) relataram um ganho de até 16,5% na redução do erro padrão da estimativa para a abordagem *k*-NN em relação ao famoso modelo de *Schumacher-Hall* em plantios de árvores no bioma Mata Atlântica. Sanquetta *et al.* (2018) relataram que a abordagem *k*-NN melhorou as estimativas de volume de *Cryptomeria japonica* no sul do Brasil, fornecendo uma estimativa mais acurada dos volumes total e comercializável. Souza *et al.* (2019) avaliaram o emprego da abordagem *k*-NN na estimativa do volume do tronco da espécie *Tectona grandis*, e expuseram que a abordagem mostrou precisão similar aos modelos clássicos de *Spurr* (ln) e *Schumacher-Hall*, porém com a vantagem de usar apenas do preditor DAP (diâmetro a 1,30m do solo) como covariável nos modelo *k*-NN. Apesar dos estudos mais recentes, outros estudos empíricos remotos também encontraram boa evidência para o uso de *k*-NN na mensuração florestal, por

exemplo, Korhonen e Kangas (1997), Maltamo e Eerikainen (2001) e Sironen *et al.* (2003).

O uso de abordagens de aprendizagem de máquina e de dados de sensores remotos para a modelagem de atributos florestais têm sido também bastante estudado. Aguirre-Salado *et al.* (2009) usou dados espectrais do sensor espacial de alta resolução SPOT 5 HRG e o algoritmo *k*-NN para predizer o estoque de carbono da parte aérea das árvores em uma floresta de *Pinus patula*, no México, e reportaram erro médio quadrático usando o método *k*-NN foi de 22,24 $Mg.ha^{-1}$ (35,43%), e afirmaram que estimativa total com uso do *k*-NN foi a mais próxima da obtida pela amostragem estratificada tradicional. McRoberts (2012) empregou o método *k*-NN na estimativa do volume médio do tronco por unidade de área usando uma combinação de observações de inventário florestal e imagens do Landsat Thematic Mapper (TM), e em suas conclusões destacou que a abordagem *k*-NN, com uso da métrica de distância euclidiana e pesos iguais, produziu resultados que foram comparáveis aos métodos de otimização mais complexos e computacionalmente intensivos.

Outra aplicação de grande destaque atual é reconhecimento de espécies florestais, que comumente enfocam o uso de imagens de folhas e do lenho da madeira (macro e microscópicas) para a extração de características e desenvolvimento de classificadores automáticos. Os estudos recentes nesta área têm envolvido o uso de algoritmos de aprendizagem de máquina tradicionais (MVS, RNA, Floresta aleatória, entre outros) e abordagens mais avançadas como Redes Neurais Convolucionais (do inglês, *Convolutional Neural Networks* - CNN) aliado ao campo da visão computacional. Martins *et al.* (2013) usaram 2.240 imagens microscópicas de 112 espécies florestais para construir um sistema de classificação automático, e o uso combinado do extrator de características conhecido como Padrão Binário Local e do classificador MVS garantiu melhor desempenho (98,6% e 86,0%) para os dois experimentos realizados.

Paula Filho *et al.* (2014) propuseram a estratégia de dividir e conquistar, e obtiveram melhoria de 9% na taxa de reconhecimento usando de imagens macroscópicas de madeiras, e a melhor precisão foi de 97,77%. O estudo de Martins *et al.* (2015) usou a mesma base de imagens microscópicas de Martins *et al.* (2013) e encontrou taxa de reconhecimento de 93,03% usando um método de seleção dinâmica de classificadores. Pesquisas com reconhecimento de espécies a partir de imagens de carvão de madeiras nativas, com uso do descritor chamado Padrões Binários Locais (do inglês, *Local Binary Patterns* - LBP) associado a classificadores de aprendizado de máquina de última geração e de Redes Neurais Convolucionais, revelaram taxas de reconhecimento superiores à 90% (MARUYAMA *et al.*, 2018). Yigit *et al.* (2019) usaram de características visuais de folhas sadias para construir um identificador automático para reconhecer 32 espécies vegetais, alcançando uma precisão de (92,91%) ao usar o algoritmo MVS. Um estudo usando imagens de 17 espécies de flores (n=1360) publicadas pela Universidade de Oxford foi realizado para construir um modelo *Deep Learning* para extrair automaticamente as características e reconhecer em campo imagens de flores (TIAN; CHEN; WANG, 2019).

2.3 PACOTE CARET: UM FRAMEWORK PARA APRENDIZADO DE MÁQUINA

O aprendizado de máquina (*machine learning*) progrediu extensivamente nas últimas duas décadas (JORDAN; MITCHELL, 2015), e inúmeras abordagens e técnicas

surgiram para solucionar problemas de classificação e regressão (TRAWIŃSKI *et al.*, 2012). Atualmente, diversos *frameworks* para aprendizado de máquina estão disponível em linguagens de código aberto, como R e Python. Na linguagem Python destaca-se a biblioteca *Scikit Learn* e em R a pacote CARET (***C**lassification **A**nd **R**egression **T**raining*). Em particular, a linguagem R têm se destacado pela disponibilização de inúmeros pacotes para tarefas de classificação e regressão. Até julho de 2019, existiam 103 pacotes publicados no *CRAN Task View: Machine Learning & Statistical Learning*, cujo mantenedor é Torsten Hothorn. Desse total, nove bibliotecas de aprendizado de máquina (cor azul) foram usadas neste estudo para a predição das variáveis biométricas (TABELA 1). Dos pacotes aqui usados, apenas o 'kknn' não está relatado no CRAN Task View, porém pode ser acessado através do pacote CARET, como será discutido adiante.

TABELA 1 – PACOTES PUBLICADOS NO CRAN TASK VIEW: MACHINE LEARNING & STATISTICAL LEARNING.

CRAN Task View: Machine Learning & Statistical Learning (Mantenedor: Torsten Hothorn, Versão: 07/06/2019)						
ahaz	deepnet	GMMBoost	LiblineaR	party	REEMtree	SIS
arules	e1071	gradDescent	LogicReg	partykit	relaxo	ssgraph
BART	earth	grf	LTRCtrees	pdp	rgenoud	stabs
bartMachine	effects	grplasso	maptree	penalized	RGF	SuperLearner
BayesTree	elasticnet	grpreg	mboost	penalizedLDA	RLT	svmpath
BDgraph	ElemStatLearn	h2o	mlr	picasso	Rmalschains	tensorflow
biglasso	evclass	hda	model4you	plotmo	rminer	tgp
bmrm	evtree	hdi	MXM	quantregForest	ROCR	tree
Boruta	frbs	hdm	naivebayes	randomForest	RoughSets	trtf
bst	GAMBoost	ICEbox	ncvreg	randomForestSRC	rpart	varSelRF
C50	gamboostLSS	ipred	nnet	ranger	RPMM	vcrpart
caret	gbm	kernlab	oem	rattle	RSNNS	wsrf
COREIearn	ggRandomForests	klaR	OneR	Rborist	RWeka	xgboost
CoxBoost	glmnet	lars	opusminer	RcppDL	RXshrink	-
Cubist	glmpath	lasso2	pamr	rdetools	sda	-

FONTE: O autor (2020).
NOTA: Web scraping realizado na página do Cran Task Viewl: Machine Learning & Statistical Learning usando o pacote 'rvest' (WICKHAM, 2019)

São inúmeros os pacotes disponíveis com diferentes algoritmos implementados para tarefas de regressão e/ou classificação. Obviamente, que a decisão de qual usar dependerá dos objetivos, dos algoritmos de interesse, do tempo de execução das tarefas, e entre outros fatores. Um boa estratégia para a tomada de decisão é também avaliar a popularidade das biblioteca por meio da inspeção do número de downloads diários de pacotes do espelho CRAN. A seguir estão os 20 principais pacotes de aprendizado de máquina mais baixados, no período de um e cinco anos (FIGURA 1).

É possível constatar que todas as bibliotecas aqui usadas, direta ou indiretamente, estão ranqueadas no Top 20 dos pacotes mais acessados. O pacote 'e1071' está no topo do número de downloads no CRAN, seja no último ano ou cinco anos. Este possui funções para implementar Naïve Bayes, Máquinas de Vetores de Suporte, Clustering Fuzzy, e entre outras. Na

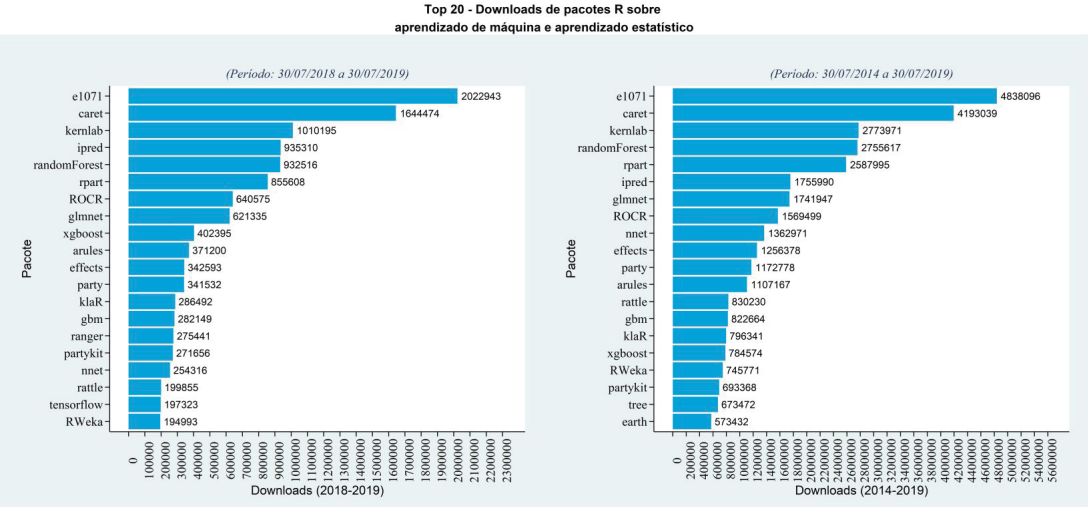
sequência, aparecem cinco pacotes acessados, direta ou indiretamente, neste estudo (CARET, kernlab, ipred, randomForest e rpart).

Uma das dificuldade de implementar aprendizado de máquina no R está nas diversas nuances sintáticas das funções dos pacotes, condição natural devido os diferentes contribuidores (KUHN, 2008). Portanto, em termos práticos, se o interesse é treinar vários algoritmos para aprender um determinado problema, o cientista deverá desprender esforços para compreender as sintaxes de diversas funções de inúmeros pacotes. Além disso, estabelecer uma abordagem metodológica em comum, que permita a comparação entre modelos ajustados, é também dificultosa.

Neste contexto, Max Kuhn e diversos contribuidores desenvolveram o *framework* CARET com os seguintes objetivos (KUHN, 2008):

- **Sintaxe**: Eliminar diferenças sintáticas entre muitas funções de construção de modelos preditivos;
- **Abordagens**: Desenvolver um conjunto de abordagens semi-automatizadas para otimizar os valores de hiperparâmetros de ajuste; e
- **Processamento Paralelo**: Criar um pacote que possa ser facilmente estendido para sistemas de processamento paralelo.

FIGURA 1 – POPULARIDADE DE PACOTES COM TÉCNICAS APRENDIZADO DE MÁQUINA E APRENDIZADO ESTATÍSTICO.



FONTE: O autor (2020).
NOTA: Os dados de downloads foram extraídos com uso do pacote 'cranlogs' (CSÁRDI, 2019).

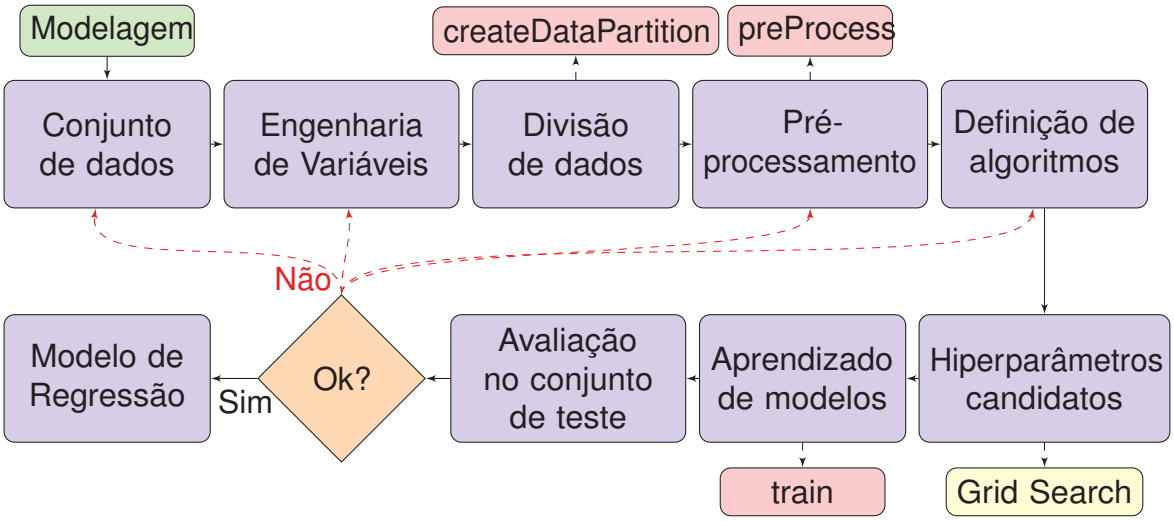
Assim, pode-se listar os seguintes motivos para escolher o pacote CARET como *framework* principal para aprendizado de máquina:

- **Diferencial**: Interface uniforme para treinamento e previsão de diversos modelos de aprendizado supervisionado;
- **Pacotes já existentes**: Torna acessível outros pacotes de AM já disponibilizados no CRAN para construir os modelos preditivos;

- **Padronizar tarefas comuns**: padroniza as entradas e saídas no processo de construção dos modelos;
- **Novas funções**: Incorpora e possibilita o uso de outras funções não disponíveis nos pacotes originais.

A acessibilidade aos pacotes existentes é uma das grande vantagem do CARET. Atualmente, no CARET (versão 6.0.84) existem 238 métodos disponíveis, os quais podem ser usados através de uma interface unificada. O comando names(getModelInfo()) pode ser usado para obter os nomes de métodos, após carrega a biblioteca no ambiente R. Um fluxograma simplificado do processo de aprendizado supervisionado usando algumas das principais funções do pacote CARET é apresentado (FIGURA 2). Em termos gerais, o pacote possui ferramentas para pré-processamento, estimativa de importância das variáveis, ajuste de modelo usando reamostragem, divisão de dados, seleção de recursos. A algumas das principais funções disponíveis no pacote são apresentadas (TABELA 2).

FIGURA 2 – FLUXOGRAMA SIMPLIFICADO DO PROCESSO DE APRENDIZADO SUPERVISIONADO USANDO FUNÇÕES DO PACOTE CARET.



FONTE: O autor (2020).

As técnicas de pré-processamento de dados geralmente se referem a adição, exclusão ou transformação dos dados do conjunto de treinamento (KUHN; JOHNSON, 2013), previamente a construção do modelo preditivo. Kuhn e Johnson (2013) afirmam que os diferentes modelos possuem sensibilidades distintas para os tipos de preditores. Assim, a forma com que os preditores entram no modelo também é importante. Portanto, a transformação de preditores é estratégica para diminuir os efeitos de variáveis com maiores escalas de medidas sobre a determinação das métricas de distância (WITTEN *et al.*, 2017).

Para implementar pré-processamento de preditoras o CARET possui a função 'preProcess()'. O escopo geral da função é apresentado (FIGURA 3). Através desta função diversas técnicas de pré-processamento podem ser empregadas (TABELA 3). A função preProcess() apenas estima os parâmetros necessários para cada método. Em seguida, deve-se usar a função predict.preProcess() para aplicar o(s) método(s) em conjuntos de dados específicos. Outra opção, mais comum, é usar o comando diretamente na função train().

TABELA 2 – ALGUMAS DAS PRINCIPAIS FUNÇÕES DISPONÍVEIS NO PACOTE CARET ÚTEIS PARA TAREFA DE REGRESSÃO.

Função	Tarefa
findCorrelation()	Encontrar máscaras altamente correlacionadas
nearZeroVar()	Identificar preditores com variância próxima de zero
preProcess()	Realizar pré-processamento de preditoras
createDataPartition()	Dividir aleatoriamente o conjunto de dados (estratificação)
train()	Ajustar modelos preditivos utilizando reamostragem
varImp()	Estimar a importância das variáveis preditoras
resamples()	Agrupar e visualizar os resultados da reamostragem
diff.resamples()	Fazer inferências sobre diferenças de desempenho de modelos
plotObsVsPred()	Gerar gráfico de valores observados como função dos preditos

FONTE: O autor (2020).

FIGURA 3 – ESCOPO GERAL DA FUNÇÃO preProcess().

```
preProcess(x, method = c("center", "scale"),
  thresh = 0.95, pcaComp = NULL, na.remove = TRUE, k = 5,
  knnSummary = mean, outcome = NULL, fudge = 0.2, numUnique = 3,
  verbose = FALSE, freqCut = 95/5, uniqueCut = 10, cutoff = 0.9,
  rangeBounds = c(0, 1), ...)
```

FONTE: Função de ajuda ?preProcess do pacote CARET.

A função createDataPartition() pode ser usada para criar divisões aleatórias estratificadas (*stratified random split*) de um conjunto de dados (KUHN, 2008). O escopo geral da função é dado está disponível (FIGURA 4). Em que: **y** = variável base para o particionamento; **times** = número de partições a serem geradas; **p** = porcentagem de dados do conjunto de treinamento; e **list** = argumento lógico indicando o formato de saída dos resultados.

A amostragem aleatória é realizada dentro dos níveis de **y**. Se **y** é um fator a amostragem aleatória é feita considerando os níveis de fatores, buscando-se obter representatividade de todas as classes, bem como equilibrar a quantidade amostrada dentro das classes. Se **y** é numérico a amostra é dividida em subgrupos com base em percentis. Assim, a amostragem aleatória é feita dentro desses subgrupos (KUHN *et al.*, 2016).

TABELA 3 – ALGUNS MÉTODOS PARA PADRONIZAÇÃO, NORMALIZAÇÃO E TRANSFORMAÇÃO DE VARIÁVEIS PREDITORAS APLICÁVEIS COM A FUNÇÃO prePrOcess().

Função	Tarefa
Métodos para padronização ou normalização de variáveis preditoras	
center()	Subtrai cada valor x_i da média $\bar{x} = x_i - \bar{x}$
scale()	Divide cada valor x_i pelo desvio padrão $DP_x = (x_i/DP_x)$
center and scale()	Padroniza os dados ($\bar{x}$ = 0 e DP_x = 1).
range()	Dimensiona os dados no intervalo [0, 1].
Métodos para transformação (distribuição mais simétrica)	
BoxCox()	Uma transformação Box-Cox (diferentes de zero e positivos).
Métodos de imputação de dados	
knnImpute()	Encontra os *k* vizinhos mais próximos (distância euclidiana) - obtêm a média
medianImpute()	Imputação via medianas de cada preditor
bagImpute()	Imputação via *bagging*
Outros métodos	
”YeoJohnson”, ”expoTrans”, ”pca”, ”ica”, ”spatialSign”,”corr”, ”zv”, ”nzv”, and ”conditionalX”	

FONTE: O autor (2020).

FIGURA 4 – ESCOPO GERAL DA FUNÇÃO ‘createDataPartition’.

```
createDataPartition(y, times = 1, p = 0.5, list = TRUE, groups = min(5,
    ↪ length(y)))
```

FONTE: Função de ajuda ?createDataPartition do pacote CARET.

Em geral, os algoritmos de AM possuem “parâmetros” que podem ser ajustados (*tuned*) para otimizar o desempenho para predizer uma determinada resposta-alvo. Esses parâmetros são denominados ‘hiperparâmetros’ (WITTEN *et al.*, 2017; PROBST *et al.*, 2018). Assim, o termo *hyperparameter tuning* (*hyperparameter optimization*) pode ser definido como o processo de encontrar boas configurações de hiperparâmetros de um algoritmo para um conjunto de dados específico (PROBST *et al.*, 2018).

Para ajustar modelos preditivos para diferentes hiperparâmetros de ajuste o CARET possui a função train(). A sintaxe geral da função está disponível (FIGURA 5) e seus principais parâmetros descritos (QUADRO 1). Em termos gerais, essa função é usada para configurar uma grade de hiperparâmetros candidatos, seja para tarefas de regressão ou classificação, ajustar os modelos e obter uma medida de desempenho baseada em processo de reamostragem (KUHN *et al.*, 2016).

FIGURA 5 – ESCOPO GERAL DA FUNÇÃO ‘ train()’.

```
train(x, y, method = "rf", preProcess = NULL, ...,
  weights = NULL, metric = ifelse(is.factor(y), "Accuracy", "RMSE"),
  maximize = ifelse(metric %in% c("RMSE", "logLoss", "MAE"), FALSE,
  TRUE), trControl = trainControl(), tuneGrid = NULL,
  tuneLength = ifelse(trControl$method == "none", 1, 3))
```

FONTE: Função de ajuda ?train do pacote CARET.

QUADRO 1 – Parâmetros da função train().

Parâmetro	Descrição
x	Uma matriz ou dataframe com as variáveis preditoras, apresentando as colunas devidamente nomeadas.
y	Um vetor numérico ou fator contendo a resposta-alvo (real) da amostra.
method	Uma *string* especificando qual modelo de classificação ou regressão usar. Para saber os métodos disponíveis bastar usar o comando names(getModelInfo()).
maxcompete	Recebe o tipo de pré-processamento a ser aplicado sobre as variáveis preditoras. As principais opções podem ser consultadas na TABELA 3.
preProcess	Número de divisões substitutas retidas. Caso definido como zero, o tempo de processamento será reduzido, uma vez que aproximadamente metade do tempo computacional é usado na busca por divisões substitutas (*default* = 5).
trControl	Recebe a configuração geral do processo de treinamento e ajuste de hiperparâmetros, repassada por ‘trainControl()’. Portanto, aqui, determina-se o método de reamostragem ("boot", "boot632", "optimism_boot", "boot_all", "cv", "repeatedcv", "LOOCV", "LGOCV", "none"), o número de partições ou o número de iterações de reamostragem, o número de repetições a realizar (apenas para "repeatedcv"). Aqui, também, pode-se atribuir ao argumento ‘summaryFunction’ uma função customizada das métricas de desempenho a serem calculadas na reamostragem.
tuneGrid	Recebe uma grade manual com os valores candidatos a hiperparâmetro de ajuste ótimo, a depender do algoritmo usado. O uso do argumento implica no emprego da estratégia *grid search*.
tuneLength	Número de níveis para cada hiperparâmetro de ajuste (usar apenas caso ‘tuneGrid’ não esteja especificado).

FONTE: O autor (2020).

2.4 TÉCNICAS DE APRENDIZADO DE MÁQUINA

Neste estudo, nove algoritmos de aprendizado de máquina foram usados para modelar a biomassa aérea total (BAT) (D $\geq$ 5cm) e o volume comercial com casca de árvores em florestas naturais inequiâneas:

1. *Weighted k-Nearest-Neighbor - wkNN*;

2. *Regression Trees* - RT (CART);
3. *Model Trees* - M5';
4. *Bagged Trees* - BT;
5. *Random Forest* - RF;
6. *Stochastic Gradient Boosting* - SGB;
7. *Extreme Gradient Boosting* - XGBoost;
8. *Support Vector Regression* - SVR; e
9. *Artificial Neural Networks* - ANN.

Portanto, dois tipos de algoritmos baseados em árvores foram considerados: i) aqueles baseados em árvore única (do inglês, *Single Trees Methods*) - algoritmos 2 e 3; e ii) aqueles baseados na combinação de modelos (do inglês, *Ensemble Methods*) - algoritmos 4 a 7. Combinar a decisão de diferentes modelos significa unir as várias saídas (respostas) em uma única predição. A maneira mais simples de fazer isso no caso de predição numérica é calcular a média aritmética ou talvez uma média ponderada (WITTEN *et al.*, 2017). A ideia básica dos métodos *ensemble* é combinar um conjunto de modelos de aprendizado mais fracos e construir um modelo mais forte (acurado), que tenha um desempenho melhor do que um único. Nas subseções da seção 2.4 estão detalhadas as estruturas desses algoritmos, seus principais aspectos, implementações no ambiente R e hiperparâmetros de ajuste.

2.4.1 *Weighted k-Nearest-Neighbor* - *wk*NN

O algoritmo baseado em instância (*k*-NN) é um tipo de método de aprendizagem supervisionado e não-paramétrico (HECHENBICHLER; SCHLIEP, 2004; SCHLIEP; HECHENBICHLER, 2016), que esteve no top 10 dos algoritmos mais usados em mineração de dados (WU *et al.*, 2008). O algoritmo *k*-NN pode ser usado para dois tipos de tarefas: classificação e regressão (SONG *et al.*, 2017), funcionando razoavelmente bem para problemas de baixa dimensão (HASTIE; TIBSHIRANI; FRIEDMAN, 2016). Em sua versão mais básica, o emprego do *k*-NN exige três elementos essenciais: um conjunto de objetos rotulados, uma métrica de distância ou similaridade para calcular a distância entre os objetos e o valor de *k* (número de vizinhos mais próximos) (LI; QIU; LIU, 2017).

Atualmente, o ambiente estatístico R disponibiliza implementações de diversas variações do algoritmo *k*-NN (VENABLES; RIPLEY, 2002; KUHN, 2015b; SCHLIEP; HECHENBICHLER, 2016; KUHN *et al.*, 2016; ROBNIK-SIKONJA; SAVICKY, 2017; MOUSELIMIS, 2018). Neste estudo, foi usada uma variação ponderada do algoritmo denominada "*Weighted k*-Nearest-Neighbor" (*wk*NN) (FIGURA 6), que está implementada no pacote "kknn" (*Weighted k-Nearest Neighbors Classification and Clustering*) (HECHENBICHLER; SCHLIEP, 2004; SCHLIEP; HECHENBICHLER, 2016).

O *wk*NN tem por ideia atribuir pesos (w_i) maiores aos vizinhos mais próximos da nova instância a ser predita, assim, um pequeno número de vizinhos com grandes pesos domina os outros vizinhos, que terão pouca influência na predição devido aos seus pesos baixos. A transformação das distâncias para pesos é feita com uso de funções kernel *k(.)*, após a padronização das menores métricas de distância com uso da distância do *k*+1 vizinho mais próximo. Essa padronização D_i sempre retorna valores no intervalo de 0 e 1. Os pesos atribuídos por funções kernel ficam menores com o crescimento absoluto do valor da distância *(d)*, mantendo-se as seguintes propriedades: i) $k(d) \geq 0$ para todo $d \in \mathbb{R}$; ii) *k(d)* é máximo para

d = 0; e iii) *k(d)* diminui monotonamente para $d \rightarrow \pm \infty$. Em seguida, a estimativa da variável resposta é obtida usando da média ponderada de todas as observações dentro da janela local (HECHENBICHLER; SCHLIEP, 2004).

O pacote 'kknn' dispõe de duas funções: **train.kknn** (*leave-one-out*) e **cv.kknn** (*k-fold cross validation*) úteis para otimizar os hiperparâmetros (métrica de distância, função kernel e número de vizinhos mais próximo) do algoritmo *wk*NN. Além disso, pode-se usar a função kknn() para prever um conjunto de teste com base em um conjunto de aprendizado. Ademais, no pacote estão disponíveis dez tipos de funções kernel para transformação de distâncias em pesos: *Rectangular* (Rctn); *Triangular* (Trng); *Epanechnikov* (Epnc); *Biweight* (Bwgh); *Triweight* (Trwg); *Gaussian* (Gssn); *Optimal* (Optm); *Cosine* (Cos); *Inversion* (Inv) e *Rank* (HECHENBICHLER; SCHLIEP, 2004; SCHLIEP; HECHENBICHLER, 2016), e para medir a similaridade entre vizinhos tem por base a família das métricas de Minkowski (Eq. 2.2), cujas distâncias Euclidiana (Eq. 2.3) e Manhattan (Eq. 2.4) são casos especiais (SHAHID *et al.*, 2009).

$$d = \left(\sum_{i=1}^{n} |q_i - x_i|^p \right)^{\frac{1}{p}} \tag{2.2}$$

$$d = \sqrt{\sum_{i=1}^{n} (q_i - x_i)^2} \tag{2.3}$$

$$d = \sum_{i=1}^{n} |q_i - x_i| \tag{2.4}$$

2.4.2 *Regression Trees* - RT

O *Regression Trees* - RT constitui um algoritmo baseado em árvore simples. A implementação mais conhecida é denominada *Classification and Regression Trees* - CART, e foi desenvolvida por Brieman, Friedman, Olshen e Stone em 1984, e constitui um método para solucionar tarefas de regressão e classificação (BREIMAN *et al.*, 1984). Tradicionalmente a terminologia "árvore de decisão" é usada para problemas de classificação e, "árvore de regressão" é usual quando a variável dependente é numérica.

O algoritmo CART foi idealizado sob a abordagem de "dividir para conquistar" (*divide and conquer*) (QUINLAN, 1992). Portanto, a ideia base do algoritmo é realizar divisões binárias recursivas no espaço de covariáveis, para produzir subgrupos de dados disjuntos tão homogêneos quanto possível em relação à variável resposta (BREIMAN *et al.*, 1984). A estrutura de uma árvore de regressão é similar a árvore de classificação (FIGURA 7) (BREIMAN *et al.*, 1984). Em termos gerais, uma árvore de regressão possui os seguintes componentes:

(a) **Nó raiz** (do inglês, *root node*): O nó raiz (R_1) constitui o primeiro nó de decisão da árvore, e contém todas as observações do conjunto de treinamento (PUT *et al.*, 2003). d_1 = representa a primeira divisão binária, baseada na amostra completa.

(b) **Nó intermediário** (do inglês, *intermediate node*): Um nó intermediário (I_2, I_3 e I_7) é um nó de decisão que admitiu divisão binária, originando nós filhos. d_2, d_3 e d_4 = representam divisões binária, baseadas em nós intermediários.

(c) **Nó terminal ou folha** (do inglês, *leaf or terminal node*): Um nó terminal é aquele que não admite divisão e, portanto, constituem a decisão final do preditor baseado em árvore. Em cada nó terminal T_i estão associados subgrupos de dados de menor variância em

FIGURA 6 – PSEUDOCÓDIGO DO ALGORITMO *wk*NN PARA PROBLEMAS DE REGRESSÃO.

Algoritmo 1 - Weighted k-Nearest-Neighbor

Entrada: Seja L = $[(y_i, x_i), i = 1, ..., n_L]$ um conjunto de aprendizado e Q, um conjunto com novas observações.

Saída: vetor da variável resposta $\hat{y}_q$.

início

[opcional] Pré-processamento de preditores (normalize q usando o desvio padrão do conjunto L);

para *cada observação* $q \in Q$ **faça**

1 Encontre no conjunto L os $k+1$ vizinhos mais próximos de cada observação q de acordo com uma métrica de distância $d(q, x_i)$.

2 Padronize as menores métricas de distâncias $d(q, x_i)$ usando a distância do $k+1$ vizinhos mais próximos:

$$D_i = D_{(q,x_i)} = \frac{d(q, x_i)}{d(q, x_{(k+1)})}$$

3 Transforme as distâncias padronizadas D_i com uso de uma função kernel $k(.)$, em pesos $w_i = k(D_i)$.

4 Estime o valor de cada $\hat{y}_q$ usando a fórmula:

$$\hat{y}_q = \frac{\sum_{i=1}^{k} k(D_i) y_i}{\sum_{i=1}^{k} k(D_i)} = \frac{\sum_{i=1}^{k} w_i y_i}{\sum_{i=1}^{k} w_i}$$

fim

fim

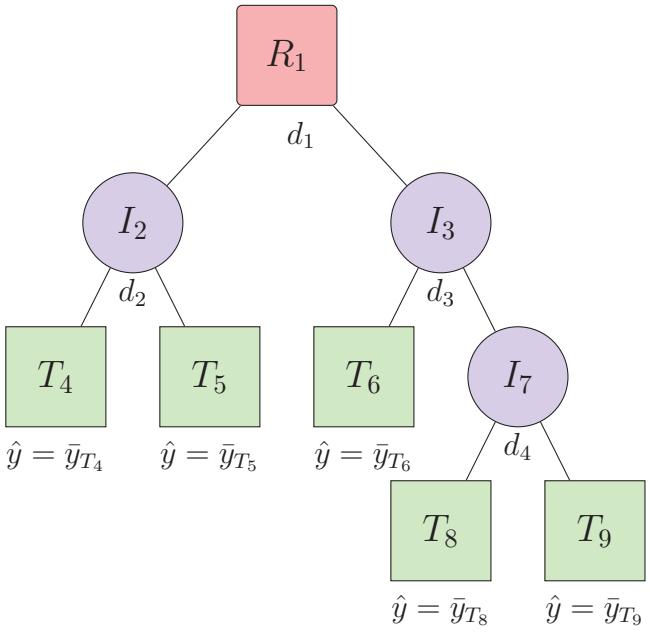
Retorna $\hat{y}_q$

FONTE: Adaptado de Hechenbichler e Schliep (2004).

relação à variável resposta. Comumente, a média aritmética da variável dependente ($\bar{y}_{T_i}$), em cada nó terminal, é usada como uma constante preditiva.

(d) **Ramos**: são conectores do tipo “se-então” (*if-else*) entre diferentes nós da árvore de decisão.

FIGURA 7 – ESTRUTURA DE UMA ÁRVORE DE REGRESSÃO.



FONTE: Adaptado de Breiman *et al.* (1984).

A primeira etapa do método CART é o "crescimento da árvore" (LEWIS, 2000). Basicamente, para construir um preditor de árvore, dado um conjunto de aprendizado L, três elementos devem ser devidamente definidos (BREIMAN *et al.*, 1984):

1. Um mecanismo de selecionar uma divisão em cada nó.
2. Uma regra para determinar quando um nó é terminal; e
3. Uma regra para atribuir um valor a cada nó terminal T_i.

Em termos práticos, James *et al.* (2013) expõem que o processo de crescimento de uma árvore de regressão pode ser resumido em dois passos:

1. Dividir o espaço de preditores, dado um conjunto de possíveis candidatos X_1, X_2, ...,X_p, em J regiões disjuntas e não sobrepostas, R_1, R_2,..,R_j.
2. Para cada observação alocada na região R_j, faz-se a mesma predição, que é simplesmente a média aritmética das observações da variável resposta do conjunto de treinamento em R_j.

Assim, em se tratando de regressão, a meta é encontrar uma árvore com regiões R_1, R_2,..,R_j que minimizem a Soma de Quadrados de Resíduos (SQR) (do inglês, *Residual Sum of Squares* - RSS) (Eq. 2.5). Em que: y_i = valor observado para a *i-ésima* observação no conjunto de treinamento; e $\hat{y}_{R_j}$ = a média aritmética da variável resposta do conjunto de treinamento na *j-ésima* região (JAMES *et al.*, 2013).

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j} \right)^2 \tag{2.5}$$

O crescimento de uma árvore começa no nó raiz (topo da árvore), que contém todas as instâncias do conjunto de treinamento (LEWIS, 2000) e, em seguida, o espaço de preditores é dividido sucessivamente e, cada divisão é indicada por dois novos ramos abaixo na árvore (JAMES *et al.*, 2013). Assim, para executar uma divisão binária, o preditor X_j e o ponto de corte s devem ser escolhidos para dividir o espaço de preditoras em regiões $\{X|X_j < s\}$ e $\{X|X_j \geq s\}$, de modo a garantir a maior redução possível no RSS (JAMES *et al.*, 2013). O algoritmo faz uma busca exaustivamente em todas as preditoras, bem como em todos os valores para cada preditora, para obter a melhor divisão (QIN; HAN, 2008). Formalmente, para algum j e ponto de corte s duas regiões podem ser representadas: $R_1(j,s) = \{X|X_j < s\}$ e $R_2(j,s) = \{X|X_j \geq s\}$. Assim, o objetivo é encontrar valores de j e s que minimizem a equação 2.6 (JAMES *et al.*, 2013). Em que: $\hat{y}_{R_1}$ e $\hat{y}_{R_2}$ = média aritmética da variável resposta do conjunto de treinamento nas regiões R_1 e R_2, respectivamente.

$$\sum_{i:\ x_i \in R_1\ (j,s)} \left(y_i - \hat{y}_{R_1}\right)^2 + \sum_{i:\ x_i \in R_2\ (j,s)} \left(y_i - \hat{y}_{R_2}\right)^2 \tag{2.6}$$

Usando os termos da publicação original de Breiman *et al.* (1984), para qualquer ponto de corte s em um nó t, obtendo-se dois nós filhos t_L (L = left) e t_R (R=right), a avaliação do mérito da divisão é baseada em maximizar a equação 2.7. Em termos práticos, isso significa que a divisão que garantir a maior redução da soma de quadrados de resíduos deve ser escolhida. Em outras palavras, da expressão 2.7 o objetivo é encontrar o menor valor para $\{R(t_L) + R(t_R)\}$, de modo a garantir a maximização de $\Delta R(s,t)$. Em que: $R(t_R)$ = soma de quadrados de resíduos no nó "filho" direito; $R(t_L)$ = soma de quadrados de resíduos no nó "filho" esquerdo; $R(t)$ = soma de quadrados de resíduos no nó "pai".

$$\Delta R(s,t) = R(t) - R(t_L) - R(t_R) \tag{2.7}$$

Modelos preditivos representados por árvores grandes e/ou complexas são propensos a sobreajustamento (*overfitting*) (KUHN; JOHNSON, 2013), ou seja, é provável que apresentem bom desempenho preditivo no conjunto de treinamento, porém pobre capacidade de generalização. Portanto, uma estratégia comum para reduzir a probabilidade de *overfitting* do modelo é realizar a "poda" (*prune*) da árvore.

Os métodos de poda podem ser divididos em dois grupos: i) pré-poda (do inglês, *pre-pruning*); e pós-poda (do inglês, *post-pruning*). A "pré-poda" implica no uso de critérios de parada, que impedem a construção de ramos que parecem não melhorar o poder preditivo da árvore. Por outro lado, a "pós-poda" implica primeiro na construção de uma árvore completa, superajustada aos dados de treinamento, e somente depois a poda é realizada (GAMA *et al.*, 2015). A implicação estrutural da pós-poda é que sub-árvores são transformadas em nós terminais. Ainda, apesar do usual termo "pré-poda", o procedimento na prática não "poda" a árvore, apenas impede o crescimento de um novo ramo (*branch*) se algum limiar (*threshold*) for atingido.

A "poda custo-complexidade" (*Cost-complexity pruning*) (Eq. 2.8), proposta por Breiman *et al.* (1984), é a técnica de pós-poda mais comum em árvores de regressão. Em que: α = parâmetro de complexidade (penalidade); $|T|$ = número de nós terminais de uma árvore T (tamanho da árvore); e $R_{(T)}$ = soma de quadrados de resíduos de uma árvore T. O parâmetro α é uma penalidade proporcional ao número de nós terminais da árvore, por conseguinte, controla a complexidade do modelo preditivo. Portanto, α é o *hiperparâmetro tuning* que deve ser encontrado por alguma técnica de reamostragem, como validação cruzada. Se α = 0, então não há penalidade, e um modelo com árvore totalmente crescida é selecionado. Por outro lado,

α = 1 implica em uma árvore sem divisões, ou seja, apenas com nó raiz.

$$R_\alpha(T) = R(T) + \alpha|T| \tag{2.8}$$

Uma vez escolhido o melhor estimador de árvore, a predição da resposta y_i desconhecida para novas observações é realizada aplicando as regras aprendidas no caminho da árvore - para determinar em qual nó terminal a nova amostra será alocada - sendo a estimativa $\hat{y}$ dada pelo valor da média aritmética da variável resposta do conjunto de treino associadas a cada nó terminal.

O algoritmo CART está implementado no pacote 'rpart' (*Recursive Partitioning and Regression Trees*) do ambiente estatístico R. O hiperparâmetro *tuning* é o "cp", conhecido como parâmetro de complexidade. A função rpart() tem o seguinte escopo (THERNEAU; ATKINSON, 2019):

FIGURA 8 – ESCOPO GERAL DA FUNÇÃO rpart().

```
rpart(formula, data, weights, subset, na.action = na.rpart, method,
      model = FALSE, x = FALSE, y = TRUE, parms, control, cost, ...)
```

FONTE: Therneau e Atkinson (2019).

O argumento 'control' é um dos principais elementos da função e, através da função rpart.control() recebe uma lista de opções que controlam detalhes do algoritmo. Os principais argumentos da função rpart.control() são apresentados (QUADRO 2). Para mais detalhes e outros argumentos recomenda-se consultar a documentação do pacote.

QUADRO 2 – Parâmetros da função rpart.control().

Parâmetro	Descrição
minsplit	Número mínimo de observações que devem existir em um nó de decisão para que uma divisão seja tentada (*default* = 20).
minbucket	Número mínimo de observações em qualquer nó terminal (*default* = round(minsplit/3)).
cp	Parâmetro de complexidade. Qualquer divisão que não diminua o erro total de ajuste por um fator de cp não é tentada. O principal papel desse parâmetro é economizar tempo de computação removendo as divisões que obviamente não valem a pena (*default* = 0,01).
maxcompete	Número de divisões do concorrente retidas na saída. É útil saber não apenas qual divisão foi escolhida, mas qual variável foi posicionada em segundo, terceiro, e assim por diante (*default* = 4).
maxsurrogate	Número de divisões substitutas retidas. Caso definido como zero, o tempo de processamento será reduzido, uma vez que aproximadamente metade do tempo computacional é usado na busca por divisões substitutas (*default* = 5).
xval	Número de validações cruzadas (*default* = 10).
maxdepth	Profundidade máxima da árvore final. O nó raiz é contado como profundidade 0 (zero) (*default* = 30).

FONTE: O autor (2020).

2.4.3 *Model Tree* - M5'

A Árvore Modelo (do inglês, *Model Tree*) foi introduzida por Quinlan (1992) como uma alternativa à árvore de regressão, proposta por Breiman *et al.* (1984), para predição de variáveis contínuas, ficando conhecida como algoritmo "M5". Pouco tempo depois, Wang e Witten (1997) propuseram uma "reconstrução racional" (*rational reconstruction*, termo usado pelos autores) do algoritmo M5. Na publicação, os autores apresentam soluções para alguns detalhes não completamente resolvidos na publicação original, nomeando de algoritmo M5'. A versão M5' está implementada no software Weka.

De modo simples, contrário às árvores de regressão, as árvores modelo empregam modelos lineares multivariados para predições nos nós terminais (QUINLAN, 1992). Basicamente, o M5 difere das árvores de regressão em três aspectos (KUHN; JOHNSON, 2013):

(a) **critério de divisão**: O critério de divisão é diferente;

(b) **modelos de regressão linear**: As predições nos nós terminais são realizadas usando um modelo de regressão linear; e

(c) **combinação de predições**: Quando a resposta y_i para uma nova amostra é estimada, geralmente é uma combinação das predições de diferentes modelos ao longo do mesmo caminho através da árvore.

O critério de divisão é usado para determinar qual a melhor preditora para dividir dados de treinamento associados a um nó específico (WITTEN *et al.*, 2017). O algoritmo M5, assim como o M5', usam uma medida conhecida como "redução do desvio padrão" (do inglês, *standard deviation reduction* - SDR) como critério de divisão para induzir uma árvore modelo. Assim, denotando todo o conjunto de dados por S e, representando por {S_1, ...,S_P} os subconjuntos P após a divisão, o critério de divisão é dado pela equação 2.9. Em que = SD = desvio padrão; n_i = número de amostras na *i-ésima* divisão (KUHN; JOHNSON, 2013).

$$SDR = SD(S) - \sum_{i=1}^{P} \frac{n_i}{n} * SD(S_i) \tag{2.9}$$

Em termos práticos, a medida SDR avalia se a variância total nos grupos, ponderada pelo tamanho da amostra, é menor do que a variância nos dados antes da divisão (KUHN; JOHNSON, 2013). Aqui, os "grupos" são originados a partir de teste sobre uma divisão potencial. Assim, para encontrar a melhor divisão, o algoritmo faz uma busca exaustiva sobre os preditores e, a divisão que maximizar SDR - garantir maior redução na variância de y - deve ser executada.

O processo de divisão é executado recursivamente sobre os nós de decisão, até que algum critério de parada seja satisfeito, indicando um nó terminal. O algoritmo M5' usa duas condições para terminar o crescimento da árvore. A primeira, se o número de amostras em um nó de decisão for < 4. A segunda, se os valores da variável resposta, que alcançam um nó, variarem apenas ligeiramente. Aqui, um valor de 5% (0,05*SD) foi estabelecido como limiar (WANG; WITTEN, 1997; WITTEN *et al.*, 2017). Em outras palavras, o processo de crescimento termina se o desvio padrão (DP) da variável resposta y em algum nó pós-divisão for menor do que 5% do DP dos valores de y antes da divisão.

As árvores modelo também incorporam um tipo de suavização (*smoothing*) para diminuir o potencial de *overfitting*. Assim, para realizar predições, a nova amostra desce pelo caminho apropriado da árvore e, movendo-se de baixo para cima, os modelos lineares ao longo desse caminho são combinados. A função de suavização é dada pela equação 2.10 (KUHN; JOHNSON, 2013). Dada a equação, para implementar o processo de suavização é necessário

também o ajuste de modelos lineares para os nós intermediários, não apenas para os terminais (WITTEN *et al.*, 2017). O processo de suavização aumenta substancialmente a precisão das predições (WANG; WITTEN, 1997).

$$\hat{y}_{(p)} = \frac{n_{(k)}\ \hat{y}_{(k)} + c\ \hat{y}_{(p)}}{n_{(k)} + c} \tag{2.10}$$

Em que: $\hat{y}_{(k)}$ = predição do modelo linear no nó “filho”; $n_{(k)}$ = número de observações do conjunto de treinamento no nó “filho”; $\hat{y}_{(p)}$ = predição do modelo linear no nó “pai”; e c = constante (*default*= 15).

Os modelos lineares ajustados possuem a forma: $w_0 + w_1 a_1 + w_2 a_2 + ... + w_k a_k$. Em que: $a_1, a_2, ..., a_k$ são valores de atributos; $w_1, w_2, ..., w_k$ são calculados usando regressão padrão. Contudo, alude-se que apenas um subconjunto dos atributos é geralmente usado, por exemplo, aqueles que são testados na subárvore abaixo do nó corrente e, talvez, aqueles que ocorrem ao longo do caminho para o nó raiz (WITTEN *et al.*, 2017).

Na linguagem R, o algoritmo M5’ implementado no software “Weka” pode ser acessado através da biblioteca ‘RWeka’. O treinamento de modelos usando o M5’ pode ser realizado através da função M5P() (FIGURA 9) (WITTEN; FRANK, 2005; HORNIK; BUCHTA; ZEILEIS, 2009) e seus principais parâmetros são descritos (QUADRO 3). Um exemplo de árvore modelo para predição do volume do fuste (m³) como função do diâmetro (cm) e altura (m) é apresentado (FIGURA 10).

FIGURA 9 – ESCOPO GERAL DA FUNÇÃO M5P().

```
M5P(formula, data, subset, na.action,
    control = Weka_control(), options = NULL)
```

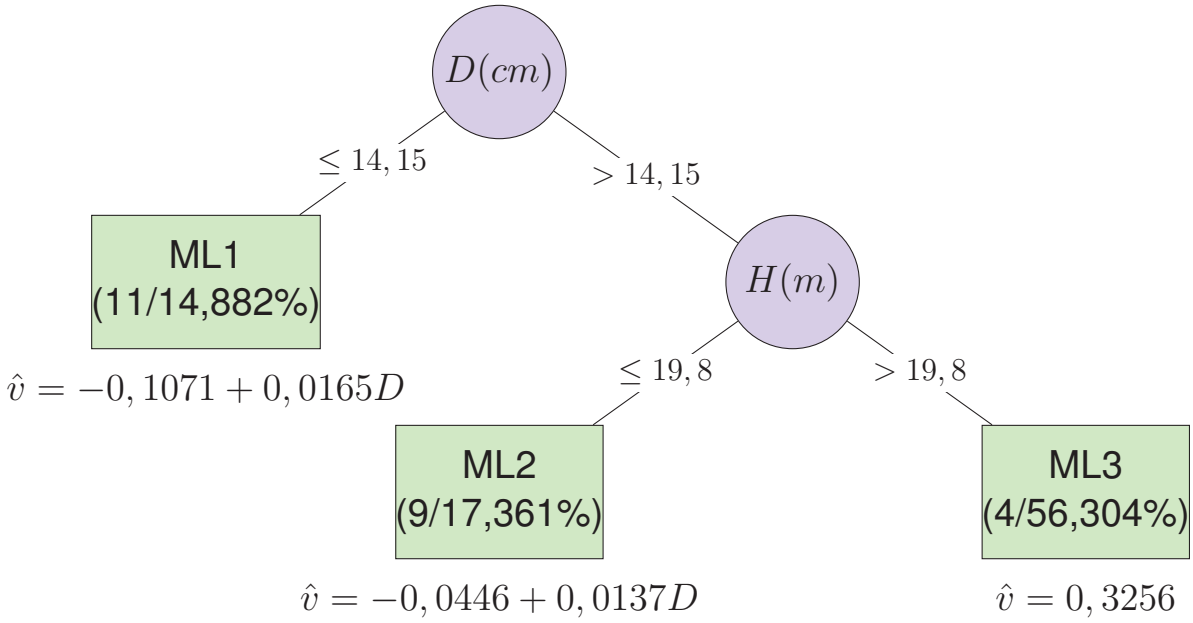
FONTE: Witten e Frank (2005) e Hornik, Buchta e Zeileis (2009).

QUADRO 3 – Parâmetros da função M5P().

Parâmetro	Descrição
formula	Uma descrição simbólica do modelo a ser estimado.
data	Um dataframe opcional contendo as variáveis no modelo.
subset	Um vetor opcional especificando um subconjunto de instâncias a serem usadas no processo de treinamento.
control	Um objeto da classe "Weka_control", que dispõe de uma lista de opções a serem repassadas para o algoritmo. As opções de controle disponíveis podem ser consultadas *on-line* usando o ‘*Weka Option Wizard*’ - WOW(“M5P”) - ou também inspecionando a documentação do Weka.

FONTE: O autor (2020).

FIGURA 10 – ÁRVORE MODELO PARA PREDIÇÃO DO VOLUME DO FUSTE.



FONTE: O autor (2020).
LEGENDA: D = diâmetro a 1,30m do solo; H = altura total; ML = Modelo Linear. Entre parêntese em cada nó terminal = (n. observações/$(RMSE_{pai}/RMSE_{filho}) * 100$).
NOTA: Árvore modelo M5' treinada com a função M5P() do 'RWeka'.

2.4.4 *Bagged Trees*

A agregação de *bootstrap* (do inglês, ***b**ootstrap **agg**regat**ing***), ou acrônimo *bagging*, constitui um dos primeiros algoritmos de conjunto (*ensemble*) desenvolvidos e proposto em 1996 por Leo Breiman, que pode ser usado para tarefas de regressão ou classificação. Em termos práticos, a ideia do *bagging* é ajustar múltiplas versões de um modelo preditivo, e depois agregá-las, para obter uma única predição combinada (BREIMAN, 1996).

O *bagging* foi idealizado com o objetivo geral de reduzir a variância de um método estatístico de aprendizagem, como as árvores de decisão, e promover melhorias substanciais na precisão (JAMES *et al.*, 2013). Para tanto, o elemento vital é a instabilidade do método de predição. Destarte, a melhoria ocorrerá em procedimentos instáveis, em que uma pequena alteração em um conjunto de aprendizado L pode resultar em grandes alterações no preditor (BREIMAN, 1996). Portanto, para modelos que produzem uma previsão instável, como as árvores de regressão, a agregação em muitas versões dos dados de treinamento reduz a variação na predição e melhoram o desempenho preditivo (KUHN; JOHNSON, 2013).

Em *bagging* o método *bootstrap* é usado em um contexto diferente, isto é, com o objetivo de melhorar os métodos estatísticos de aprendizado (JAMES *et al.*, 2013). O *bootstrap* é um método de reamostragem que consiste em produzir replicações de tamanho n, com mesmo tamanho do conjunto de aprendizado, por amostragem aleatória com reposição (GAMA *et al.*, 2015). Assim, dado um conjunto de treinamento original L de tamanho n, uma amostra *bootstrap* (bi) deverá conter, em média, aproximadamente 63% dos dados originais de treinamento, já que cada amostra possui uma probabilidade $1 - (1 - 1/n)n$ (n = número de observações do conjunto de aprendizado) de ser aleatorizada em cada bi (TAN; STEINBACH; KUMAR, 2009). Os conjuntos de dados gerados por reamostragem são diferentes uns dos outros, mas

certamente não são independentes porque são todos baseados em um mesmo conjunto de dados (WITTEN *et al.*, 2017).

O *bagging* pode ser usado, por exemplo, para combinar árvores modelos (WITTEN *et al.*, 2017), e mais comumente árvores de decisão (JAMES *et al.*, 2013). Para construir "árvores ensacadas" (*Bagged Trees*) pode-se usar como o aprendiz de base, por exemplo, o algoritmo CART. O procedimento é relativamente simples e, para o caso de regressão, pode ser resumido nos seguintes passos:

1. Dado um conjunto de treinamento original L de tamanho n, gerar uma amostra *bootstrap* (bi) de tamanho n;
2. Treinar um modelo de árvore de regressão completamente crescida (sem poda) sobre a amostra *bootstrap* bi; e
3. Salvar o modelo preditivo.

Esse procedimento pode ser realizado B vezes, em que B é a quantidade de amostras *bootstrap* pré-definida. Uma vez que todos os modelos foram construídos usando as B amostras, pode-se obter as predições de cada modelo. Então, uma predição combinada pode ser obtida através da média das predições dos modelos individuais $\{\hat{f}^{*b}(x)\}$ treinados em cada amostra *bootstrap* bi, conforme equação 2.11 dada por James *et al.* (2013). As árvores geradas em cada amostra *bootstrap* são profundas e não são podadas. Portanto, cada árvore individual possui alta variância, mas baixo viés. A média das B árvores de regressão reduz a variação (JAMES *et al.*, 2013). Ainda, no *bagging*, as árvores construídas não dependem de árvores anteriores (VOYANT *et al.*, 2018).

$$\hat{f}_{(bag)}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x) \tag{2.11}$$

Para um conjunto de treinamento L, o procedimento de amostragem *bootstrap* replica algumas das instâncias, porém, inevitavelmente outras ficam de fora (WITTEN *et al.*, 2017). As instâncias não selecionadas compõem a amostra "fora do saco" (ou "fora da bolsa") (do inglês, *Out-of-Bag*) e, portanto, não são usadas para construir os modelos preditivos (KUHN; JOHNSON, 2013).

Assim, uma das vantagens do *bagging* é poder fornecer uma estimativa interna de desempenho preditivo, para cada modelo específico. Essa estimativa é dada pela média das métricas de desempenho nas amostras "fora do saco" e, comumente conhecida como "estimativa fora do saco" (do inglês, *Out-Of-Bag estimate* - OOB). A estimativa OOB é bem correlacionada com as estimativas de validação cruzada ou de conjuntos de teste (KUHN; JOHNSON, 2013).

Em termos gerais, os modelos *ensemble* compartilham a desvantagem de serem pouco interpretáveis, podendo incluir dezenas ou até centenas de modelos individuais. Assim, embora tenham bom desempenho, não é fácil entender em termos intuitivos quais fatores estão contribuindo para melhorar as decisões. Felizmente, métodos têm sido desenvolvidos para combinar os benefícios de desempenho dos métodos de conjunto com modelos compreensíveis (WITTEN *et al.*, 2017).

O pacote 'ipred' disponível na linguagem R possui a função bagging() para treinar um modelo *ensemble*, tendo por padrão o uso de árvores de decisão como aprendizes de base. O escopo geral da função bagging() (FIGURA 11) e os seus principais parâmetros (QUADRO 4) estão disponíveis (PETERS; HOTHORN, 2019):

FIGURA 11 – ESCOPO GERAL DA FUNÇÃO bagging().

```
bagging(formula, data, subset, na.action=na.rpart, nbagg, coob=FALSE,
    ↪ control= rpart.control(xval=0), ...)
```

FONTE: Peters e Hothorn (2019).

QUADRO 4 – Parâmetros da função bagging().

Parâmetro	Descrição
formula	Uma fórmula na forma $y \sim x$. Em que y é a variável resposta e, x representa um conjunto de preditores.
data	Um quadro de dados contendo as variáveis especificadas no argumento “formula”.
subset	Argumento opcional especificando um subconjunto de observações a serem usadas.
nbagg	Um inteiro especificando o número de replicações *bootstrap*.
coob	Argumento lógico indicando se a estimativa OOB deve ser computada. (*default* = FALSE)
control	pode usar o argumento ‘rpart.control’ para controlar detalhes da função rpart(). Por exemplo, especificar ‘cp’, ‘minsplit’, ‘xval’.

FONTE: O autor (2020).

2.4.5 *Random Forest*

Um dos problemas com *“bagged trees”* é que as árvores construídas não são independentes uma das outras, já que todos os preditores originais são considerados em cada divisão das árvores construídas. Assim, é possível que as árvores no *ensemble* apresentem muitas semelhanças estruturais (especialmente no topo das árvores) devido à relação subjacente e, por conseguinte, as predições das árvores podem ser altamente correlacionadas (KUHN; JOHNSON, 2013). Destarte, calcular a média de muitas grandezas altamente correlacionadas não leva a uma redução tão grande na variância quanto a média de muitas quantidades não correlacionadas (JAMES *et al.*, 2013).

Uma vez que a presença de correlação de árvore impede que o *bagging* reduza de maneira ideal a variação dos valores preditos (KUHN; JOHNSON, 2013), um novo algoritmo conhecido como Floresta Aleatória (do inglês, *Random Forests*-RF) foi proposto por Leo Breiman em 2001 (BREIMAN, 2001a). O *Random Forests* é uma modificação substancial do *bagging*, e objetiva construir uma coleção de árvores não correlacionadas e, em seguida, calcular a média das predições individuais, no caso de regressão (HASTIE; TIBSHIRANI; FRIEDMAN, 2016).

O *Random Forest* possui algumas características desejáveis (BREIMAN, 2001a):

(a) É relativamente robusto para *outliers* e ruído;

(b) É mais rápido do que *bagging* ou *boosting*;

(c) Fornece estimativas internas de erro, força, correlação e importância variável; e

(d) É simples e facilmente paralelizado.

Assim, a principal diferença entre o *bagging* e o *Random Forest* é a escolha do tamanho k no espaço de preditores originais (JAMES *et al.*, 2013). Em regressão, um valor

típico para k é usar $p/3$ e um número mínimo de observações no nó de decisão igual 5. No entanto, em termos práticos, o melhor valor de k dependerá do problema e, portanto, k é considerado um hiperparâmetro de ajuste (HASTIE; TIBSHIRANI; FRIEDMAN, 2016). O algoritmo é relativamente simples e, possui apenas uma pequena, mas substancial mudança em relação ao *bagging*:

1. Dado um conjunto de treinamento original L de tamanho n, gerar uma amostra *bootstrap* (bi) de tamanho n;
2. Treinar um modelo de árvore de regressão (para o tamanho máximo e sem podar) sobre a amostra *bootstrap* bi. Porém, no processo de crescimento da árvore implementar a seguinte modificação:
   a) Em cada nó de decisão, selecionar um subconjunto aleatório de preditores, k;
   b) A melhor divisão será escolhida dentre os k preditores candidatos.
3. Salvar o modelo preditivo.

Esse procedimento pode ser realizado B vezes, em que B é a quantidade de amostras *bootstrap* pré-definida. Uma vez que todos os modelos foram construídos usando as B amostras, pode-se obter as predições de cada modelo. Então, uma predição combinada pode ser obtida através da média das predições dos modelos individuais treinados em cada amostra *bootstrap* bi. O pseudocódigo do *Random Forests* para tarefas de regressão pode ser facilmente compreendido (FIGURA 12).

FIGURA 12 – PSEUDOCÓDIGO PARA O ALGORITMO ‘Random Forest’ PARA PROBLEMAS DE REGRESSÃO.

Algoritmo 2 - ‘Random Forest’

```
1 Selecionar o número de modelos para construir, m
  para i = 1 até m faça
2     Gerar uma amostra de bootstrap dos dados originais
3     Treinar um modelo de árvore usando a amostra bootstrap
      para cada divisão faça
4         Selecionar aleatoriamente k (< P) dos preditores originais
5         Selecionar o melhor preditor entre os k preditores e particionar
          os dados
      fim
6     Usar os critérios de parada típicos do modelo de árvore para
      determinar quando uma árvore está completa (mas não podar)
  fim
  Retorna f̂_rf^B(x) = (1/B) Σ_{b=1}^{B} T_b(x)
```

FONTE: Adaptado de Kuhn e Johnson (2013).

FIGURA 13 – ESCOPO GERAL DA FUNÇÃO randomForest().

```
randomForest(x, y=NULL,  xtest=NULL, ytest=NULL, ntree=500,
             mtry=if (!is.null(y) && !is.factor(y))
             max(floor(ncol(x)/3), 1) else floor(sqrt(ncol(x))),
             replace=TRUE, classwt=NULL, cutoff, strata,
             sampsize = if (replace) nrow(x) else ceiling(.632*nrow(x)),
             nodesize = if (!is.null(y) && !is.factor(y)) 5 else 1,
             maxnodes = NULL,
             importance=FALSE, localImp=FALSE, nPerm=1,
             proximity, oob.prox=proximity,
             norm.votes=TRUE, do.trace=FALSE,
             keep.forest=!is.null(y) && is.null(xtest), corr.bias=FALSE,
             keep.inbag=FALSE, ...)
```

FONTE: Liaw e Wiener (2002).

A função randomForest() da biblioteca “randomForest”, acessível no ambiente estatístico R, disponibiliza a implementação original do algoritmo proposto por Breiman (2001a). O escopo geral da função randomForest() e seus principais argumentos são detalhados (FIGURA 13 e QUADRO 5) (LIAW; WIENER, 2002).

QUADRO 5 – Parâmetros da função randomForest().

Parâmetro	Descrição
data	Um quadro de dados opcional contendo as variáveis no modelo.
subset	Argumento opcional especificando um subconjunto de observações a serem usadas.
x	Um quadro de dados ou uma matriz de preditores, ou uma fórmula descrevendo o modelo a ser ajustado.
y	Um vetor com os valores da variável resposta. Se for um fator, assume-se que o problema é de classificação, caso contrário assume-se que é regressão. Se NULL, o randomForest será executado no modo não supervisionado.
ntree	Número de árvores a serem crescidas. (*default*=500).
mtry	número de preditoras amostradas aleatoriamente como candidatas em cada divisão. Para regressão, o padrão é $p/3$. Em que, p é o número de variáveis em x.
nodesize	número mínimo de observações nos nós terminais. Valor alto implica em árvores menos profundas (< custo computacional). Para regressão, o *default* é 5.
importance	argumento lógico indicando se a importância dos preditores deve ser avaliada. (*default*=FALSE).

FONTE: O autor (2020).

2.4.6 *Stochastic Gradient Boosting* - SGB

O *boosting* constitui outra abordagem *ensemble* para melhorar o desempenho de modelos preditivos. O algoritmo ‘AdaBoost’, abreviação de *Adaptive Boosting*, foi a primeira implementação bem sucedida da classe de modelos *boosting*, introduzida por Yoav Freund e Rob Schapire em 1997, e fortemente idealizada para problemas de classificação binária (FREUND; SCHAPIRE, 1997).

A ideia base do 'AdaBoost' é ajustar sequencialmente classificadores "fracos" (*weak*) a diferentes pesos das observações em um conjunto de dados. Em cada iteração, uma avaliação do desempenho é realizada e, aquelas observações mal preditas pelo classificador prévio recebem maior peso na próxima iteração. O classificador 'AdaBoost' final é uma média ponderada de todos os classificadores fracos (RIDGEWAY, 1999). Em síntese, a ideia é construir um preditor forte a partir de múltiplos modelos/aprendizes fracos (*weak learners*).

Usando uma conexão entre boosting e otimização (FRIEDMAN; HASTIE; TIBSHIRANI, 2000), o algoritmo conhecido como *Gradient Boosting Machine* - GBM foi proposto (RIDGEWAY, 1999). A nova abordagem facilitou várias generalizações algorítmicas do *boosting* para tarefas de classificação e, além disso permitiu que o método fosse estendido para problemas de regressão (KUHN; JOHNSON, 2013).

Basicamente, o GBM segue os seguintes princípios básicos: dada uma função de perda (ex., erro quadrático) e um aprediz fraco (ex., árvores de regressão), o algoritmo procura encontrar um 'modelo aditivo' que 'minimize a função de perda'. O algoritmo é tipicamente inicializado com o melhor palpite da resposta (por exemplo, a média da variável resposta na regressão). O gradiente (residual) é calculado e um modelo é então ajustado aos resíduos para minimizar a função de perda. O modelo atual é incluído no modelo anterior e o procedimento continua para um número de iterações especificado pelo usuário (KUHN; JOHNSON, 2013). O pseudocódigo para o algoritmo 'Gradient Boosting' simplificado para problemas de regressão é detalhado (FIGURA 14).

O *boosting* e, assim como *bagging* e *Random Forest*, usa comumente as árvores de decisão como aprendizes de base, porém algumas diferenças substanciais podem ser destacadas (JAMES *et al.*, 2013; KUHN; JOHNSON, 2013):

a) No *boosting* as árvores são crescidas sequencialmente, ou seja, cada árvore é crescida usando informações de árvores construídas anteriormente (dependência de árvores antigas). No *bagging* e *Random Forest* as árvores são criadas independentemente, o que é vantajoso para emprego de processamento paralelo;

b) O *boosting* não envolve amostragem *bootstrap* e/ou seleção de variáveis aleatórias (no caso de *Random Forest*);

c) As árvores no *boosting* são crescidas para ter profundidade mínima, ou seja, árvore menores são construídas, o que facilita a interpretabilidade. No *bagging* e *Random Forest* as árvores são crescidas para profundidade máxima, até que algum critério de parada seja satisfeito, e podas são realizadas; e

d) No *boosting* os preditores individuais contribuem de forma desigual para o modelo final.

FIGURA 14 – PSEUDOCÓDIGO PARA O ALGORITMO ‘Gradient Boosting’ PARA PROBLEMAS DE REGRESSÃO.

Algoritmo 3 - Gradient Boosting simplificado

1 Selecionar a profundidade da árvore, D, e o número de iterações, K
2 Calcular a resposta média, $\bar{y}$, e usar como o valor predito inicial para cada amostra
para *k = 1 até* K **faça**
3 Calcular o residual para cada amostra - diferença entre o valor observado e predito atual
4 Ajustar uma árvore de regressão de profundidade D, usando os resíduos como resposta
5 Prever cada amostra usando a árvore de regressão ajustada na etapa anterior
6 Atualizar o valor previsto de cada amostra, adicionando o valor previsto da iteração anterior ao valor previsto gerado na etapa anterior
fim

FONTE: Adaptado de Kuhn e Johnson (2013).

Motivado pelas ideias expressas na pesquisa de Breiman (1999), Jerome Friedman propôs uma pequena modificação ao GBM para incorporar um “componente de aleatoriedade” como parte integrante do algoritmo. A nova versão do algoritmo foi denominada “Aumento de Gradiente Estocástico” (do inglês, *Stochastic Gradient Boosting* - SGB) . Especificamente, a cada iteração, uma subamostra (ou fração) dos dados de treinamento é selecionada aleatoriamente (sem reposição) do conjunto de treinamento completo. Essa fração de dados, ao invés da amostra completa, é usada para ajustar um aprediz fraco e calcular a atualização do modelo para a iteração atual (FRIEDMAN, 2002).

Assim, essa simples modificação introduzida ao GBM melhorou o desempenho preditivo, reduziu a possibilidade de *overfitting* e também o custo computacional (FRIEDMAN, 2002; KUHN; JOHNSON, 2013). Em SGB, a estocasticidade é controlada por meio de uma "fração de amostra"que especifica a proporção de dados a serem selecionados em cada etapa. A fração de dados de treinamento usada no SGB, é conhecida como fração de amostra (do inglês, *bag fraction*). A fração de amostra usual é 0,5, o que significa que, a cada iteração, 50% dos dados são sorteados aleatoriamente, sem substituição, do conjunto completo de treinamento (ELITH; LEATHWICK; HASTIE, 2008). Porém, embora o valor 0,5 seja recomendado, a fração de amostra é um hiperparâmetro de ajuste na nova versão proposta (KUHN; JOHNSON, 2013) e, seu valor ideal pode ser determinado por reamostragem.

Como todo algoritmo de aprendizado de máquina, o *boosting* possui hiperparâmetros de ajuste. Algumas implementações modernas, como a disponível no pacote ‘gbm’ (GREENWELL *et al.*, 2019), têm considerado diversos hiperparâmetros. Três principais são destacados por James *et al.* (2013), que podem ser escolhidos por algum método de reamostragem, como validação cruzada:

1. **número de árvores**, B (ou iterações K, como no algoritmo 4). Ao contrário do *bagging* e do *Random Forest*, o *boosting* pode sobreajustar caso B seja muito grande, embora o *overfitting* tenda a ocorrer lentamente, se for o caso. Usamos validação cruzada para selecionar B;

2. **parâmetro *shrinkage***, λ . Hiperparâmetro que controla a taxa de aprendizagem do *boosting*. Os valores usuais são 0,01 ou 0,001. Porém, a escolha ideal depende do problema. Em geral, λ pequeno pode requerer o aumento de B para alcançar um bom desempenho preditivo;

3. **número de divisões em cada árvore**, d (ou profundidade D, como no algoritmo 4). Hiperparâmetro que controla a complexidade das árvores no *boosting*. Muitas vezes $d = 1$ funciona bem. Isso implica em árvores com uma única divisão (dois nós terminais).

As versões *Gradiente Boosting Machine* e *Stochastic Gradient Boosting* podem ser acessadas através do pacote 'gbm', que dispõe da função gbm() para treinar modelos preditivos (GREENWELL *et al.*, 2019). A descrição dos principais parâmetros da função está disponível (QUADRO 6).

FIGURA 15 – ESCOPO GERAL DA FUNÇÃO gbm().

```
gbm(formula = formula(data), distribution = "bernoulli",
  data = list(), weights, var.monotone = NULL, n.trees = 100,
  interaction.depth = 1, n.minobsinnode = 10, shrinkage = 0.1,
  bag.fraction = 0.5, train.fraction = 1, cv.folds = 0,
  keep.data = TRUE, verbose = FALSE, class.stratify.cv = NULL,
  n.cores = NULL)
```

FONTE: Greenwell *et al.* (2019).

QUADRO 6 – Parâmetros da função gbm().

Parâmetro	Descrição
formula	Uma descrição simbólica do modelo a ser ajustado.
distribution	Uma cadeia de caracteres especificando o nome da distribuição a ser considerada.
data	Um quadro de dados opcional contendo as variáveis no modelo.
n.trees	Número total de árvores (ou número de iterações). (*default* = 100).
interaction.depth	Número inteiro especificando a profundidade máxima de cada árvore. (*default* = 1).
n.minobsinnode	Número inteiro especificando o número mínimo de observações nos nós terminais das árvores. (*default* = 10).
shrinkage	Taxa de aprendizado ou tamanho do passo. Na prática, controla a velocidade com que o algoritmo continua descendo o gradiente. Normalmente, taxas de aprendizado pequenas requerem o crescimento de maior quantidade de árvores (n.trees) para encontrar o mínimo. (*default* = 0,1).
bag.fraction	Fração de observações do conjunto de treinamento selecionadas aleatoriamente para crescer a árvore. Esse procedimento introduz um componente de aleatoriedade no ajuste do modelo. Usar menos de 100% das observações de treinamento implica no uso da implementação do algoritmo *Stochastic Gradient Boosting*. (*default* = 0,5).

FONTE: O autor (2020).

A taxa de aprendizado (λ) é usada para reduzir a contribuição de cada árvore à medida que é adicionada ao modelo *boosting*. Diminuir λ aumenta o número de árvores B necessárias. Em geral, um menor λ e maior B (número de árvores) é preferível, a depender do número de observações e do tempo disponível para cálculo. A complexidade das árvores (*interaction.depth* na função gbm()) também afeta B, ou seja, se a complexidade é aumentada, então menos árvores seriam necessárias para atingir um erro mínimo, e também um menor valor de λ deve ser ideal (ELITH; LEATHWICK; HASTIE, 2008).

2.4.7 *Extreme Gradient Boosting* - XGBoost

O algoritmo e**X**treme **G**radient **Boost**ing (XGBoost) é uma implementação eficiente e escalável do *framework Gradient Boosting Machine* (FRIEDMAN; HASTIE; TIBSHIRANI, 2000; FRIEDMAN, 2001), que foi desenvolvida por Tianqi Chen. Uma descrição completa e detalhada sobre o algoritmo e suas especificidades pode ser acessada na página da web mantida pelos desenvolvedores do XGBoost (Documentação XGBoost), e também uma breve apresentação do sistema XGBoost é realizada em Chen e Guestrin (2016). Em termos de velocidade, o XGBoost é 10 vezes mais rápido que o clássico GBM, com computação paralela automática. Além disso, suporta várias funções objetivos e também personalizações e é otimizado para matrizes esparsas (em uma matriz esparsa as células que contêm zero não são armazenadas na memória) (CHEN *et al.*, 2019).

O algoritmo têm sido amplamente usado por cientistas de dados para alcançar resultados de ponta em muitos desafios de aprendizado de máquina, como as competições da 'Kaggle' (CHEN; GUESTRIN, 2016). A preferência pelo XGBoost está associada ao excelente desempenho preditivo, à implementação altamente otimizada de máquinas multicore e distribuídas e à capacidade de lidar com dados esparsos (MITCHELL; FRANK, 2017). Em 2015, dentre as 29 soluções vencedoras de desafios, 17 usaram o XGBoost. O sucesso do sistema também foi testemunhado no 'KDDCup' 2015, onde o XGBoost foi usado por todas as equipes vencedoras no top 10 (CHEN; GUESTRIN, 2016).

No ambiente estatístico R, o XGBoost pode ser acessado através do pacote 'xgboost'. O pacote inclui um solucionador de modelo linear eficiente (gblinear) e um baseado no aumento/gradiente de árvores (gbtree). Duas funções podem ser usadas para treinar um XGBoost: xgb.train() ou xgboost(). Particularmente, xgb.train() admite apenas um objeto 'xgb.DMatrix' (classe própria de um XGBoost) como entrada. A função xgboost() aceita objetos das classes: 'xgb.DMatrix', 'matrix', 'dgCMatrix' ou nome de um arquivo de dados local (CHEN *et al.*, 2019). O escopo geral da função xgboost() (FIGURA 16) e seus principais argumentos estão descritos (QUADRO 7). Ainda, a lista e descrição detalhada dos parâmetros disponíveis pode ser acessada no link: Parâmetros XGBoost.

FIGURA 16 – ESCOPO GERAL DA FUNÇÃO xgboost().

```
xgboost(data = NULL, label = NULL, missing = NA, weight = NULL,
  params = list(), nrounds, verbose = 1, print_every_n = 1L,
  early_stopping_rounds = NULL, maximize = NULL, save_period = NULL,
  save_name = "xgboost.model", xgb_model = NULL, callbacks = list(),
  ...)
```

FONTE: Chen *et al.* (2019).

QUADRO 7 – Parâmetros da função xgboost().

Parâmetro	Descrição
data	conjunto de treinamento. A função xgb.train() aceita apenas um 'xgb.DMatrix' como entrada. A xgboost(), além disso, também aceita as classes 'matrix', 'dgCMatrix' ou nome de um arquivo de dados local.
label	Vetor da variável resposta. Não deve ser fornecido quando os dados são um nome de arquivo de dados local ou um 'xgb.DMatrix'.
nrounds	número máximo de iterações de reforço (para 'gbtree' é o número de árvores).
params	uma lista de parâmetros (Parâmetros XGBoost).
eta	Um argumento que deve ser passado para **params**, e controla a taxa de aprendizado ($0 < \text{eta} < 1$). Usado para prevenir *overfitting*, tornando o processo de reforço mais conservador. Menor valor para 'eta' requer um valor maior para 'nrounds'. (*default* = 0,3).
gamma	Um argumento que deve ser passado para **params**, e controla a redução mínima de perda necessária para fazer uma partição adicional em um nó da árvore. Quanto maior, mais conservador será o algoritmo.
max_depth	Um argumento que deve ser passado para **params**, e controla a profundidade máxima de uma árvore. (*default* = 6).
min_child_weight	Um argumento que deve ser passado para **params**, e constitui a soma mínima do peso da instância (hessian) necessária em um nó filho. Se a etapa de partição em árvore resultar em um nó folha com a soma do peso da instância menor do que min_child_weight, o processo de particionamento não continuará. (*default* = 1).
subsample	Um argumento que deve ser passado para **params**, e controla a fração de instâncias do conjunto de treinamento amostradas aleatoriamente. Por exemplo, usar 0,5 significa que a metade das instâncias de dados serão selecionadas para crescer as árvores. Particularmente, o procedimento auxilia à prevenir *overfitting*. (*default* = 1).
colsample_bytree	Um argumento que deve ser passado para **params**, e controla a proporção de subamostras de colunas usadas para crescer cada árvore. (*default* = 1).

FONTE: O autor (2020).

2.4.8 *Support Vector Regression* - SVR

O algoritmo conhecido como Máquinas de Vetores de Suporte (do inglês, *Support Vector Machine* - SVM) constitui uma técnica de modelagem poderosa e altamente flexível, originalmente desenvolvida para problemas de classificação (KUHN; JOHNSON, 2013). Em 1995, porém, Vapnik estendeu o SVM para solucionar tarefas de regressão, ao propor o algoritmo ϵ-SV (TORGO, 2017), método conhecido tradicionalmente por Regressão por Vetores de Suporte (do inglês, *Support Vector Regression* - SVR) (KAVAKLIOGLU, 2011).

A ideia básica do algoritmo ϵ-SV é encontrar uma função $f(x)$ que tenha no máximo ϵ desvios para os valores reais y_i do conjunto de treinamento e, ao mesmo tempo, que seja mais plana o possível (SMOLA; SCHÖLKOPF, 2004). Em outras palavras, o objetivo é encontrar um hiperplano cuja distância para todos os dados de treinamento seja no máximo ϵ (TORGO, 2017), ou seja, são admitidos apenas erros menores do que ϵ (SMOLA; SCHÖLKOPF, 2004).

A generalização do SVM para SVR é realizada através da introdução de uma região ϵ-insensível ao redor da função, chamada ϵ-tube (AWAD; KHANNA, 2015). Em ϵ-SV, uma nova função perda é introduzida: a perda ϵ-insensível, expressa na formulação 2.12 (SMOLA; SCHÖLKOPF, 2004).

$$|\xi|_\epsilon = \begin{cases} 0, & se \;\; |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon, & do \;\; contrário \end{cases} \tag{2.12}$$

Em ϵ-SV, para a situação mais geral em que são admitidos alguns casos fora do "envelope" (ou ϵ-tube) (cada um sujeito a uma penalidade C), tem-se o problema de minimização expresso em 2.13 para solucionar, sujeito às restrições dispostas em 2.14 (SMOLA; SCHÖLKOPF, 2004; TORGO, 2017). Em que: ξ_i e ξ_i^* são chamadas de variáveis de folga, que representam restrições dos dois lados de um hiperplano, e permitem que algumas amostras fiquem fora da região ϵ-tube.

$$Minimizar : \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \tag{2.13}$$

$$Sujeito \;\; a : \begin{cases} y_i - (\boldsymbol{w}.\boldsymbol{x}) - b & \leq \epsilon + \xi_i \\ (\boldsymbol{w}.\boldsymbol{x}) + b - y_i & \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \tag{2.14}$$

O modelo produzido pelo ϵ-SV depende apenas de um subconjunto dos dados de treinamento, uma vez que a função de custo para a construção do modelo ignora quaisquer dados de treinamento que estejam próximos (dentro de um limite ϵ) da previsão do modelo (BASAK; PAL; PATRANABIS, 2007).

No ambiente R, existem dois pacotes principais com implementações do algoritmo SVM: 'e1071' (função svm()) (MEYER *et al.*, 2019) e 'kernlab' (função ksvm()) (KARATZOGLOU *et al.*, 2004). Em se tratando de regressão, a ksvm() suporta as formulações ϵ-svr (regressão épsilon) e ν-svr (regressão nu). O escopo geral da função ksvm() é apresentado (FIGURA 17) e seus principais argumentos são detalhados (QUADRO 8).

FIGURA 17 – ESCOPO GERAL DA FUNÇÃO ksvm().

```
ksvm(x, y = NULL, scaled = TRUE, type = NULL,
     kernel ="rbfdot", kpar = "automatic",
     C = 1, nu = 0.2, epsilon = 0.1, prob.model = FALSE,
     class.weights = NULL, cross = 0, fit = TRUE, cache = 40,
     tol = 0.001, shrinking = TRUE, ...,
     subset, na.action = na.omit)
```

FONTE: Karatzoglou *et al.* (2004).

QUADRO 8 – Parâmetros da função ksvm().

Parâmetro	Descrição
x	Uma descrição simbólica do modelo a ser ajustado. Admite-se o uso de fórmula, uma matriz da classe 'kernelMatrix' ou uma lista de vetores.
data [opcional]	Um dataframe contendo os dados de treinamento, quando usado uma fórmula.
y	Um vetor de resposta numérico (regressão) ou fator (classificação).
scaled	Um vetor lógico indicando se as variáveis devem ser padronizadas (center e scale). (*default* = TRUE).
type	Especifica o tipo de tarefa. Para regressão, o padrão é 'eps-svr' (regressão épsilon). Outras opções para regressão são: 'nu-svr' (regressão nu) e 'eps-bsvr' (regressão svm bound-constraint).
kernel	Especifica a função do kernel a ser usada no treinamento e na predição. As opções disponíveis são: 'rbfdot' (Radial Basis kernel "Gaussian"), 'polydot' (Polynomial kernel), 'vanilladot' (Linear kernel), 'tanhdot' (Hyperbolic tangent kernel), 'laplacedot' (Laplacian kernel), 'besseldot' (Bessel kernel), 'anovadot' (ANOVA RBF kernel), 'splinedot' (Spline kernel) e 'stringdot' (String kernel).
kpar	Lista de hiperparâmetros kernel. Por exemplo, 'degree', 'scale', 'offset' para o kernel Polinomial ("polydot").
C	Custo de violação de restrições. Constitui a 'constante' C do termo de regularização na formulação de *Lagrange*. (*default* = 1).
nu	Parâmetro necessário para 'nu-svc', 'one-svc' e 'nu-svr'. O parâmetro 'nu' define o limite superior no erro de treinamento e o limite inferior na fração de pontos de dados para se tornar Vetores de Suporte. (*default* = 0,2).
epsilon	Função de perda insensível usada para 'eps-svr', 'nu-svr' e 'eps-bsvm'. (*default* = 0,1).
cross	Argumento para uso de validação cruzada. Se especificado um valor inteiro $k > 0$, o método *k-fold cross validation* é aplicado sobre os dados de treinamento. Para regressão, a métrica 'Erro Quadrático Médio' é usada.

FONTE: O autor (2020).

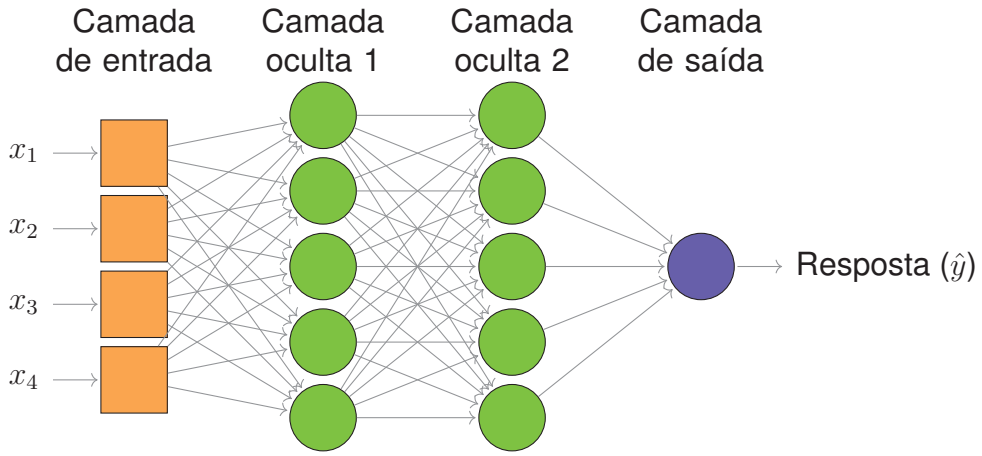
2.4.9 *Artificial Neural Networks* - ANN

As Redes Neurais Artificiais (do inglês, *Artificial Neural Networks* - ANN) foram introduzidas em 1943 por McCulloch e Pitts no trabalho intitulado "*A logical calculus of the ideas immanent in nervous activity*" (HAYKIN, 2001; DASE; PAWAR, 2010). Em 1958, Frank Rosenblat introduziu o conceito de aprendizado supervisionado em redes neurais ao propor o **modelo Perceptron**, que possuía uma arquitetura de rede, tendo como unidades básicas **neurônios MCP** (McCulloch e Pitts) (HAYKIN, 2001; BRAGA; CARVALHO; LUDERMIR, 2016). Algum tempo depois, Frank Rosenblat demonstrou que um neurônio 'MCP' treinado com o algoritmo Perceptron sempre converge caso o problema seja linearmente separável, o conhecido **teorema de convergência do Perceptron** (BRAGA; CARVALHO; LUDERMIR, 2016).

Na década de 1970, ocorreu um grande desinteresse nas pesquisas sobre RNAs. Esse desinteresse esteve fortemente associado às críticas de Minsky e Papert, em 1969, que discutiram a limitação do Perceptron à problemas linearmente separáveis (GAMA *et al.*, 2015). Além disso, o desconhecimento de algoritmos para treinamento de redes com uma ou mais camadas intermediárias (ou ocultas) (BRAGA; CARVALHO; LUDERMIR, 2016), e a necessidade de alto poder computacional para o emprego de RNAs, o que não estava disponível na época, constituíram também fatores para o desentusiasmo (NASTOS *et al.*, 2013). No entanto, o interesse por RNAs foi novamente impulsionado, em meados da década de 80, em parte pela proposição do algoritmo de retropropagação de erros (do inglês, *back-propagation error*) (BRAGA; CARVALHO; LUDERMIR, 2016), e também pelo surgimento de computadores mais rápidos (GAMA *et al.*, 2015).

As redes do tipo **Perceptron de Múltiplas Camadas** (do inglês, *Multilayer Perceptron* - MLP) incluem três tipos de camadas: uma camada de entrada, uma ou mais camadas ocultas (ou intermediárias) e uma camada de saída (FATH; MADANIFAR; ABBASI, 2018). A arquitetura mais comum para uma MLP é a completamente conectada (ou densa), ou seja, todos os neurônios de uma camada C são conectados a todos os demais neurônios da camada $C + 1$. Admitindo que C é a primeira camada oculta, cada um dos neurônios de C é conectado a todos os atributos do espaço de preditores (GAMA *et al.*, 2015). Uma representação simples da arquitetura de uma rede perceptron de múltiplas camadas é apresentada (FIGURA 18).

FIGURA 18 – REPRESENTAÇÃO DA ARQUITETURA DE UMA REDE PERCEPTRON DE MÚLTIPLAS CAMADAS COM DUAS CAMADAS OCULTAS.



FONTE: O autor (2020).

O *back-propagation* (retropropagação de erro) é o algoritmo mais popular para treinamento de uma rede MLP. O algoritmo é baseado na regra de “aprendizagem por correção de erro” (HAYKIN, 2001). O treinamento ocorre em duas fases (BRAGA; CARVALHO; LUDERMIR, 2016):

(a) **Fase forward** (ou propagação): dado um padrão de entrada, esta fase consiste em obter uma saída e, por extensão o erro de saída, após a propagação do sinal por todas as camadas da rede (Elemento 9); e

(b) **Fase backward** (ou retropropagação): usa a saída desejada e a saída predita para atualizar os pesos das conexões (Elemento 10).

Diferentes hiperparâmetros de ajuste são possíveis para uma RNA, dependendo da implementação. O ambiente R dispõe de diversos pacotes para construir modelos de redes

QUADRO 9 – Descrição da fase *forward*.

Fase *forward*

1. O vetor de entrada **x** é apresentado às entradas da rede, e as saídas dos neurônios da primeira camada escondida C_1 são calculadas;
2. As saídas da camada escondida C_1 proverão as entradas da camada seguinte, C_2. O processo se repete até à camada de saída C_k; e
3. As saídas produzidas pelos neurônios da camada de saída são então comparadas às saídas desejadas y_d para aquele vetor de entrada **x**, e o erro correspondente y_d - y é calculado.

QUADRO 10 – Descrição da fase *backward*.

Fase *backward*

1. O erro da camada de saída C_k é utilizado para ajustar diretamente os seus pesos, usando do gradiente descendente do erro;
2. Os erros dos neurônios da camada de saída C_k são propagados para a camada anterior C_{k-1}, utilizando-se dos pesos das conexões entre as camadas, que serão multiplicados pelos erros correspondentes. Assim, têm-se um valor de erro estimado para cada neurônio da camada escondida que representa uma medida da influência de cada neurônio na camada C_{k-1} no erro de saída da camada C_k;
3. Os erros calculados para os neurônios da camada C_{k-1} são então usados para ajustar os seus pesos pelo gradiente descendente, analogamente ao procedimento utilizado para a camada C_k; e
4. O processo é repetido até que os pesos da camada C_1 sejam ajustados, concluindo-se assim o ajuste dos pesos de toda a rede para o vetor de entrada **x** e sua saída desejada y_d.

neurais artificiais, com destaque para ‘nnet’ (VENABLES; RIPLEY, 2002), ‘neuralnet’ (FRITSCH; GUENTHER; WRIGHT, 2019), ‘RSNNS’ (BERGMEIR; BENÍTEZ, 2012), ‘h2o’. O pacote ‘nnet’, usado neste estudo, dispõe da função nnet() para ajustar **redes neurais de camada única alimentadas para a frente** (*feedforward*) (VENABLES; RIPLEY, 2002), estrutura mais simples (FIGURA 19). Alguns dos principais argumentos são descritos (QUADRO 11).

FIGURA 19 – ESCOPO GERAL DA FUNÇÃO nnet().

```
nnet(x, y, weights, size, Wts, mask,
    linout = FALSE, entropy = FALSE, softmax = FALSE,
    censored = FALSE, skip = FALSE, rang = 0.7, decay = 0,
    maxit = 100, Hess = FALSE, trace = TRUE, MaxNWts = 1000,
    abstol = 1.0e-4, reltol = 1.0e-8, ...)
```

FONTE: Venables e Ripley (2002).

QUADRO 11 – Parâmetros da função nnet().

Parâmetro	Descrição
x	Matriz ou quadro de dados de valores x.
y	Matriz ou quadro de dados para valores de resposta.
linout	Operador lógico que define a função de ativação. Para regressão usar 'TRUE'. (*default* = logística ou sigmóide).
decay	Parâmetro de decaimento de pesos. Parâmetro de regularização para prevenir *overfitting*. (*default* = 0).
maxit	Número máximo de iterações ou atualizações de pesos durante a construção do modelo. (default = 100).
size	Número de neurônios na camada oculta (intermediária).
rang	pesos aleatórios iniciais.

FONTE: O autor (2020).

3 METODOLOGIA

3.1 BIOMASSA DE ÁRVORES EM FLORESTAS TROPICAIS

3.1.1 Conjunto de dados

Nesta pesquisa, foi utilizada uma base de dados global compilada de diversos estudos conduzido por ecologistas ou silvicultores experientes. O conjunto de dados possui 4004 árvores-amostras (diâmetro $\geq$ 5cm) colhidas e distribuídas em 58 sítios de diferentes países, abrangendo uma variedade de condições climáticas e tipos de vegetação (53 locais com vegetações não perturbada e cinco com florestas secundárias). Neste conjunto estão incluídos dados de Chave *et al.* (2005) em que a densidade básica da madeira, o diâmetro e a atura total das árvores foram medidas. Assim, o conjunto de dados abrangeu as variáveis: BAT = biomassa aérea total, em kg (do inglês, *Above-Ground Biomass* - *AGB*), ρ = densidade básica da madeira, em g.cm^{-3} (do inglês, *Wood Specific Gravity* - *WSG*); d = diâmetro a 1,30m do solo, em cm; h = altura total das árvores, em m; e local de coleta. Além disso, incluiu-se no conjunto de dados original a variável localização (Sudeste Asiático e Austrália, América Latina, África) baseado na tabela S1 disponível como documento suplementar disponível em Chave *et al.* (2014). Para quantificação da biomassa aérea total foi utilizado o método destrutivo de árvores individuais, e obtido o peso ou volume fresco (no caso de árvores de grande porte) das seções. O peso fresco foi convertido em peso seco por meio do teor de umidade médio dos diferentes compartimentos da árvore. O volume fresco foi multiplicado pela densidade básica da madeira para obter o peso seco (CHAVE *et al.*, 2014). Para informações mais detalhadas sobre a origem dos dados compilados e dúvidas metodológicas recomenda-se consultar as pesquisas originais de Chave *et al.* (2005) e de Chave *et al.* (2014).

3.1.2 Modelo Pantropical

Na busca por modelos alométricos mais acurados e genéricos para estimativa de biomassa da parte aéreas de árvores em florestas tropicais, Chave *et al.* (2014) ajustaram modelos alométricos usando as mesmas 4004 árvores usadas neste estudo e encontraram no modelo linear logaritmo $ln(AGB) = \beta_0 + \beta_1\ ln(\rho\ D^2H) + \epsilon_i$ o melhor desempenho (CV$_{(j)}$ = 56,5% e Bias$_{(j)}$ = 5,31%) na predição da biomassa da parte aérea em florestas tropicais. A estrutura final do modelo ficou $AGB_{est.} = 0,0673\ (\rho\ D^2H)^{0,976}$ (σ = 0,357; *AIC* = 3130; *DF* = 4002), sendo designado de **Modelo Pantropical** (MP) . Em que: ρ representa a densidade básica da madeira, em g.cm^{-3}. O MP ajustado encontra-se disponível para uso fácil e prático no pacote BIOMASS por meio da função computeAGB(), bem como a incerteza associada às estimativas de BAT (RÉJOU-MÉCHAIN *et al.*, 2017a,b).

Um dos objetivos deste estudo foi avaliar o desempenho de modernos algoritmos de aprendizado de máquina (ver seção 2.4) para a predição da biomassa de árvores. Para além disso, interessou-se também em fazer uma comparação frente ao MP, já que foi construído usando a mesma base de dados. Devido a isso, para uma comparação sem tendência optou-se por replicar de forma fidedigna a modelagem preditiva feita por Chave *et al.* (2014), com intuito de obter as estimativas de BAT fiéis à análise original. De tal modo, obteve-se os coeficientes originais do MP (modelo 4 em Chave *et al.* (2014)) (β_0 = -2,7628 e β_1 = 0,9759) e as estimativas de BAT para cada árvore corrigidas pelo fator de Baskerville (1972).

Para tomada de decisão sobre o melhor modelo alométrico Chave *et al.* (2014) obtiveram as medidas de viés (Bias$_{(j)}$) e coeficiente de variação (CV$_{(j)}$) de cada sítio *j* por meio das equações 3.1 e 3.2, respectivamente. Em que: BAT$_{obs(j)}$ ou MBAT$_{(j)}$ = média empírica da

BAT no sítio *j*; BAT$_{est(j)}$ ou BAT$_{est(i,j)}$ = BAT estimada de cada árvore *i* no sítio *j*; BAT$_{obs(i,j)}$ = BAT observada de cada árvore *i* no sítio *j*; N$_j$ = número de árvores no sítio *j*; p = número de parâmetros do modelo estatístico. Na comparação do modelo de aprendizagem de máquina de melhor desempenho preditivo com o MP, replicou-se a abordagem de cálculo do Bias$_{(j)}$ e CV$_{(j)}$ empregada por Chave *et al.* (2014). A fórmula para obter o CV$_{(j)}$ é equivalente a medida rRMSE (do inglês, *Relative Root Mean Square Error*) bastante usual em aprendizado de máquina (MAUYA *et al.*, 2015; TANAKA *et al.*, 2014), com exceção da subtração do número de parâmetros p.

$$Bias_{(j)} = \frac{\left(BAT_{est(j)} - BAT_{obs(j)}\right)}{BAT_{obs(j)}} \tag{3.1}$$

$$CV_{(j)} = \frac{100}{MBAT_j}\sqrt{\frac{1}{N_j - p}\sum_{i\in(j)}\left(BAT_{est(i,j)} - BAT_{obs(i,j)}\right)^2} \tag{3.2}$$

O uso de Modelos Lineares Generalizados (MLG), supondo distribuições pertencentes à família exponencial para variável resposta, também foi considerado. Dois preditores lineares foram ajustados: $\eta = \beta_0 + \beta_1 ln(D) + \beta_2 ln(H) + \beta_3 ln(\rho)$ ou $\eta = \beta_0 + \beta_1 ln(\rho\ D^2 H)$, com diferentes funções de ligação (identidade, log ou inversa). No entanto, o uso de MLG, apesar de ser mais flexível do que a RL, foi desconsiderado pois não resolveu o problema da distribuição residual não Gaussiana. Por exemplo, os gráficos meio-normais com envelopes de simulação construídos com uso do pacote "hnp" (MORAL; HINDE; DEMÉTRIO, 2017) mostram evidências de que o modelo com resposta Gamma foi inadequado, uma vez que vários pontos estão fora do envelope simulado. Esse padrão foi também observado para outras distribuições contínuas da família exponencial.

3.2 VOLUME COMERCIAL DE ÁRVORES EM FLORESTA NATURAL INEQUIÂNEA

3.2.1 Conjunto de dados

As informações de volume de espécies comerciais foram obtidas por meio de cubagem no processo de romaneio em Unidades de Produção Anual (UPAs) do Plano de Manejo da Floresta Nacional do Jamari. A Floresta Nacional do Jamari é uma Unidade de Conservação (UC) situada no bioma Amazônia, em Rondônia, com área de 222.156,58 hectares, criada pelo decreto n. 90.224, de 25 de setembro de 1984. Para mais informações sobre a Flona do Jamari pode acessar o site do Instituto Chico Mendes de Conservação da Biodiversidade (ICMBio).

O conjunto de dados possui 13.831 árvores-amostras de 38 espécies comerciais Amazônicas pertencentes a 14 famílias botânicas. Do total de espécies, 16 pertencem à Fabaceae (TABELA 4). Estão disponíveis as seguintes variáveis: v = Volume comercial com casca, em m³; d = diâmetro a 1,30m do solo, em cm; h = Altura comercial, em m; e nomes científicos das espécies florestais. Uma descrição detalhada das espécies (distribuição geográfica, domínios fitogeográficos, tipo de vegetação de ocorrência, entre outros) pode ser consultada no site Flora do Brasil 2020.

A cubagem em romaneio foi realizada usando do método de *Smalian*. Pelo método de *Smalian* o volume de uma "tora" ou "seção" é obtido através da média aritmética das áreas seccionais dos extremos das seções pelo seu comprimento "*l*". Então, o volume de cada seção do fuste pode ser determinado pela equação 3.3. Em que: v = volume da tora (seção); g_1 = área seccional na base da tora, em m²; g_2 = área seccional no topo da tora, em m²; l = comprimento

da tora, em metros; d_1^2 (c_1^2) = diâmetro (ou circunferência) na base da tora, em m; e d_2^2 (c_2^2) = diâmetro (ou circunferência) no topo da tora, em m.

$$v = \frac{\left(g_1 + g_2\right)}{2} l \qquad \begin{cases} g_1 = \frac{\pi d_1^2}{4} = \frac{c_1^2}{4\pi} \\ g_2 = \frac{\pi d_2^2}{4} = \frac{c_2^2}{4\pi} \end{cases} \tag{3.3}$$

TABELA 4 – LISTA DE ESPÉCIES FLORESTAIS COMERCIAIS DA AMAZÔNIA BRASILEIRA AMOSTRADAS NA CUBAGEM EM ROMANEIO.

Nome científico	Família	N
Allantoma decandra (Ducke) S.A.Mori, Y.-Y.Huang & Prance	Lecythidaceae	282
Apuleia leiocarpa (Vogel) J.F.Macbr.	Fabaceae	759
Astronium lecointei Ducke	Anacardiaceae	1507
Bagassa guianensis Aubl.	Moraceae	163
Bowdichia nitida Spruce ex Benth.	Fabaceae	187
Brosimum rubescens Taub.	Moraceae	212
Cariniana micrantha Ducke	Lecythidaceae	454
Caryocar glabrum (Aubl.) Pers.	Caryocaraceae	254
Caryocar villosum (Aubl.) Pers.	Caryocaraceae	132
Cedrela fissilis Vell.	Meliaceae	115
Cedrelinga cateniformis (Ducke) Ducke	Fabaceae	134
Clarisia racemosa Ruiz & Pav.	Moraceae	675
Cordia goeldiana Huber	Boraginaceae	173
Couratari stellata A.C.Sm.	Lecythidaceae	1364
Dinizia excelsa Ducke	Fabaceae	984
Diplotropis rodriguesii H.C.Lima	Fabaceae	109
Dipteryx alata Vogel	Fabaceae	113
Dipteryx odorata (Aubl.) Willd.	Fabaceae	941
Erisma bicolor Ducke	Vochysiaceae	291
Erisma fuscum Ducke	Vochysiaceae	210
Goupia glabra Aubl.	Goupiaceae	429
Handroanthus incanus (A.H.Gentry) S.Grose	Bignoniaceae	291
Hymenaea intermedia Ducke	Fabaceae	35
Hymenaea oblongifolia var. palustris (Ducke) Y.T.Lee & Langenh.	Fabaceae	64
Hymenolobium heterocarpum Ducke	Fabaceae	904
Hymenolobium modestum Ducke	Fabaceae	63
Iryanthera paradoxa (Schwacke) Warb.	Myristicaceae	79
Martiodendron elatum (Ducke) Gleason	Fabaceae	82
Mezilaurus synandra (Mez) Kosterm.	Lauraceae	54
Peltogyne paniculata Benth.	Fabaceae	1712
Peltogyne venosa subsp. densiflora (Spruce ex Benth.) M.F.Silva	Fabaceae	50
Pouteria eugeniifolia (Pierre) Baehni	Sapotaceae	49
Pouteria guianensis Aubl.	Sapotaceae	152
Qualea paraensis Ducke	Vochysiaceae	542
Simarouba amara Aubl.	Simarorubaceae	32
Handroanthus impetiginosus (Mart. ex DC.) Mattos	Bignoniaceae	121
Vatairea guianensis Aubl.	Fabaceae	49
Vataireopsis speciosa Ducke	Fabaceae	64

FONTE: O autor (2020).

NOTA: Os nomes científicos das espécies são hiperlinks para o site **Flora do Brasil 2020**.

3.2.2 Modelos volumétricos tradicionais

Na Mensuração Florestal, é habitual usar a técnica de regressão clássica para modelar o volume de árvores individuais. Aqui, em particular, 10 formas funcionais de uso tradicional foram admitidas para modelagem do volume comercial com casca de espécies manejadas em florestas naturais inequiâneas (TABELA 5). Comumente, na área de domínio, as equações de volume são classificadas em modelos de simples ou dupla entrada. Os modelos de simples entrada incorporam apenas o diâmetro a 1,30m do solo (d) como variável preditora (em sua forma natural ou transformada), enquanto os modelos de dupla entrada incluem o d e a altura (h) da árvore, bem como suas transformações e interações entre variáveis.

TABELA 5 – MODELOS VOLUMÉTRICOS TRADICIONAIS SELECIONADOS PARA AJUSTE.

Símbolo	**Modelo estatístico**	**Autor**
Bk	$v = \beta_0 + \beta_1 d + \epsilon_i$	*Berkhout*
K-G	$v = \beta_0 + \beta_1 d^2 + \epsilon_i$	*Kopezky-Gerhardt*
H-K	$v = \beta_0 + \beta_1 d + \beta_2 d^2 + \epsilon_i$	*Hohenadl-Krenn* (1944)
H	$ln(v) = \beta_0 + \beta_1 ln(d) + \epsilon_i$	*Husch*
B	$ln(v) = \beta_0 + \beta_1 ln(d) + \beta_2 (1/d) + \epsilon_i$	*Brenac*
S	$v = \beta_0 + \beta_1 (d^2 h) + \epsilon_i$	*Spurr* (1952)
S(ln)	$ln(v) = \beta_0 + \beta_1 ln(d^2 h) + \epsilon_i$	*Spurr* (ln)
S-H	$ln(v) = \beta_0 + \beta_1 ln(d) + \beta_2 ln(h) + \epsilon_i$	*Schumacher-Hall* (1933)
N	$v = \beta_0 + \beta_1 d^2 + \beta_2 (d^2 h) + \beta_3 (d h^2) + \beta_4 h^2 + \epsilon_i$	*Näslund*
M	$v = \beta_0 + \beta_1 d + \beta_2 d^2 + \beta_3 (dh) + \beta_4 (d^2 h) + \beta_5 h + \epsilon_i$	*Meyer*

FONTE: O autor (2020).
NOTA: Os modelos foram extraídos de Loetsch, Zohrer e Haller (1973) e Prodan (1997).
LEGENDA: v = volume (m³); d = diâmetro a 1,30m do solo (cm); h = altura total (m); β_0, β_1, β_2, β_3, β_4 e β_5 = coeficientes da regressão; ln = logaritmo neperiano; e ϵ_i = termo de erro estocástico.

3.2.3 Estimação de parâmetros e seleção de modelos

A estimação de parâmetros dos modelos volumétricos (simples e dupla entrada) foi realizada através do Método de Mínimos Quadrados Ordinários (MQO) , procedimento usual em regressão linear clássica. O ajuste dos modelos volumétrico foi realizado usando a função lm() do pacote 'stats' disponível no ambiente estatístico R (R CORE TEAM, 2019).

A escolha do melhor modelo volumétrico genérico baseou-se nos seguintes critérios estatísticos de qualidade de ajuste: coeficiente de determinação ajustado (R^2_a) (do inglês, *Adjusted R-squared*) (Eq. 3.5), Erro Padrão Residual (do inglês, *Residual Standard Error* - RSE) - absoluto (Eq. 3.6) e relativo (Eq. 3.7), Critério de Informação de Akaike (do inglês, *Akaike Information Criterion* - AIC). Modelos com maior R^2_a, e menor RSE e AIC são preferíveis.

Além disso, a estatística $PRESS$ (do inglês, *Prediction Residual Error Sum of Squares*) proposta por David Allen, em 1974, foi calculada para estimar a capacidade preditiva dos modelos de RL sobre amostras futuras. A estatística $PRESS$ é baseada na abordagem *Leave-One-Out Cross Validation* (LOOCV). Por este método, *n*-1 amostras são usadas para ajustar um modelo, e a única observação de fora é utilizada para obter uma estimativa imparcial do desempenho do modelo. O método continua iterativamente até que cada observação seja usada exatamente uma vez para validar um modelo ajustado. Assim, os erros de predição para as "amostras de fora´´ são elevados ao quadrado e somados para formar a estatística PRESS

(Eq. 3.8) (ALLEN, 1974). Uma implementação básica para obtenção da estatística PRESS usando a linguagem R está disponível (Apéndice D).

O teste *t*-Student (α = 0,05) foi usado para avaliar a significância dos parâmetros estimados (intercepto e angular). Os Intervalos de Confiança (IC) associados aos parâmetros estimados da RL foram calculados (Eq. 3.9). No caso de equações logarítmicas, as estimativas de volume individual foram corrigidas pelo Fator de Correção ($FC = \frac{RSE^2}{2}$) apresentado em Baskerville (1972), bastante usual em estudos sobre modelagem de variáveis biométricas, como volume (OLIVEIRA *et al.*, 2018) e biomassa (CHAVE *et al.*, 2014). Assim, as estimativas das equações logarítmicas foram corrigidas usando a expressão Eq. 3.10 (BASKERVILLE, 1972). Além disso, para os modelos logarítmicos, todas as medidas de qualidade de ajuste foram recalculadas e expressas na escala natural da variável.

$$R^2 = \frac{SQ_{Reg.}}{SQ_{Tot.}} = \frac{\sum_{i=1}^{n}\left(\hat{y}_i - \bar{y}\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2} \tag{3.4}$$

$$R^2_a = 1 - \left(1 - R^2\right)\left(\frac{n-1}{n-(p+1)}\right) \tag{3.5}$$

$$RSE\ (S_{yx}) = \sqrt{\frac{1}{n-p}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{3.6}$$

$$RSE\ (\%) = \left(\frac{\sqrt{\frac{1}{n-p}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}}{\bar{y}}\right) 100 \tag{3.7}$$

$$PRESS = \sum_{i=1}^{n}\left(y_i - \hat{y}_{i,-i}\right)^2 \tag{3.8}$$

$$IC^{\beta_i}_{0,95} = \left[\hat{\beta}_i - 1,96 \times EP(\hat{\beta}_i)\ ,\ \hat{\beta}_i + 1,96 \times EP(\hat{\beta}_i)\right] \tag{3.9}$$

$$\hat{y}_i = exp^{[\frac{RSE^2}{2} + \hat{\mu}]} \tag{3.10}$$

Em que: n = número de observações; $\hat{y}_i$ = valor predito para a *i*-ésima árvore na amostra; y_i = valor observado para a *i*-ésima árvore na amostra; $\bar{y}$ = média empírica da variável resposta; p = número de parâmetros do modelo; $SQ_{Reg.}$ = Soma de Quadrados da Regressão; $SQ_{Tot.}$ = Soma de Quadrados Totais. $\hat{\mu}$ = valor predito da variável resposta, em unidades logarítmicas. O RSE na Eq. 3.10 é também obtido em unidades logarítmicas. Na Eq. 3.8, $\hat{y}_{i,-i}$ = estimador da $E(y_i)$ excluída a *i*-ésima observação.

A realização de inferências (testes de hipóteses e intervalos de confiança) em Modelos Clássicos de Regressão Linear (MCRL) requer o atendimento de suposições (normalidade, homocedasticidade e independência) fixadas sobre o termo de erro estocástico (ϵ_i) para serem consideradas confiáveis. Portanto, os resíduos do modelo clássico de regressão linear foram

julgados quanto às pressuposições de homocedasticidade (*Breusch-Pagan*, α = 0,05) (ZEILEIS; HOTHORN, 2002), autocorrelação (*Durbin-Watson*, α = 0,05) (FOX; WEISBERG, 2019) e normalidade (*Jarque–Bera*, α = 0,05) (GAVRILOV; PUSEV, 2014). O teste de *Jarque–Bera* foi preferido devido o teste de *Shapiro-Wilk* possuir limitações para o tamanho da amostra ($3 \leq n \leq 5000$). Adicionalmente, os gráficos de resíduos padronizados - e_i dividido pela raiz quadrada do quadrado médio dos resíduos (Eq. 3.11) - versus valores ajustados (escala da variável dependente) e, também gráficos Quantil-Quantil (Q-Qplot) das distribuições residuais, que relaciona os quantis empíricos dos resíduos e os quantis teóricos (esperados).

$$Z_{(i)} = \frac{\hat{y_i} - y_i}{\sqrt{QMR}} \tag{3.11}$$

Em modelos múltiplos (dois ou mais preditores), o atendimento à hipótese de ausência de colinearidade (ou multicolinearidade) é um aspecto importante. Em termos informais, inexistência de colinearidade significa que nenhum dos regressores pode ser expresso como uma *combinação linear exata* (ou perfeita) dos demais regressores do modelo (GUJARATI; PORTER, 2011).

Em geral, existem várias regras práticas para detectar a presença de multicolinearidade, sendo a estatística "Fator de Inflação da Variância´´ (FIV) (do inglês, *Variance Inflation Factor* - VIF) (Eq. 3.12) um indicador bastante usual. Porém, não existe consenso entre pesquisadores sobre o limiar para multicolinearidade "problemática". Como regra prática, Gujarati e Porter (2011) citam VIF_j > 10 para identificar uma variável altamente colinear. Por outro lado, Sileshi (2014) é mais rigoroso ao considerar VIF_j > 5. Quanto maior o valor de VIF_j mais "problemática´´ ou colinear será a variável X_j (GUJARATI; PORTER, 2011).

$$VIF = \frac{1}{\left(1 - r_{23}^2\right)} \tag{3.12}$$

Em que: r_{23} = coeficiente de correlação entre os regressores X_1 e X_2.

A seguir estão os pacotes e funções do ambiente estatístico R, versão 3.6.0, utilizados para análises das pressuposições do modelo de regressão linear, existência de multicolinearidade e o grau de associação entre variáveis (TABELA 6).

TABELA 6 – MÉTODOS USADOS PARA TESTAR HIPÓTESES, MULTICOLINEARIDADE, CORRELAÇÃO E QUALIDADE DE AJUSTE DA REGRESSÃO LINEAR.

Método	Teste de hipótese	Pacote	Função	Autor
Breusch-Pagan	H_0: as variâncias dos erros não são diferentes; H_1: as variâncias dos erros são diferentes.	lmtest	bptest()	Zeileis e Hothorn (2002)
Durbin-Watson	H_0: inexistência de autocorrelação residual; H_1: existência de autocorrelação residual.	car	durbinWatsonTest()	Fox e Weisberg (2019)
Fator de Inflação de Variância	Atestar a existência de multicolinearidade	faraway	vif()	Faraway (2016)
Jarque–Bera (n > 5000)	H_0: os resíduos são Gaussianos; H_1: os resíduos não são Gaussianos.	normtest	jb.norm.test()	Gavrilov e Pusev (2014)
Critério de Informação de Akaike	Avaliar a qualidade de ajuste dos modelos	stats	AIC()	R Core Team (2019)
Correlação linear de Pearson	Medir o grau de associação entre duas variáveis	stats	cor() cor.test()	

FONTE: O autor (2020).

3.3 CONSTRUÇÃO DOS MODELOS DE APRENDIZADO DE MÁQUINA

Na seção 2.4 foram apresentados e descritos diversos pacotes disponíveis no ambiente estatístico R para o emprego de técnicas de aprendizado de máquina. É fato que cada pacote foi construído por diferentes contribuidores e, portanto, sob um sintaxe peculiar de cada idealizador. A grande diversidade de bibliotecas de aprendizado de máquina é um ponto bastante positivo, porém, Kuhn (2008) destaca que acompanhar as nuances sintáticas de cada função dos inúmeros pacotes é bastante dificultoso. Assim, treinar e comparar vários algoritmos sob uma mesma base metodológica, exige o conhecimento específico de vários pacotes e funções, e em algumas vezes uma programação extra para além das funções internamente disponíveis nas bibliotecas.

Felizmente, para superar esse problema, Max Kuhn e diversos contribuidores desenvolveram o *framework* CARET (***C****lassification* ***A****nd* ***Re****gression* ***T****raining*), que constitui um conjunto de funções que tentam simplificar o processo treinamento e previsão de diversos modelos de aprendizagem de máquina implementados no ambiente estatístico R (KUHN, 2008; KUHN; JOHNSON, 2013; KUHN, 2015b; KUHN *et al.*, 2016). Em síntese, o CARET surgiu com o objetivo de: i) eliminar diferenças sintáticas entre muitas funções de construção de modelos preditivos; ii) desenvolver um conjunto de abordagens semi-automatizadas para otimizar os valores de hiperparâmetros de ajuste; e iii) criar um pacote que possa ser facilmente estendido para sistemas de processamento paralelo (KUHN, 2008). Enfim, devido as facilidades e padronização do processo de treinamento e avaliação de modelos de aprendizado de máquina (MAM) , optou-se por usar a interface do pacote CARET para construir os modelos preditivos descritos na seção 2.4.

3.3.1 Engenharia de variáveis e importância de preditoras

A criação de novos preditores (covariáveis) é bastante comum na modelagem por regressão tradicional. Em aprendizado de máquina esta tarefa é comumente designada de "engenharia de variáveis" (*feature engineering*). Assim, a ideia básica da engenharia de recursos consiste em usar o conhecimento de domínio para criar recursos, buscando melhores representações nos preditores para explicar a variável dependente.

Neste estudo, foram criados novos preditores potencias e de uso comum em modelos alométricos para estimar biomassa (BROWN; GILLESPIE; LUGO, 1989; CHAVE *et al.*, 2005, 2014) e o volume comercial com casca de árvores (BARRETO *et al.*, 2014; TONINI; BORGES, 2015; CYSNEIROS *et al.*, 2017) em florestas naturais inequiâneas. Portanto, para o conjunto de dados de biomassa, além das preditoras originais (d, h, ρ), o espaço de covariáveis foi aumentado com a construção das seguintes medidas derivadas: d^2, $1/d^2$, d^2h, ρd^2h, ln(d), $[ln(d)]^2$, ln(h), ln(ρ), ln(ρd^2h) e ln(d^2h). Do mesmo modo, para o conjunto de dados de volume, além das preditoras originais (d, h), foram considerados os seguintes preditores: d^2, $1/d^2$, d^2h, ln(d), ln(h), h^2, dh^2, ln(d^2h) e dh. Além disso, o relacionamento de cada covariável e as variáveis respostas foi avaliado por meio de gráficos de dispersão, e traçadas curvas LOESS (do inglês, *Local Polynomial Regression*) não otimizadas.

Após criar as novas representações para o espaço de covariáveis avaliou-se a importância destas para os modelos de aprendizado de máquina. De modo geral, Kuhn e Johnson (2013) descrevem duas estratégias para avaliar a importância de uma variável preditora em tarefas de aprendizado de máquina: 1) aquelas que usam as informações do modelo ajustado; e 2) aquelas que não usam as informações do modelo ajustado. Quando não existir uma maneira específica de estimar a importância dos preditores baseado no modelo ajustado, uma abordagem de "filtro" pode ser usada, na qual a importância individual de cada preditor para a variável resposta é medida (KUHN, 2008).

Para alguns dos algoritmos descritos na seção 2.4 não existe uma abordagem específica baseada em modelo, a exemplo do algoritmo *Weighted k*-Nearest-Neighbor. Para estes casos, o uso de métricas independentes do modelo ajustado foi a abordagem escolhida para estimar a importância individual das covariáveis para a resposta. Para tanto, as funções 'filterVarImp' e 'varImp' do pacote CARET com a especificação do argumento 'nonpara = TRUE' podem ser usadas para modelos não-paramétricos. Por este método, o padrão é ajustar curvas suavizadas entre cada preditor e a variável resposta usando o método de regressão LOESS, permitindo, assim, obter estimativas dentro de uma janela de vizinhança local. A regressão LOESS possui o argumento 'span' que controla o tamanho da vizinhança e, portanto, o grau de suavização da regressão, e pode variar de 0 a 1. Aqui, usa-se o padrão (span=0.75) da função loess() do pacote stats do R-base (R CORE TEAM, 2019).

Por outro lado, preditores com percentUnique < 20 (para mais detalhes veja a documentação da função nearZeroVar() da biblioteca CARET) têm sua importância estimada por modelos de regressão linear. O parâmetro "percentUnique" é a porcentagem de dados exclusivos de uma determinada variável em relação ao total de observações. Kuhn (2008) expõe que após o ajuste dos modelos individuais os valores de coeficiente de determinação (R^2) são tomados e computados como uma medida relativa da importância da variável. Além disso, os valores de importância podem ser redimensionados para uma escala de 0 e 100 (transformação Min-Max).

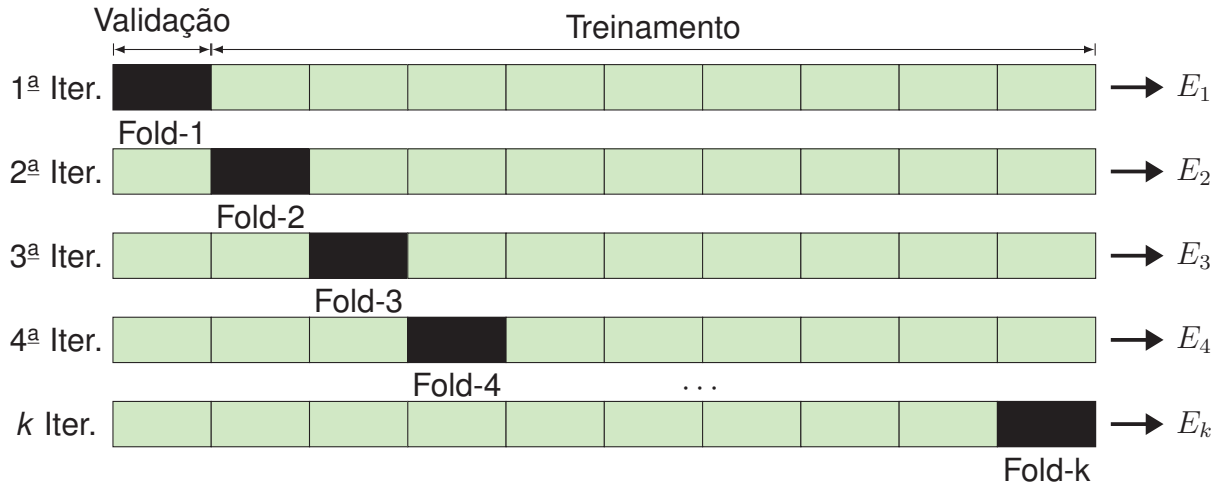
3.3.2 Método de reamostragem

Uma abordagem comum para estimar o desempenho esperado de um modelo preditivo quando submetido a novos dados é implementar algum método de reamostragem (*resampling method*) dos dados originais (MOLINARO; SIMON; PFEIFFER, 2005; KUHN; JOHNSON, 2013). O *k-fold cross-validation* (*k*-fold CV) é um método de reamostragem frequentemente usado para estimar a capacidade de generalização de um modelo de aprendizagem (VANWINCKELEN; BLOCKEEL, 2012; JIANG; WANG, 2017), muito embora, em vários problemas reais, o erro de predição não pode ser calculado com precisão porque a distribuição de probabilidade subjacente é desconhecida (JIANG; WANG, 2017). Os fatores que podem afetar a estimativa de desempenho no *k*-fold CV são: o número de partições (*folds*); o número de instâncias em cada dobra; o número de repetições de validação cruzada; e o nível de média (WONG, 2015). Uma representação esquemática do método *k-fold cross-validation* é apresentada (FIGURA 20).

Na abordagem *k*-fold CV todo o conjunto de dados é comumente estratificado antes de serem divididos em *k* subgrupos (*folds*) aproximadamente iguais (N/k) e distintos. Por este método, *k*-1 subgrupos são usados para treinar ou aprender um modelo, e o subgrupo mantido de fora (*hold-out set*) é usado para obter estimativas imparciais do desempenho de um modelo (REFAEILZADEH; TANG; LIU, 2009; AYER *et al.*, 2010; ISHIBUCHI; NOJIMA, 2013). Assim, o método continua iterativamente até que cada subgrupo seja usado exatamente uma vez para validar um modelo aprendido. O desempenho do modelo construído é medido para cada subgrupo mantido de fora e, em seguida, uma média aritmética é calculada (MOLINARO; SIMON; PFEIFFER, 2005; AYER *et al.*, 2010; VANWINCKELEN; BLOCKEEL, 2012; JAMES *et al.*, 2013; KUHN; JOHNSON, 2013; TORGO, 2017).

No que se refere a validação cruzada, existem argumentos de que uma única execução do *k*-fold CV geralmente produz alta variância em comparação a outros métodos e, por isso, pode não ser adequada (KUHN; JOHNSON, 2013; JIANG; WANG, 2017). Por outro lado, o *k*-fold CV repetido tem sido defendido sob a ideia de que o uso da média de repetidas validações cruzadas pode reduzir a variância do estimador *k*-fold CV (VANWINCKELEN; BLOCKEEL, 2012; LIANG; DAVIER, 2014), apesar do argumento de existência de sobreposição de dados de treino e validação a cada rodada do *k*-fold CV repetido (REFAEILZADEH; TANG; LIU,

FIGURA 20 – REPRESENTAÇÃO ESQUEMÁTICA DO MÉTODO DE REAMOSTRAGEM *k-FOLD CROSS-VALIDATION*.



FONTE: O autor (2020).

2009). Assim, repetir o procedimento de reamostragem pode produzir valores muito diferentes, porém se aplicado o suficiente estimará eficientemente o valor verdadeiro (KUHN; JOHNSON, 2013). Kim (2009), por exemplo, encontrou que o *k*-fold CV repetido superou o bootstrap, reduzindo a variância do estimador *k*-fold CV e, em razão disso, afirmou que o emprego de uma computação mais pesada pode ser válido. Molinaro, Simon e Pfeiffer (2005) por meio de simulações encontram que o estimador *k*-fold CV com 10 repetições teve desempenho próximo ao *leave-one-out cross-validation* (LOOCV), além de reduzirem o erro quadrado médio, o viés e a variância do 10-fold CV. Jiang e Wang (2017) inferiram que a variância de um estimador pode estar relacionada a variáveis relevantes, como tamanho da amostra, o número de subgrupos *k* e o número de repetições (no caso de *k*-fold CV repetido). Finalmente, para obter uma estimativa ou comparação de desempenho aceitável deve ser preferível um grande número de estimativa (YADAV; SHUKLA, 2016) e, ainda, que *k* = 10 seria um bom compromisso com a estimativa da precisão (REFAEILZADEH; TANG; LIU, 2009).

O pacote CARET dispõe de vários métodos de reamostragem para estimar o desempenho de um modelo preditivo (KUHN *et al.*, 2016). No entanto, apoiado nas evidências encontradas por Molinaro, Simon e Pfeiffer (2005) e Kim (2009), na modelagem da biomassa aérea total optou-se pelo uso do *k*-fold CV repetido (*repeated k-fold cross-validation*) sob o esquema 5x10-folds CV (5 repetições de 10-folds cross-validation) para obter estimativas imparciais do desempenho das variantes dos algoritmos de aprendizado de máquina. No *k*-fold CV repetido o conjunto de dados de aprendizado é reorganizado e repartido em *k*-folds a cada nova rodada da validação cruzada (REFAEILZADEH; TANG; LIU, 2009). Para isso, configurou-se a função trainControl() do CARET para implementar o método “repeatedcv”, de tal modo que se obteve para cada variante dos algoritmos estimativas de desempenho para 50 diferentes partições. Na modelagem do volume comercial, porém, com intuito de reduzir o custo computacional a abordagem *10*-fold CV tradicional (sem repetições) foi preferida.

3.3.3 Pré-processamento e ajuste de hiperparâmetros

Os algoritmos de aprendizado de máquina descritos na seção seção 2.4 possuem hiperparâmetros específicos que devem ser ajustados/sintonizados para encontrar a configuração de melhor desempenho preditivo. Para isso, como discutido na subseção 3.3.2, a abordagem mais comum é usar algum método de reamostragem dos dados originais. Neste estudo, todos

os modelos de aprendizado de máquina foram treinados e validados usando a interface do pacote CARET (KUHN *et al.*, 2016), do ambiente estatístico R (versão 3.6.0). Para realizar a modelagem preditiva o pacote dispõe da função train(), que foi configurada para a construção dos modelos. A função foi usada para:

- **Algoritmo de aprendizado**: Indicar o algoritmo de aprendizado de máquina a ser usado no processo de aprendizado dos dados. Isso é feito com uso do argumento 'method';
- **Pré-processamento**: Aplicar algum tipo de pré-processamento sobre as variáveis preditoras. Neste estudo, em particular, foram testadas as transformações: a) *center* and *scale*; b) *BoxCox*; c) *spatialSign*; e d) *YeoJohnson* (ver detalhes na TABELA 3 da seção seção 2.3);
- **Estimar modelos preditivos**: Estimar a capacidade preditiva para diferentes candidatos a hiperparâmetros de ajuste ótimo (*optimal tuning hyperparameters*), com uso de técnicas de reamostragem. Neste estudo, usou-se a estratégia *grid search*.
- **Indicar o modelo final**: Indicar o modelo de melhor desempenho preditivo na reamostragem, baseado em alguma métrica de avaliação. Neste estudo, uma configuração manual foi realizada para obter diversas métricas de desempenho. Por padrão, o CARET retorna apenas as métricas RMSE (do inglês, *Root Mean Square Error*) e MAE (do inglês, *Mean Absolute Error*) para cada dobra da validação cruzada. As métricas serão melhor detalhadas na subseção 3.3.4.

Na construção dos modelos de aprendizado de máquina para estimativa do volume comercial com casca foram consideradas duas estratégias de modelagem preditiva: **Abordagem 1**: usar todas as p preditoras (p = 11) (ver subseção 3.3.1) como entradas no processo de construção dos modelos; **Abordagem 2**: usar apenas preditoras que continham o diâmetro incluso (p = 4) como entradas no processo de construção dos modelos. A abordagem 2 foi idealizada para refletir a situação em que apenas a variável diâmetro da árvore estivesse disponível e, a abordagem 2 reflete a situação em que além do diâmetro, a altura das árvores também foi medida.

As variantes de hiperparâmetros, para cada algoritmo, usadas na construção de modelos de aprendizado de máquina para predição de biomassa aérea total (TABELA 7) e volume comercial com casca (TABELA 8) estão disponíveis, assim como os respectivos pacotes do ambiente estatístico R usados.

TABELA 7 – ALGORITMOS DE APRENDIZAGEM, VARIANTES DE HIPERPARÂMETROS DE AJUSTE E BIBLIOTECAS DO AMBIENTE ESTATÍSTICO R USADAS PARA MODELAGEM DA BIOMASSA AÉREA TOTAL EM FLORESTAS TROPICAIS.

Algoritmo	Variantes de hiperparâmetros	Método*	Pacote R	Autor
Weighted k-Nearest-Neighbor - *wk*NN	**kmax** = {seq(2,150,2)} **kernel** = {"rectangular","biweight", cos,"epanechnikov","gaussian", "inv","optimal","rank","triangular", triweight} **distance** = {1:3}	kknn	kknn	Schliep e Hechenbichler (2016)
Regression Trees - RT	**cp** = {seq(0, 0.05, 0.005)}	rpart	rpart	Therneau e Atkinson (2019)
Model Tree - M5'	**pruned** = (Yes, No) **smoothed** = (Yes, No)	M5	RWeka	Hornik, Buchta e Zeileis (2009)
Random Forest - RF	**mtry** = {1, 4, 6, 8} **ntree** = {50, 100, 150, 200, 500}	rf	randomForest	Liaw e Wiener (2002)
Bagged Trees - BT	**nbagg** = {25, 50, 100, 150, 200}	treebag	ipred	Peters e Hothorn (2019)
Stochastic Gradient Boosting - SGB	**interaction.depth** = {seq(1,5,1)} **shrinkage** = {0.01, 0.1, 0.3} **n.trees** = {10, 50, 100, 500, 1000, 1500} **n.minobsinnode** = {5, 10, 15, 30}	gbm	gbm	Greenwell *et al.* (2019)
Extreme Gradient Boosting - XGBoost	**nrounds** = {seq(200, 1000, 50)} **eta** = {0.025, 0.05, 0.1, 0.3} **max_depth** = {2, 4, 5, 6} **gamma** = 1 **colsample_bytree** = seq(0.5,1,0.1) **min_child_weight** = 1 **subsample** = 1	xgbTree	xgboost	Chen *et al.* (2019)
Support Vector Regression - SVR	**C** = {seq(0.1, 1, 0.025)}	svmLinear	kernlab	Karatzoglou *et al.* (2004)
	C = {2^c(5,6,7,8,9,10)} **sigma** = {c(0.0005,0.0001,0.005,0.001,0.1,0.5)}	svmRadial		
	C = {2^seq(-2,7,1)} **scale** = {c(0.001, 0.005, 0.01, 0.02)} **degree** = {1, 2}	svmPoly		
Artificial Neural Network - ANN	**size** = {seq(1,10,1)} **decay** = {seq(0.1, 0.7, 0.1)}	nnet	nnet	Venables e Ripley (2002)

FONTE: O autor (2020).
NOTA: *nome do método na função train() do pacote CARET.

TABELA 8 – ALGORITMOS DE APRENDIZAGEM, VARIANTES DE HIPERPARÂMETROS DE AJUSTE E BIBLIOTECAS DO AMBIENTE ESTATÍSTICO R USADAS PARA MODELAGEM DO VOLUME COMERCIAL COM CASCA DE ESPÉCIES MANEJADAS NA AMAZÔNIA BRASILEIRA.

Algoritmo	Variantes de hiperparâmetros	Método*	Pacote R	Autor
Weighted k-Nearest-Neighbor - *wk*NN	**kmax** = {seq(2,25,2)} **kernel** = {"rectangular","biweight", cos,"epanechnikov","gaussian", "inv","optimal","rank","triangular", triweight} **distance** = {1:3}	kknn	kknn	Schliep e Hechenbichler (2016)
Regression Trees - RT	**cp** = {seq(0.00001, 0.001, 0.00001)}	rpart	rpart	Therneau e Atkinson (2019)
Model Tree - M5'	**pruned** = (Yes, No) **smoothed** = (Yes, No)	M5	RWeka	Hornik, Buchta e Zeileis (2009)
Random Forest - RF	**mtry** = {1:11}** **ntree** = {seq(50,500,50)}	rf	randomForest	Liaw e Wiener (2002)
Bagged Trees - BT	**nbagg** = {25,50,100,150,200,500,1000, 1500}	treebag	ipred	Peters e Hothorn (2019)
Stochastic Gradient Boosting - SGB	**interaction.depth** = {1:5} **shrinkage** = {seq(0.001,0.1,0.01)} **n.trees** = {c(100,200,300,500,700)} **n.minobsinnode** = {c(5,10,20,30,50)}	gbm	gbm	Greenwell *et al.* (2019)
Extreme Gradient Boosting - XGBoost	**nrounds** = {seq(200,1000,50)} **eta** = {seq(0.01,0.07,0.01)} **max_depth** = {1:6} **gamma** = 0 **colsample_bytree** = {seq(0.7,1,0.1)} **min_child_weight** = {seq(5,50,5)} **subsample** = {seq(0.7,1,0.1)}	xgbTree	xgboost	Chen *et al.* (2019)
Support Vector Regression - SVR	**C** = {seq(0.1, 1, 0.025)}	svmLinear	kernlab	Karatzoglou *et al.* (2004)
	C = {2^c(4,5,6,7)} **sigma** = {seq(0,.01,0.001)}	svmRadial		
	C = {2^seq(-2,7,1)} **scale** = {c(0.001, 0.005, 0.01, 0.02)} **degree** = {1, 2}	svmPoly		
Artificial Neural Network - ANN	**size** = {seq(1,11,1)} **decay** = {seq(0.001, 0.01, 0.001)}	nnet	nnet	Venables e Ripley (2002)

FONTE: O autor (2020).

NOTA: *nome do método na função train() do pacote CARET.**Para a abordagem 2, usou-se mtry = 1:4.

3.3.4 Desempenho e seleção de modelos

A estimativa de desempenho das variantes dos algoritmos de aprendizado de máquina foi medida por meio de estatísticas de qualidade de ajuste (*Goodness-of-fit statistics*). Assim, para cada subconjunto disjunto *k* no esquema 5x10-folds CV, obteve-se diversas métricas de avaliação: a) Raiz do Erro Quadrático Médio (do inglês, *Root Mean Square Error* - RMSE) (Eq. 3.13); b) Raiz do Erro Quadrático Médio Relativo (do inglês, *Relative Root Mean Square Error* - rRMSE) (Eq. 3.14); c) Coeficiente de Determinação (do inglês, *Coefficient of Determination* - R^2) (Eq. 3.15); d) Viés (do inglês, *Bias*), em porcentagem (Eq. 3.16); e) Coeficiente de Correlação de Pearson - (do inglês, *Pearson Correlation Coefficient* - r); e f) Erro Absoluto Médio (do inglês, *Mean Absolute Error* - MAE) (Eq. 3.17) (KVÅLSETH, 1985; PRETZSCH, 2009; KUHN; JOHNSON, 2013; CHAI; DRAXLER, 2014; MAUYA *et al.*, 2015; TANAKA *et al.*, 2014; ZHANG *et al.*, 2015). O valor de r foi usado para quantificar a correlação entre os valores observados e preditos por cada modelo preditivo (ZHANG *et al.*, 2015). O r=1 indica perfeita correlacão entre valores preditos e observados. O R^2 fornecido pelo pacote CARET pode ser obtido fazendo-se o quadrado do r (KUHN; JOHNSON, 2013). Quanto mais próximo de 1 estiver o R^2, maior será

percentual de variância explicada pelo modelo preditivo (SONG *et al.*, 2017).

O desempenho final de cada modelo foi obtido fazendo-se a média aritmética das estimativas nos 50 subconjuntos disjuntos *k* do esquema 5x10-folds CV (Eq. 3.18). A variância de desempenho nas partições de validação também foi medida (3.19). Após determinar o ótimo ajuste de hiperparâmetros para cada algoritmo, por meio de validação cruzada, as melhores configurações indicadas foram usadas para ajustar os modelos a todo conjunto de treinamento. Os resíduos padronizados, que correspondem ao resíduo e_i dividido pela raiz quadrada do quadrado médio dos resíduos (ver Eq. 3.11), foram obtidos para os dados de treinamento. Também os modelos *tuning* (com configurações ótimas de hiperparâmetros), ajustados a todo conjunto de treino, foram avaliados sobre o conjunto de teste, dados não usados na construção dos modelos. Por fim, um modelo final foi obtido usando todo o conjunto de dados disponível.

$$RMSE_{(k)} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y_i} - y_i\right)^2} \tag{3.13}$$

$$rRMSE_{(k)} = \frac{100}{\bar{y}}\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y_i} - y_i\right)^2} \tag{3.14}$$

$$R^2_{(k)} = \left(\frac{\sum_{i=1}^{n}\left(\hat{y_i} - \bar{\hat{y}}\right)\left(y_i - \bar{y}\right)}{\sqrt{\left[\sum_{i=1}^{n}\left(\hat{y_i} - \bar{\hat{y}}\right)^2\right]\left[\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2\right]}}\right)^2 \tag{3.15}$$

$$Viés_{(k)}(\%) = \frac{100}{\bar{y}}\left(\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y_i} - y_i\right)\right) \tag{3.16}$$

$$MAE_{(k)} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y_i} - y_i| \tag{3.17}$$

$$\bar{E} = \frac{1}{r.k}\sum_{k=1}^{rk}\hat{E_k} \tag{3.18}$$

$$CV(\%) = \frac{\sqrt{Var(\hat{E_k})}}{\bar{E}}.100 \tag{3.19}$$

Em que: n = número de observações do subconjunto *k*; $\hat{y_i}$ = valor predito para a *i*-ésima árvore no subconjunto *k*; y_i = valor observado para a *i*-ésima árvore no subconjunto *k*; $\bar{y}$ = média observada da variável resposta na dobra *k* da reamostragem; $\bar{\hat{y}}$ = média das predições da variável resposta na dobra *k* da reamostragem. $\bar{E}$ = estimativa da média de desempenho nas partições *k* de validação cruzada; *r* = número de repetições do processo de validação cruzada; *k* = número de partições em cada repetição *r* da validação cruzada; $\hat{E_k}$ = estimativa de desempenho tomada em cada dobra da validação *k*, medida com Eq. 3.13, 3.14, 3.15, 3.1, 3.16 ou 3.17. QMR = Quadrado Médio do Resíduo.

3.3.5 Diferença de desempenho entre modelos

Em modelos de aprendizagem de máquina é comum usar técnicas de reamostragem, como *k*-fold cross validation, para estimar a capacidade de generalização de um modelo. Para essa situação, o desempenho final de cada modelo é obtido fazendo-se a média aritmética das estimativas nos subconjuntos disjuntos *k* da reamostragem (KUHN; JOHNSON, 2013). Em seguida, as estimativas de desempenho médio podem ser comparadas diretamente para escolha entre modelos. No entanto, em algumas situações não existe evidência clara, em termos de desempenho, para escolha de um ou outro modelo. Neste cenário, várias questões podem ser consideradas para a tomada de decisão sobre o melhor modelo. Por exemplo, uma recomendação é considerar escolher o modelo mais simples que se aproxime razoavelmente do desempenho dos métodos mais complexos (KUHN; JOHNSON, 2013). Essa decisão pode ser apoiada por métodos estatísticos que comparam o desempenho médios de algoritmos de aprendizado de máquina, baseado em estimativas de reamostragem.

O *framework* CARET possui o método de Bland-Altman facilmente disponível para comparação entre modelos baseado nas estimativas da validação cruzada. Para tanto, a função resamples() pode ser usada para criar uma lista de modelos e, em seguida, a função xyplot() utilizada para plotagem gráfica. No entanto, para a correta aplicação do método é imprescindível o uso da mesma semente de reprodutibilidade dada pela função set.seed() antes da chamada da função train() (KUHN, 2015a), pois assegura-se que na contrução de cada modelo de aprendizado de máquina sejam usados subgrupos *k* idênticos em cada iteração da reamostragem. Apesar de disponibilizado no CARET, foi preferível elaborar o gráfico de Bland-Altman com auxílio do pacote “blandr” (DATTA, 2017), que torna explicito o viés (diferença média), os limites de concordância inferior e superior, além dos Intervalos de Confiança (IC) para as estimativas, sobre probabilidade especificada.

O método de Bland-Altman foi proposto para avaliar o grau de concordância entre duas medidas quantitativas (ALTMAN; BLAND, 1983; BLAND; ALTMAN, 2010). A abordagem geral consiste em elaborar um gráfico de dispersão no qual o eixo x representa a média de um par de medições [(A + B)/2], e o eixo y mostra a diferença entre as duas medições pareadas (A - B) (ODOR; BAMPOE; CECCONI, 2017). Neste estudo, as estimativas nas partições pareadas da validação cruzada foram usadas para avaliar o grau de concordância entre os modelos de aprendizado de máquina. Assim, o eixo x foi representado pela média aritmética das estimativas, e no eixo y foi plotada a diferença das estimativas de desempenho. As medidas foram calculadas sempre considerando um par de algoritmos.

A inspeção visual do gráfico de Bland-Altman permite avaliar diferentes aspectos da comparação entre dois métodos. Uma medida consistente de viés (diferença média) é estimada pela média aritmética das diferenças na amostragem pareada entre dois métodos (ODOR; BAMPOE; CECCONI, 2017), consistindo na estimativa do quanto as diferenças entre dois métodos se afastam de zero, em média (HIRAKATA; CAMEY, 2009). Em geral, quanto mais próxima a diferença média (viés) estiver de zero, melhor será considerada a concordância entre as medidas (HIRAKATA; CAMEY, 2009; ODOR; BAMPOE; CECCONI, 2017).

O método também propõe limites inferior e superior de concordância, os quais podem ser estimados por $\bar{d} \pm 1,96 * DP$. Em que: $\bar{d}$ = diferença média ou viés; DP = desvio padrão das diferenças entre métodos. No entanto, para que os limites sejam válidos é necessário garantir que as diferenças entre métodos sejam normalmente distribuídas. Para constatar a normalidade da distribuição das diferenças é útil inspecionar o histograma de frequências e também realizar testes de hipóteses, como *Shapiro-Wilk* test e *Kolmogorov-Smirnov*. Admitida a suposição de distribuição normal, espera-se que 95% das diferenças estejam contidas no intervalo $\bar{d} \pm 1,96 * DP$ (GIAVARINA, 2015).

As medidas de viés e limites de concordância são estimativas pontuais e, portanto,

a definição de Intervalos de Confiança (ICs) são importantes para estabelecer regiões de confiança em torno desses valores. Para construir ICs é necessário também admitir que a distribuição amostral das diferenças seja gaussiana (GIAVARINA, 2015). Assim, para o viés $\bar{d}$ o erro padrão é dado pela equação 3.20. Em que s_d = desvio padrão das diferenças e *n* = tamanho da amostra. Então, os ICs para a estimativa de viés podem ser calculados conforme Eq. 3.21, com *t* sendo o valor tabelado da distribuição *t* para *n*-1 graus de liberdade. Para os limites de concordância o erro padrão é dado por $1,71 * EP_{\bar{d}}$. Portanto, o IC para o limite inferior de concordância é dado por $LIC = [(\bar{d} - 1,96 * DP) \pm t * 1,71 * EP_{\bar{d}}]$ e, o IC para o limite superior é obtido por $LSC = [(\bar{d} + 1,96 * DP) \pm t * 1,71 * EP_{\bar{d}}]$ (HIRAKATA; CAMEY, 2009).

$$EP_{\bar{d}} = s_d/\sqrt{n} \tag{3.20}$$

$$IC[\bar{d} - (t_{n-1} * EP_{\bar{d}}) \leq \bar{D} \leq \bar{d} + (t_{n-1} * EP_{\bar{d}})] = P \tag{3.21}$$

Todas as análises foram realizadas usando o ambiente estatístico R, versão 3.6.0. O código reproduzível para o ajuste do modelo pantropical de Chave *et al.* (2014) (n = 4004) e da mesma forma funcional usando o conjunto de treino (n = 3226) (Apéndice B), e também de modelos de aprendizado de máquina (Apéndice C) para predição de biomassa aérea total em florestas tropicais estão disponíveis. Do mesmo modo, estão acessíveis os códigos para ajustes de modelos de regressão tradicional (Apéndice D) e de MAM (Apéndice E) para predição do volume comercial com casca de espécies manejadas na Amazônia brasileira.

3.4 APLICAÇÃO WEB COM MODELOS DE APRENDIZADO DE MÁQUINA

No Brasil, em particular, o aumento no interesse em técnica de inteligência artificial para a predição de variáveis florestais ocorreu provavelmente nas últimas duas décadas. Nas ciências florestais, muitas pesquisas científicas têm reportado sucesso no uso de algoritmos de aprendizado de máquina para predição de variáveis biométricas, a exemplo de Fehrmann *et al.* (2008), Sanquetta *et al.* (2015), Montaño *et al.* (2017) e Sanquetta *et al.* (2018), e também no reconhecimento de espécies a partir de imagens, como as pesquisas de Martins *et al.* (2013), Paula Filho *et al.* (2014) e Martins *et al.* (2015).

Os modelos de aprendizado de máquina (MAM), em sua maioria, possuem estrutura bastante complexa e, portanto, o uso prático e intuitivo desses modelos treinados pode ser dificultoso, em especial para não programadores ou pessoas com pouco conhecimento em tecnologias. Na mensuração florestal, por exemplo, as pesquisas sempre concentraram maiores esforços no desenvolvimento de modelos de regressão tradicional para predizer as variáveis florestais. Em regressão tradicional, é fácil usar um determinado modelo ajustado para realizar novas predições, pois tem-se um modelo físico palpável. Por exemplo, considerando o modelo ajustado de *Berkhout* dado por $v = -5,2020 + 0,0223d + \epsilon_i$ (em que: v = volume, em m³; d = diâmetro da árvore, em cm) poder-se-ia, dado um novo vetor de diâmetros, calcular facilmente os volumes de novas árvores, usando simplesmente uma planilha do microsoft excel, abordagem mais usual. Porém, para modelos de IA essa abordagem extremamente intuitiva para obtenção das predições é, em geral, impraticável.

Diante desse contexto, uma indagação surgiu: uma vez encontrado um MAM acurado, como disponibilizá-lo ao usuário final para uso prático e fácil? Neste estudo, a ideia foi simples: desenvolver uma aplicação web com o(s) modelo(s) escolhido(s), usando o ambiente estatístico R. Para tanto, utilizou-se do pacote 'Shiny' que constitui um *framework* para construção de aplicações Web em R (CHANG *et al.*, 2019). Assim, o desenvolvimento de aplicações web constituiu uma forma amigável de disponibilizar modelos de aprendizado de máquina. Em particular, duas aplicações web foram construídos disponibilizando os melhores MAM para predição

da biomassa aérea total em florestas tropicais e volume comercial de árvores de espécies manejadas na Amazônia brasileira. Informações mais detalhadas podem ser consultadas na subseção 4.1.7 e subseção 4.2.6.

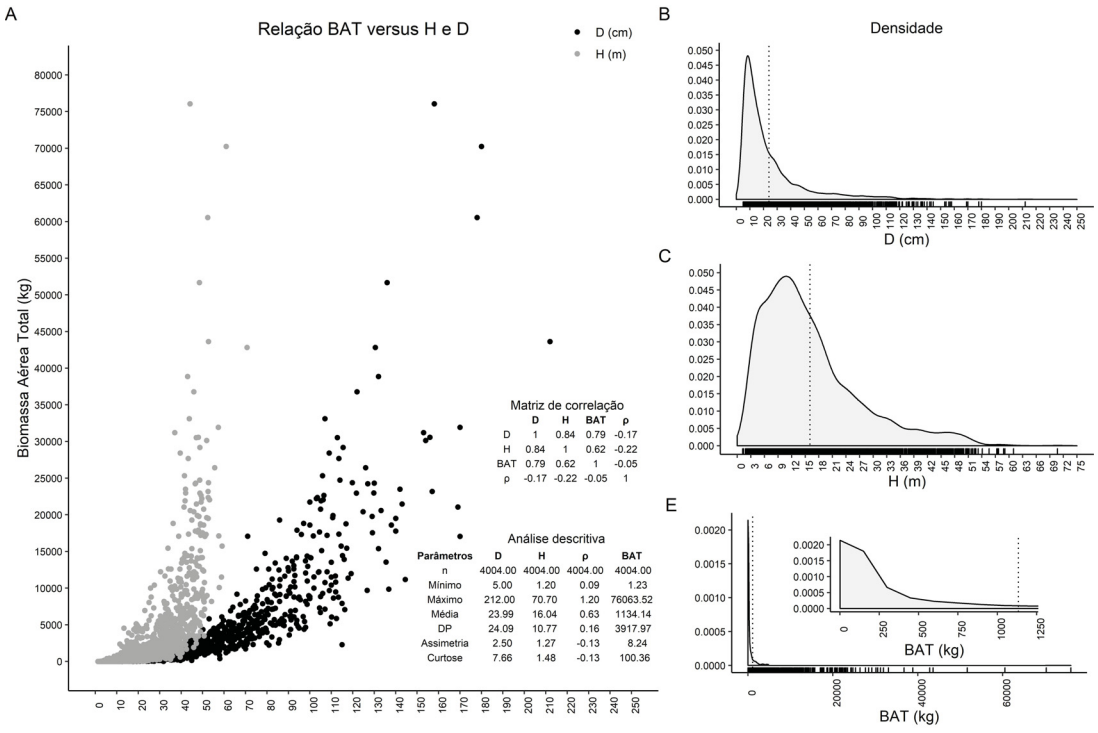
4 RESULTADOS E DISCUSSÃO

4.1 **ESTUDO DE CASO 1:** MODELOS DE APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DA BIOMASSA DA PARTE AÉREA EM FLORESTAS TROPICAIS

4.1.1 Análise exploratória dos dados originais e engenharia de variáveis

O conjunto de dados original (n=4004) usado na modelagem preditiva dos algoritmos de aprendizado de máquina apresentou variáveis biométricas com elevada dispersão em torno da média, fato esperado por se tratar de medidas de árvores de florestas multiâneas e, além disso serem oriundas de diversos locais com variadas condições climáticas e tipos de vegetação. A análise exploratória dos dados fornece informações valiosas sobre a distribuição, variância e o relacionamento entre as covariáveis e a biomassa de árvores (FIGURA 21).

FIGURA 21 – RELAÇÃO ENTRE BIOMASSA AÉREA TOTAL E ALTURA E DIÂMETRO DAS ÁRVORES, MATRIZ DE CORRELAÇÃO DE PEARSON E ANÁLISE EXPLORATÓRIA DOS DADOS ORIGINAIS (A); GRÁFICOS DE DENSIDADE (B, C, D); LINHA PONTILHADA NA VERTICAL REPRESENTA A MÉDIA ARITMÉTICA PARA CADA VARIÁVEL.



FONTE: O autor (2020).

LEGENDA: BAT = Biomassa Aérea Total (kg); D = Diâmetro a 1,30m do solo (cm); H = Altura total (m); ρ = Densidade básica da madeira (g.cm^{-3}); n = número de árvores; DP = desvio padrão.

A distribuição da variável resposta original (BAT, em kg) apresentou elevada assimetria, com grande dispersão em torno da média (CV = 345,46%), e árvores com peso superior a 30Mg (n=14) tiveram pouca representatividade. O grau de dispersão para as variáveis D, H e ρ foi de 100,42%, 67,14% e 25,40%, respectivamente. A maior densidade de árvores esteve nas classes inferiores de altura (H) e diâmetro (D), evidenciando distribuições com assimetria positiva (g_1>0) e leptocúrtica (g_2>0). Por outro lado, a variável densidade básica da madeira (ρ) mostrou curva mais próxima à Gaussiana, com moderada assimetria negativa e forma platicúrtica.

A análise dos diagramas de dispersão das covariáveis originais (diâmetro e altura) versus BAT revelou uma relação de variação não constante (heterocedasticidade). Assim, menores valores de D e H foram condizentes com menor dispersão da BAT, enquanto os maiores implicaram em maior variação da biomassa. É provável que a variância da preditora ρ esteja sendo subestimada. Este fato pode ser explicado, em parte, pela imputação de dados faltantes (média de densidade da espécie, gênero ou família), já que apenas 59% das árvores tiveram a densidade básica diretamente medida (ver Chave *et al.* (2014)).

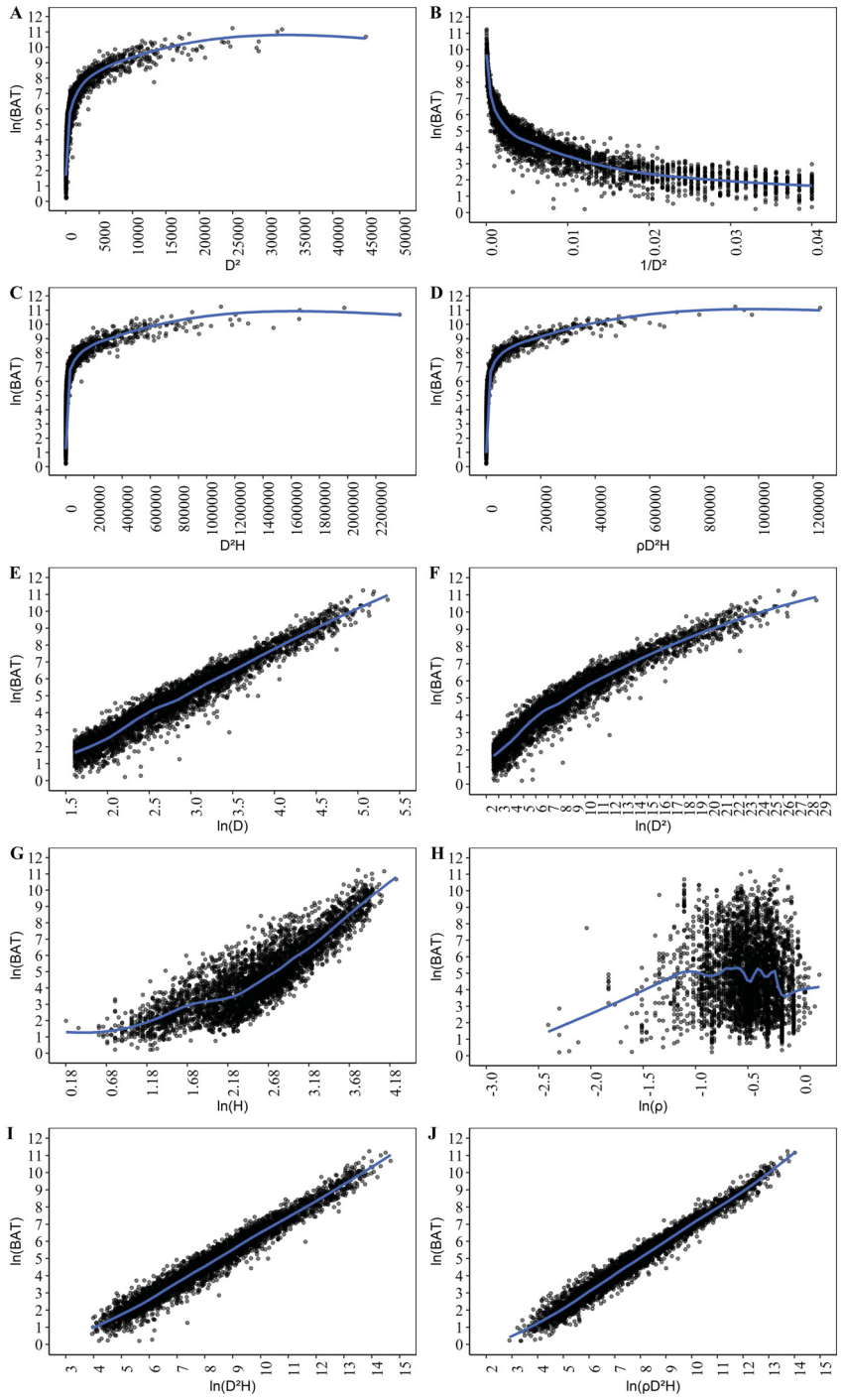
O coeficiente de correlação de Pearson (r) revelou correlações positivas e significativas entre todas as variáveis preditoras originais e a biomassa aérea total, através do teste *t-Student*, ao nível de 1% de probabilidade (α=0,01). A biomassa de árvores esteve mais fortemente associada ao diâmetro e altura, respectivamente. A densidade básica da madeira, em sua forma natural, não mostrou forte correlação linear com BAT.

A relação de variação não constante entre biomassa e as covariáveis torna o processo de modelagem um grande desafio e, portanto, etapas adicionais de transformações de dados e criação de medidas derivadas são estratégias bastante usuais e, em geral, objetivam melhorar as chances de sucesso do processo de construção de modelos preditivos. Neste estudo, portanto, o espaço de covariáveis foi aumentado com construção de preditoras de uso comum em modelos alométricos tradicionais (ver subseção 3.3.1).

A ideia de criar novas covariáveis esteve apoiada na hipótese de que transformações e/ou combinações dos preditores originais poderiam oferecer melhores representações do espaço de covariáveis e, por conseguinte, a obtenção de modelos de inteligência artificial com melhor qualidade preditiva. As 10 covariáveis construídas a partir das preditoras originais, e o relacionamento com logaritmo neperiano da biomassa [ln(BAT)] é apresentado (FIGURA 22). Neste estudo, usar ln(BAT) como resposta-alvo no processo de construção dos modelos de aprendizado de máquina foi a melhor estratégia encontrada para garantir uma distribuição residual mais comportada, além de permitir uma comparação direta do desempenho com o modelo proposto por Chave *et al.* (2014), que também usaram ln(BAT) como variável dependente na modelagem por regressão tradicional. A abordagem de aplicar transformação logarítmica a variável resposta e ao(s) preditor(es) é particularmente comum em modelos de predição de biomassa e, em geral, objetivam garantir a simetria e reduzir a heterocedasticidade da distribuição residual (SILESHI, 2014).

Ao aplicar a transformação ln(BAT) construiu-se um relacionamento não linear com as preditoras não-logaritmizadas (Fig. 23A, 23B, 23C e 23D). Quando as preditoras também foram transformadas para escala logarítmica, a relação com log-neperiano da biomassa mostrou, em alguns casos, comportamento linear, a exemplo da dispersão de ln(BAT) como função de $\ln(D^2H)$ ou $\ln(\rho D^2H)$.

FIGURA 22 – RELAÇÃO ENTRE ln(BAT) E COVARIÁVEIS CONSTRUÍDAS A PARTIR DE COMBINAÇÕES DE PREDITORAS ORIGINAIS E TRANSFORMAÇÕES COM LOGARITMO NATURAL.



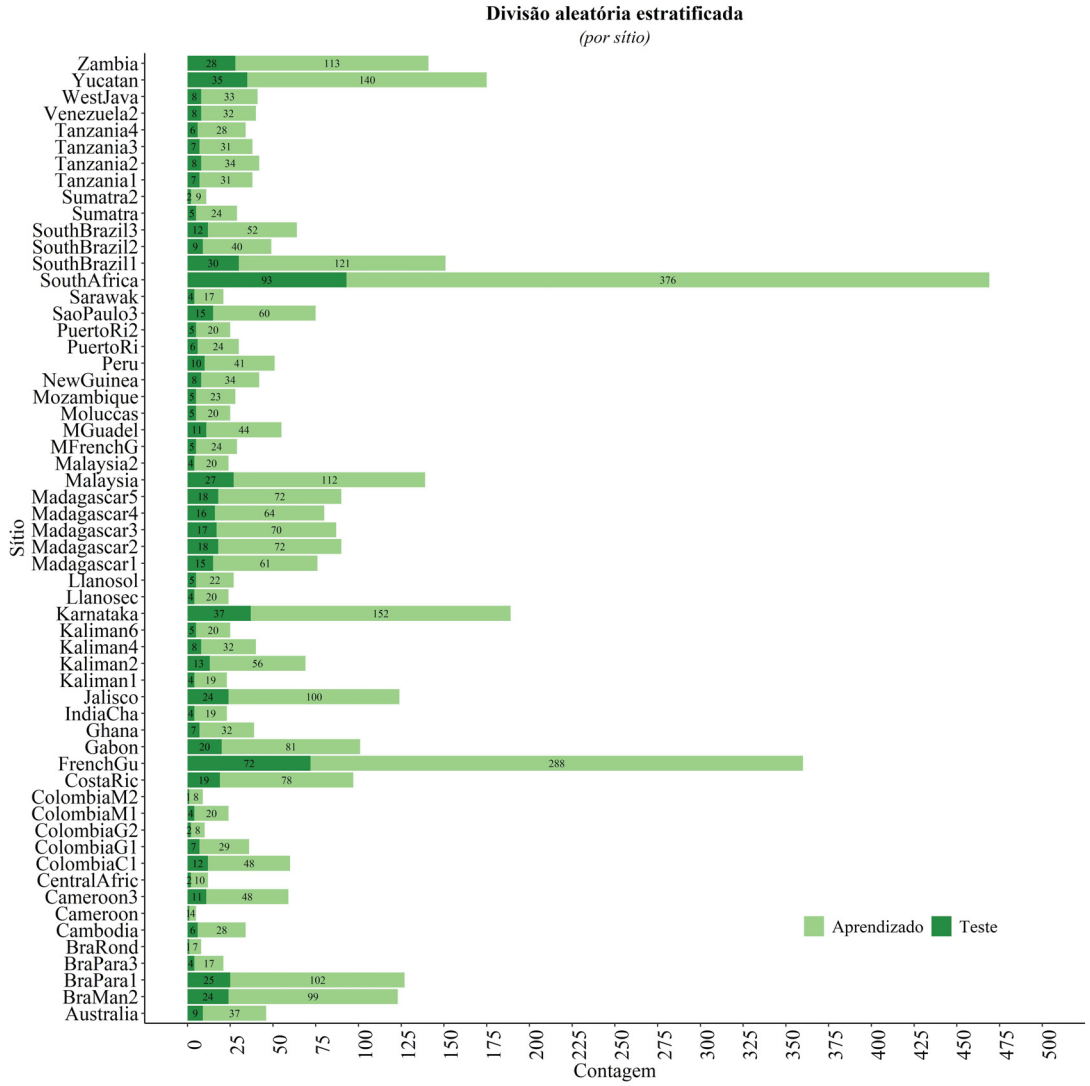
FONTE: O autor (2020).

LEGENDA: BAT = Biomassa Aérea Total (kg); D = Diâmetro a 1,30m do solo (cm); H = Altura total (m); ρ = Densidade básica da madeira (g.cm^{-3}); ln = logaritmo neperiano. Linha azul representa a curva de suavização LOESS (span=0,3).

4.1.2 Divisão de dados: mantendo as propriedades estatísticas

Uma adequada divisão de dados em aprendizado de máquina é fator importante para o sucesso da modelagem preditiva, pois pode influenciar na estimativa da capacidade de desempenho do modelo. Assim, antes do processo de treinamento dos modelos preditivos realizou-se a divisão do conjunto de dados completos em dois grupos: conjunto de treinamento e conjunto de teste. Para realizar a divisão usou-se a função createDataPartition() do CARET que implementa uma amostragem estratificada (*stratified random split*) (FIGURA 23).

FIGURA 23 – CONJUNTO DE DADOS DE TREINAMENTO E TESTE OBTIDOS POR AMOSTRAGEM ESTRATIFICADA.



FONTE: O autor (2020).

Em termos práticos, quando o vetor de entrada é numérico a função divide o conjunto de dados em subgrupos com base em percentis e, a amostragem é feita dentro desses subgrupos (KUHN *et al.*, 2016). Para a base de dados de biomassa (n = 4004) a estratificação foi realizada em função dos sítios (estratos), assegurando a representatividade de árvores de todos sítios nos conjuntos de treinamento (80%) e teste (20%). Além disso, o emprego do método manteve a proporção (80%-20%) dentro de cada estrato. Adicionalmente, garantiu-se que as propriedades estatísticas, em especial, a média, o desvio padrão e a forma das

distribuições (assimetria e curtose), de cada variável, dos conjuntos de treinamento e teste fossem semelhantes entre si e equiparáveis ao conjunto de dados completos. Ademais, os valores extremos de cada variável estiveram contidos no conjunto de treinamento. De tal modo, pôde-se garantir a máxima cobertura do espaço de características e dos padrões contidos nos dados completos (TABELA 9).

TABELA 9 – COMPARAÇÃO DAS PROPRIEDADES ESTATÍSTICAS DO CONJUNTO ORIGINAL, TREINAMENTO E TESTE.

Variáveis	**Dados completos**						
	n	Mínimo	Máximo	Média	DP	Assimetria	Curtose
D	4.004	5	212	23,99	24,09	2,50	7,66
H		1,2	70,7	16,04	10,77	1,27	1,48
ρ		0,09	1,2	0,63	0,16	-0,13	-0,13
BAT		1,23	76.063,52	1.134,14	3.917,97	8,24	100,36
	Dados de treinamento						
D	3.226	5	212	24,21	24,54	2,53	7,94
H		1,2	70,7	16,06	10,83	1,28	1,54
ρ		0,09	1,2	0,63	0,16	-0,12	-0,15
BAT		1,24	76.063,52	1.170,32	4.106,47	8,34	100,22
	Dados de teste						
D	778	5	132	23,07	22,09	2,23	5,25
H		2,1	52,7	15,94	10,54	1,21	1,18
ρ		0,1	1,08	0,64	0,17	-0,16	-0,04
BAT		1,23	30.530,99	984,12	3.010,70	5,67	38,14

FONTE: O autor (2020).

LEGENDA: BAT = Biomassa Aérea Total (kg); D = Diâmetro a 1,30m do solo (cm); H = Altura total (m); ρ = Densidade básica da madeira (g.cm^{-3}); DP = desvio padrão; n = número de observações.

4.1.3 Modelos de aprendizado de máquina: configuração, seleção e comparação

A escolha da ótima configuração de hiperparâmetros em modelos de aprendizado de máquina é um grande desafio, pois essa questão está diretamente relacionada à necessidade de evitar *overfitting* e de encontrar o modelo com melhor equilíbrio entre viés e variância, de modo a minimizar o erro de predição. Neste estudo, a primeira estratégia de modelagem empregada foi usar todas as 13 preditoras (originais e derivadas) como entradas no processo de construção dos MAM para predizer a resposta-alvo [ln(BAT)]. Para tanto, uma grade de valores candidatos foi usada para encontrar o(s) hiperparâmetro(s) de ajuste ótimo para cada algoritmo, usado o esquema 5x10-*folds* CV.

Em termos gerais, todos os algoritmos avaliados no esquema 5x10-*folds* CV mostraram estimativas de desempenho adequadas, com valores de RMSE variando entre 0,3482 (7,30%; ANN/MLP) e 0,3788 (7,98%; RT) (TABELA 10). Conforme esperado, os modelos de conjuntos (*ensemble*) (SGB, XGBoost, RF e BT) mostraram melhor desempenho do que o modelo baseado em árvore individual (RT), embora a melhoria não tenha sido proeminente. O modelo M5' de ótimo ajuste, também baseado em árvore individual, merece destaque dado ao seu desempenho competitivo frente à modernos algoritmos, como o XGBoost. No processo de ajuste de hiperparâmetros, alguns algoritmos têm melhoria no desempenho com uso de um adequado pré-processamento, além da diminuição do custo computacional.

TABELA 10 – CONFIGURAÇÃO ÓTIMA DE HIPERPARÂMETROS PARA CADA ALGORITMO E ESTIMATIVA DE DESEMPENHO MÉDIO NO ESQUEMA 5x10-FOLDS CROSS-VALIDATION.

Modelo	**Pré-Processamento**	**Hiperparâmetro**	**Conjunto de validação (folds = 50)**					
			$RMSE$	$rRMSE$	r	R^2	MAE	$Bias(\%)$
ANN	Center and Scale; BoxCox	**size** = 9 **decay** = 0,2	0,3482 (0,0151)	7,30 (0,33)	0,9866 (0,0015)	0,9733 (0,0029)	0,2652 (0,0121)	0,0019 (0,4699)
SGB	Center and Scale; BoxCox	**interaction.depth** = 5 **shrinkage** = 0,01 **n.trees** = 1500 **n.minobsinnode** = 5	0,3492 (0,0152)	7,32 (0,34)	0,9865 (0,0015)	0,9732 (0,0029)	0,2659 (0,0114)	0,0104 (0,4752)
SVR (Radial)	Center and Scale; BoxCox	**sigma** = 0,005 **C** = 512	0,3496 (0,0176)	7,33 (0,36)	0,9865 (0,0014)	0,9732 (0,0028)	0,2656 (0,0124)	-0,0078 (0,4343)
XGBoost	Center and Scale; BoxCox	**rounds** = 500 **eta** = 0,05 **max_depth** = 2	0,3514 (0,0185)	7,36 (0,38)	0,9864 (0,0016)	0,9729 (0,0032)	0,2685 (0,0134)	0,0037 (0,4558)
M5'	-	**pruned** = Yes **smoothed** = No	0,3557 (0,0183)	7,46 (0,38)	0,9860 (0,0015)	0,9723 (0,0029)	0,2716 (0,0128)	0,0051 (0,4832)
*wk*NN	Center and Scale; BoxCox	**k** = 66 **kernel** = triweight **d** = 1	0,3557 (0,0172)	7,46 (0,37)	0,9860 (0,0016)	0,9723 (0,0031)	0,2713 (0,0131)	0,1935 (0,4744)
RF	-	**mtry** = 1 **ntree** = 500	0,3615 (0,0167)	7,58 (0,36)	0,9855 (0,0016)	0,9713 (0,0032)	0,2743 (0,0130)	0,0254 (0,4881)
BT*	-	**nbagg** = 200	0,3698 (0,0168)	7,75 (0,36)	0,9847 (0,0016)	0,9590 (0,0032)	0,2832 (0,0131)	-0,0118 (0,5007)
RT	-	**cp** = 0,00016	0,3788 (0,0182)	7,94 (0,38)	0,9841 (0,0016)	0,9686 (0,0032)	0,2915 (0,0137)	-0,0077 (0,5234)

FONTE: O autor (2020).

NOTA: Todas as métricas foram calculadas na escala da variável resposta do modelo estatístico. *As árvores foram crescidas usando a configuração: rpart.control(minsplit = 2, cp = 0).

LEGENDA: ANN = Artificial Neural Networks (rede MLP); SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); $RMSE$ = Root Mean Square Error; $rRMSE$ = Relative Root Mean Square Error; r = Coeficiente de correlação de Pearson; R^2 = Coeficiente de determinação; MAE = Mean Absolute Error; Entre parênteses está o desvio padrão das métricas na reamostragem.

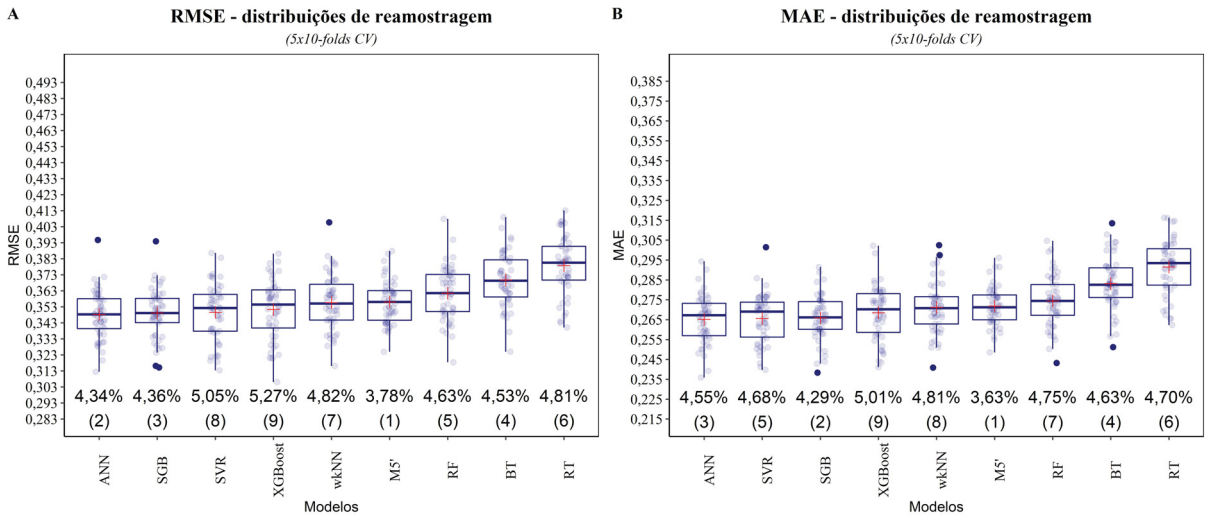
O melhor modelo ANN (rede MLP) evidenciou que nove neurônios na camada oculta e uma taxa de decaimento de pesos igual a 0,2, promoveu o melhor aprendizado dos pesos da rede neural e, portanto, um melhor modelo preditivo. O melhor modelo SGB foi obtido usando 1500 árvores, cada uma com 5 divisões (ou seis nós terminais), com um número mínimo de cinco observações por nó terminal, e uma taxa de aprendizado de 0,01. O melhor modelo XGBoost foi obtido usando 500 árvores como aprendizes de base, profundidade máxima=2 e taxa de aprendizado=0,05 e, para os demais parâmetros o padrão foi mantido.

O coeficiente de variação na validação cruzada repetida, para as métricas RMSE e MAE, foi inferior a 5% para a maioria dos modelos, indicando a estabilidade dos modelos para predizer a resposta média para amostras futuras. Para alguns modelos, medidas de desempenho discrepantes (Q_1 – 1.5*IQR e Q_3 + 1,5*IQR) foram identificadas (FIGURA 24). IQR é o intervalo interquartil (do inglês, *Interquartile Range* - IQR), dado pela diferença entre o terceiro (Q_3) e primeiro quartil (Q_1). O modelo XGBoost mostrou a maior variância na

reamostragem, para RMSE e MAE.

O emprego de técnicas de pré-processamento de preditoras faz sentido apenas quando usado um espaço m-dimensional de covariáveis. Este procedimento é estratégico, pois diminui os efeitos de preditores com maiores escalas de medidas sobre a determinação das métricas de distância (WITTEN *et al.*, 2017), além de facilitar e diminuir o tempo durante o processo de aprendizagem, como constatado na prática para as redes neurais e *wk*NN. Em termos gerais, o emprego conjunto dos métodos *Center and Scale* e *BoxCox* mostraram-se mais adequados para o pré-processamento das preditoras disponíveis. Outras técnicas como *spatialSign* e *YeoJohnson* não foram adequadas. Kuhn e Johnson (2013) afirmam que a forma com que os preditores entram no modelo também é importante, pois os diferentes modelos possuem sensibilidades distintas para os tipos de preditores. Neste estudo, a assertiva de Kuhn e Johnson (2013) foi verdadeiramente constatada. Ademais, para os algoritmos CART, M5' e Floresta Aleatória nenhum pré-processamento sobre as covariáveis foi realizado.

FIGURA 24 – DISTRIBUIÇÃO DAS ESTIMATIVAS DE DESEMPENHO (RMSE E MAE), NO ESQUEMA 5x10-FOLDS CV, PARA OS MODELOS DE APRENDIZADO DE MÁQUINA.



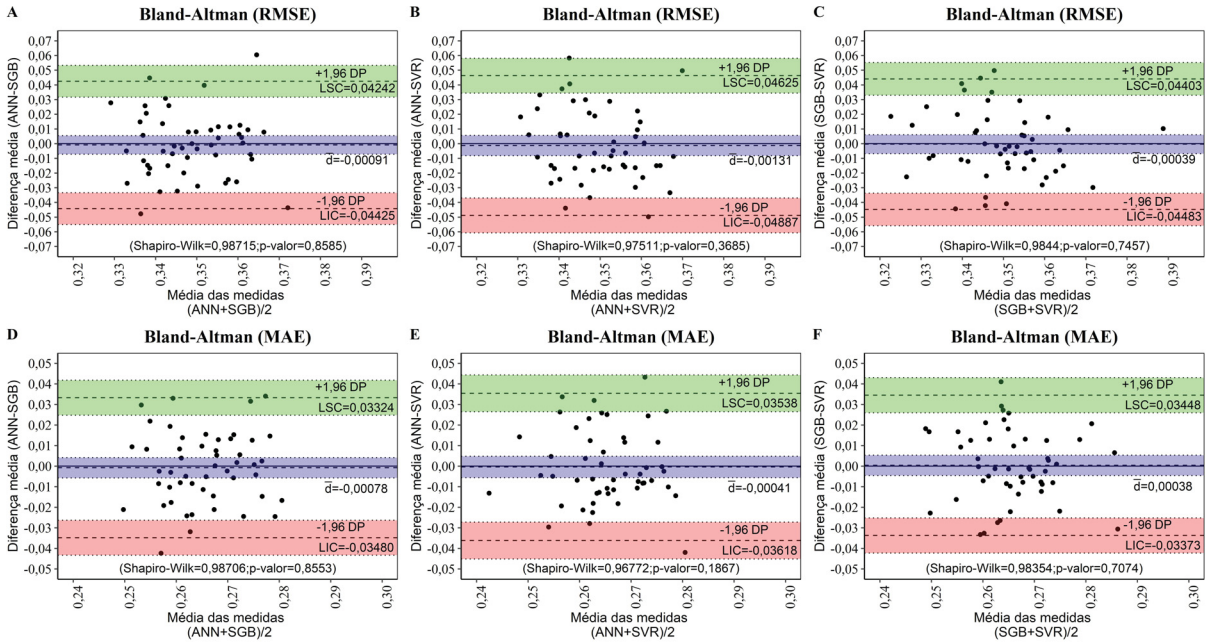
FONTE: O autor (2020).

LEGENDA: Barras na vertical (cor azul) representam: Q1 – 1.5*IQR (1º quartil menos 1,5 vezes o intervalo interquartil) e Q3 + 1,5*IQR (3º quartil mais 1,5 vezes o intervalo interquartil); Cruz em vermelho representa a média de desempenho dos modelos na reamostragem. O coeficiente de variação (CV) na reamostragem está expresso abaixo das caixas boxplot. ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); RMSE = Root Mean Square Error.

O gráfico de Bland-Altman constitui uma abordagem para avaliar o grau de concordância global entre modelos de aprendizado de máquina. Aqui, para fins de comparação foram considerados apenas os três melhores modelos indicados na reamostragem: ANN, SGB e SVR (FIGURA 25). Para todas as comparações, as estimativas de viés não foram consideradas significativas, e a linha de igualdade (zero) esteve dentro dos intervalos de confiança da diferença

média. Além disso, a maioria das diferenças entre modelos, para RMSE e MAE, estiveram dentro dos limites de concordância ($\bar{d} \pm 1,96 * DP$) e, portanto, a suposição de normalidade das distribuições das diferenças foi admitida através do teste *Shapiro-Wilk* (α = 0,05). Assim, houve evidências para admitir que os modelos possuem desempenho médio concordante e, portanto, é razoável admitir uma precisão similar dos modelos para predição de amostras futuras.

FIGURA 25 – ANÁLISE DE CONCORDÂNCIA DE BLAND-ALTMAN ENTRE OS MODELOS RNA, GB E MVS BASEADO NOS VALORES DE DESEMPENHO MÉDIO (RMSE E MAE) NA REAMOSTRAGEM.



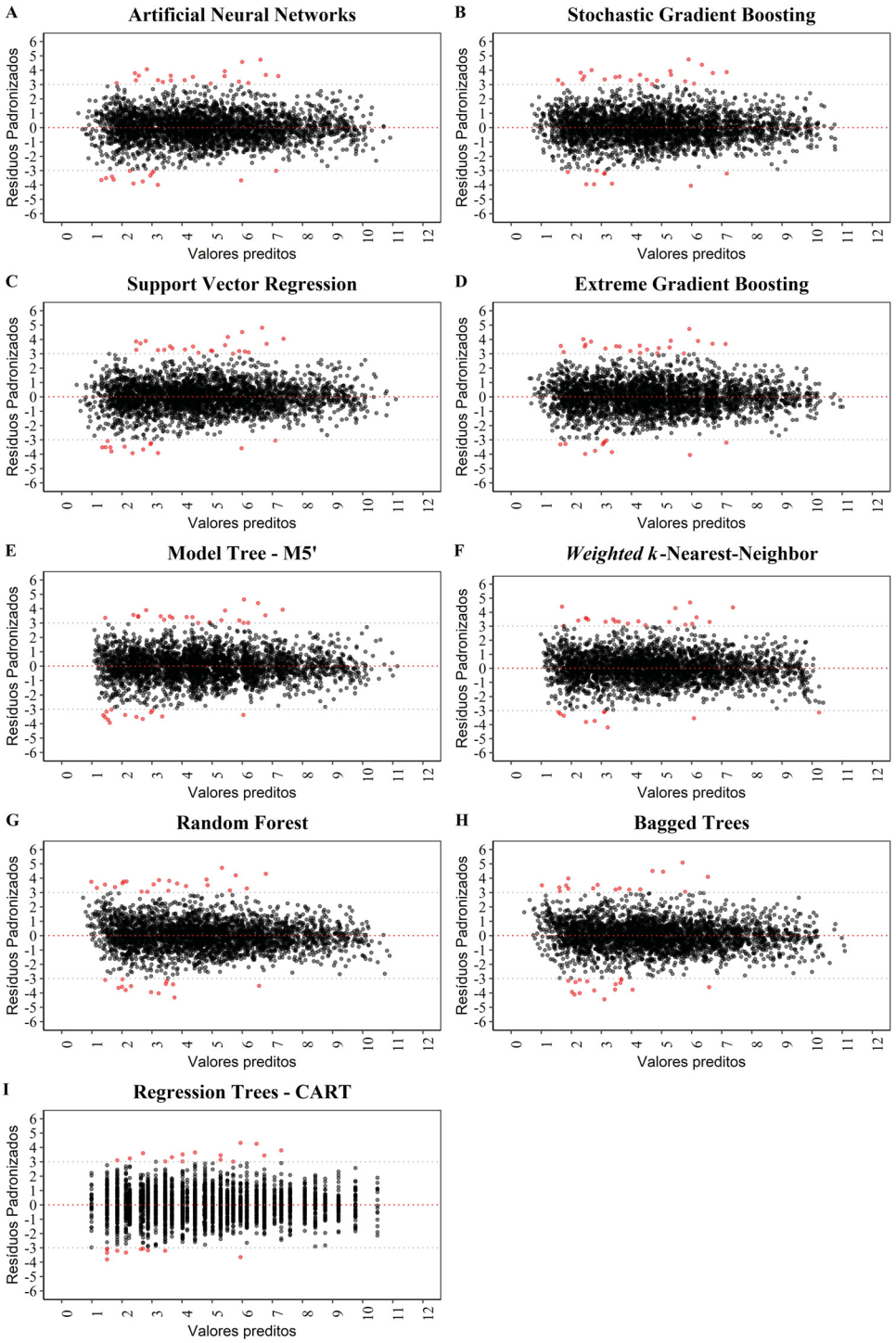
FONTE: O autor (2020).

NOTA: O gráfico de Bland-Altman foi elaborado com auxílio do pacote “blandr” (DATTA, 2017)

LEGENDA: $\bar{d}$ = diferença média entre dois modelos (linha pontilhada horizontal central); DP = desvio padrão; LIC = limite de concordância inferior; LSC = limite de concordância superior; regiões hachuradas = intervalo de confiança de 95% para os LIC (vermelha), LSC (verde) e diferença média (azul). ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; RMSE = Root Mean Square Error; MAE = Mean Absolute Error.

Os modelos foram ajustados a toda base de treinamento (*n* = 3226) usando os hiperparâmetros de ajuste ótimo indicados na reamostragem. Em seguida, o comportamento dos resíduos padronizados como função dos valores preditos foi avaliado. Todos os modelos apresentaram uma quantidade razoável de resíduos fora do intervalo $-3 \leq z_i \leq 3$, indicando a presença de pontos discrepantes (FIGURA 26).

FIGURA 26 – RESÍDUOS PADRONIZADOS NO CONJUNTO DE TREINAMENTO USANDO OS MODELOS COM CONFIGURAÇÃO ÓTIMA DE HIPERPARÂMETROS.



FONTE: O autor (2020).

NOTA: Pontos em vermelho indicam resíduos com valores fora do intervalo $-3 \leq z_i \leq 3$.

4.1.4 Interpretando modelos de aprendizado de máquina: importância e relação de variáveis

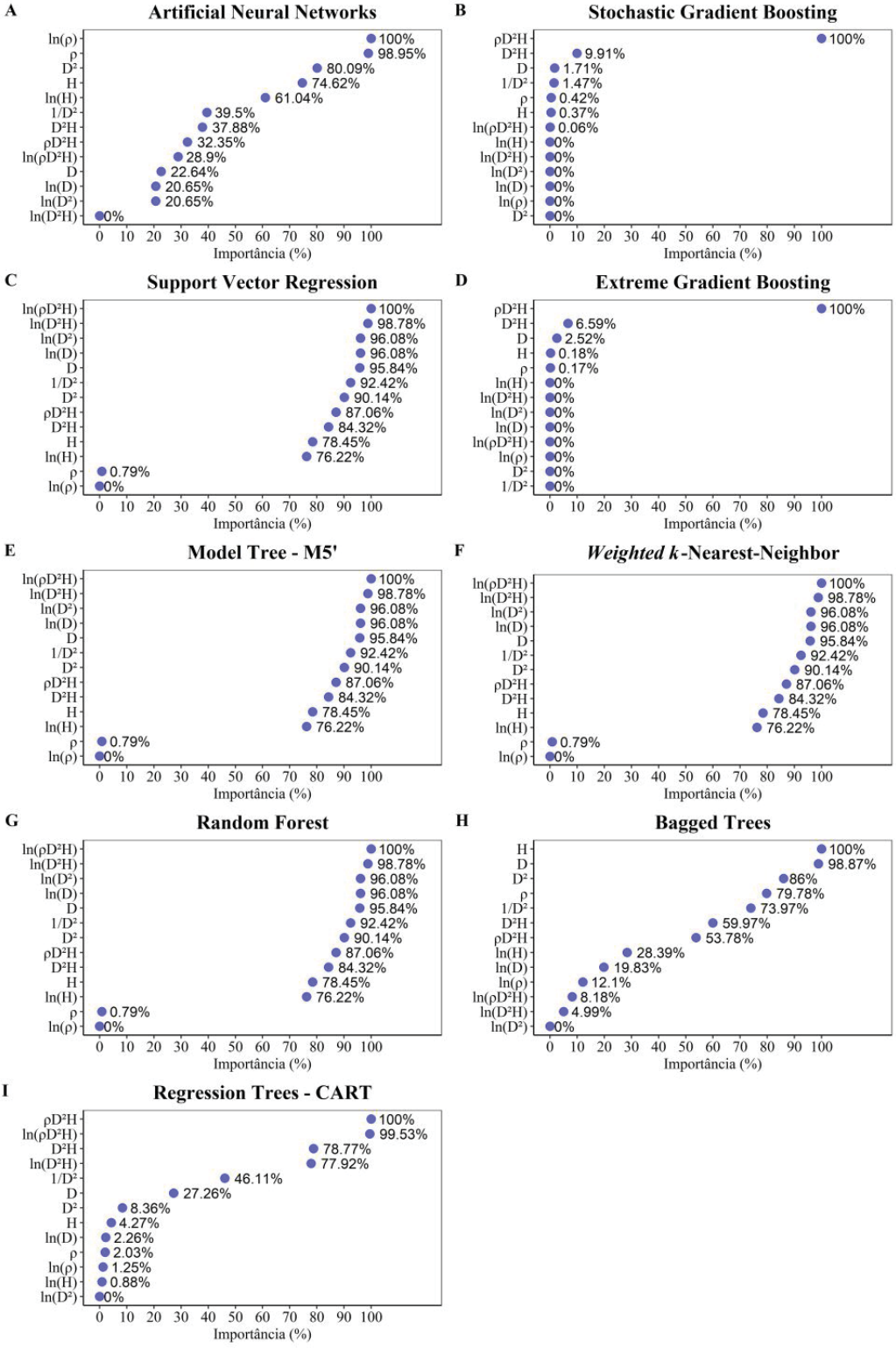
Nas últimas décadas, o desenvolvimento de poderosos algoritmos de aprendizado de máquina tem impulsionado o uso da abordagem na solução de problemas complexos. Esses algoritmos modernos são caracterizados pela alta flexibilidade e complexidade, isto é, têm a capacidade de modelar diversas formas funcionais, geralmente com alta precisão. Apesar disso, comumente a metáfora da "caixa preta" (do inglês, *Black-Box*) é usada para questionar a interpretabilidade de modelos de aprendizado de máquina. Para aproximar funções para a variável resposta os algoritmos recorrentemente utilizam de uma variedade de hiperparâmetros otimizáveis. Assim, para muitos cientistas e áreas de domínio questões como: Qual a importância relativa de preditoras para o modelo preditivo? Qual a relação (ou efeito) marginal entre resposta-preditor no modelo? são recorrentes quando a modelagem por aprendizado de máquina é usada.

Felizmente, recentemente algumas iniciativas têm demonstrado que é possível obter algum nível de interpretabilidade para modelos de aprendizado supervisionado, fato que incentivou a criação de bibliotecas especializadas, como DALEX (*Descriptive mAchine Learning Explanations*) (BIECEK, 2018), LIME (*Local Interpretable Model-Agnostic Explanations*) (PEDERSEN; BENESTY, 2019) e iml (*Interpretable Machine Learning*) (MOLNAR; BISCHL; CASALICCHIO, 2018), disponíveis no ambiente estatístico R. A biblioteca DALEX possui uma coleção consistente de "explicadores" (ou funções), que incorporam as abordagens mais conhecidas para explicação de modelos preditivos (BIECEK, 2018). Aqui, em particular, duas abordagens foram empregadas: 1) gráficos de importância de variáveis preditoras; e 2) gráficos de dependência parcial (do inglês, *Partial Dependence Plots* - PDP) (FRIEDMAN, 2001).

Estimar a importância das variáveis (IV) preditoras em aprendizado de máquina é útil para a identificar covariáveis que fornecem pouca informação para a modelagem da variável resposta [ln(BAT)] e que, por conseguinte, podem ser eliminadas do processo de treinamento, diminuindo o custo computacional. Isso é particularmente útil quando o espaço de covariáveis possui alta dimensão, além de fornecer indicativo da necessidade de engenharia de variáveis.

Em geral, as estratégias para avaliar a importância de uma variável preditora podem basear-se em métricas independentes ou específicas do modelo ajustado (KUHN; JOHNSON, 2013). Para os algoritmos ANN, SGB, XGBoost, RF e BT, métodos específicos baseados na estrutura dos modelos ajustados estão disponíveis. Porém, para melhor compreender os algoritmos de importância de variáveis e sua matemática subjacente, recomenda-se uma profunda inspeção dos códigos fontes e entendimento dos estudos vinculados aos algoritmos desenvolvidos. Por exemplo, Kuhn *et al.* (2016) afirma que o método usado pela função varImp() para ANN foi baseado em Gevrey, Dimopoulos e Lek (2003), que usa combinações dos valores absolutos dos pesos da rede.

FIGURA 27 – IMPORTÂNCIA RELATIVA DE PREDITORAS EM MODELOS DE APRENDIZADO DE MÁQUINA NO CONJUNTO DE TREINAMENTO.



FONTE: O autor (2020).

NOTA: Elaborado com auxílio do pacote CARET (KUHN *et al.*, 2016).

LEGENDA: D = Diâmetro a 1,30m do solo (cm); H = Altura total (m); ρ = Densidade básica da madeira (g.cm^{-3}); ln = logaritmo neperiano.

Por outro lado, para os algoritmos SVR, M5' e *wk*NN, em particular, a função varImp() do pacote CARET não estima a importância das preditoras baseado em métricas específicas do modelo ajustado. Portanto, o valor de R^2 do ajuste marginal entre cada preditor com a variável resposta [ln(BAT)], através da regressão LOESS (span = 0,75), é tomado como um indicativo da importância da covariável (ver subseção 3.3.1). A importância relativa das covariáveis para a estratégia de modelagem usando todas as 13 preditoras (originais e derivadas) é apresentada (FIGURA 27).

Para o caso de métricas independentes do modelo ajustado, sete preditoras [ln(ρD^2H), ln(D^2H), ln(D^2); ln(D); D; $1/D^2$ e D^2] mostraram importância superior a 90%, com valores de R^2 superior a 0,87 no ajuste marginal. A covariável combinada ln(ρD^2H) mostrou a melhor relação com a resposta, sendo um bom indicativo para o seu uso marginal ou combinado no processo de ajuste de hiperparâmetros dos algoritmos SVR, M5', RF e *wk*NN. Dentre as variáveis não combinadas, o diâmetro da árvore forneceu melhores informações para a modelagem da variável resposta. Por outro lado, o emprego de ρ e ln(ρ) como regressores marginais não se mostrou promissor ($R^2 < 0,02$).

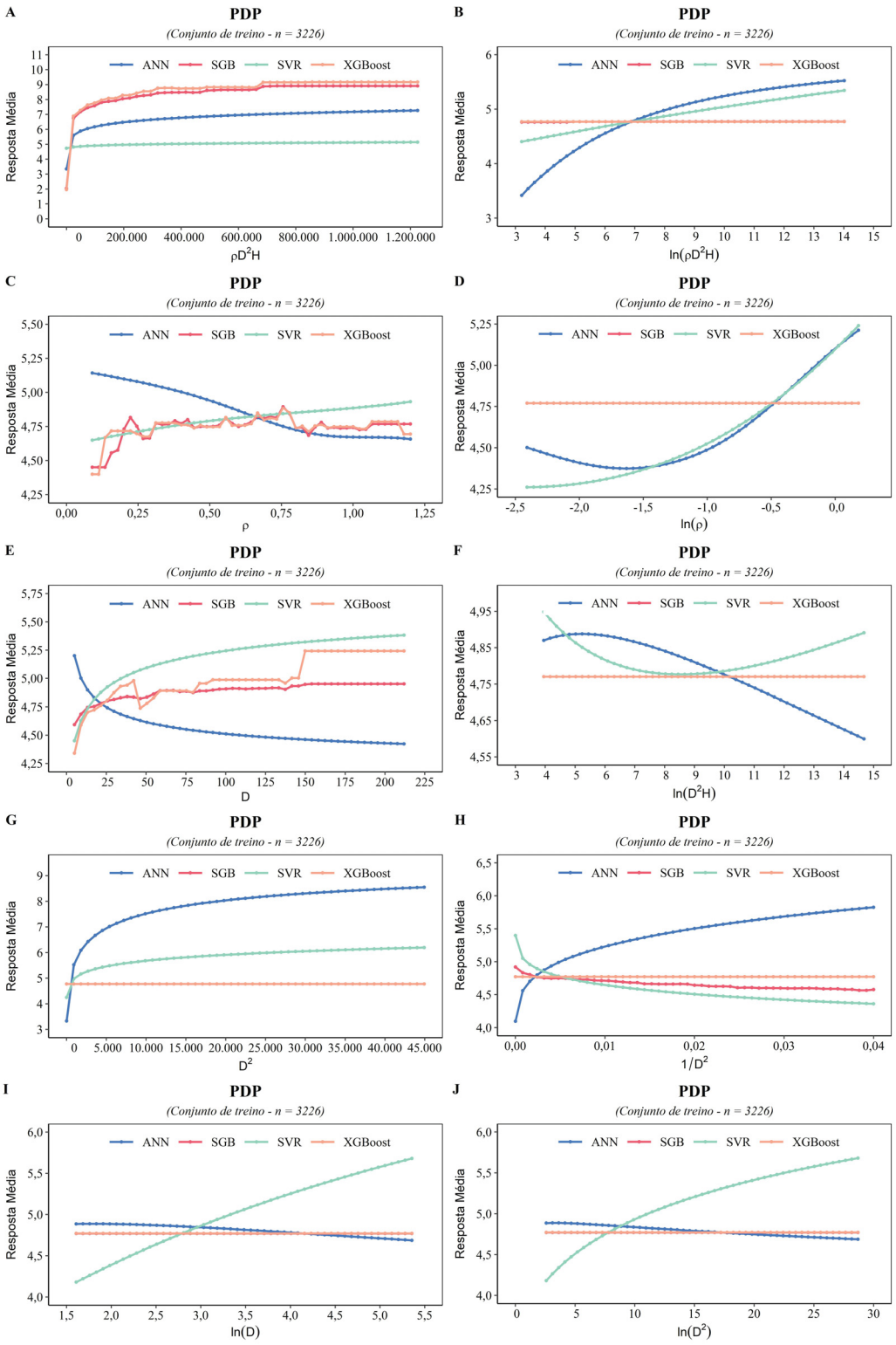
Nas situações em que a estrutura dos modelos ajustados foi usada para quantificar a importância das variáveis, poucas preditoras mostraram altas taxas de importância. A covariável ρD^2H forneceu melhores informações para a modelagem de ln(BAT) através dos algoritmos SGB e XGBoost, enquanto as demais preditoras ofereceram dados pouco úteis para modelagem. Para RT e BT as preditoras mais importantes foram ρD^2H, ln(ρD^2H), D^2H e ln(D^2H). Por outro lado, contrariando a tendência constatada para os demais algoritmos, as covariáveis ρ e ln(ρ) apresentaram maior importância para modelagem preditiva usando ANN.

Na busca por compreender como as predições dependem parcialmente dos valores de preditoras nos modelos aprendizado de máquina, algumas abordagens têm sido propostas, como os gráficos de dependência parcial (ou gráficos PDP). Aqui, foram construídos gráficos PDP bidimensionais para examinar o relacionamento entre uma única preditora e a variável resposta. Para tanto, foram consideradas apenas as preditoras mais influentes (limiar = 90%) para os quatro modelos melhor ranqueados na estimativa de reamostragem (FIGURA 28).

Em termos gerais, os modelos parecem ter aprendido melhor a tendência da relação empírica marginal entre a variável resposta e um preditor para as suas variáveis mais importantes. Por exemplo, a variável de interação ρD^2H, mais influente para os modelos SGB e XGBoost, descreveu um relacionamento crescente e não-linear com a resposta média, ou seja, maior ρD^2H parece induzir escores mais altos para as predições. Portanto, o perfil PDP (FIGURA 28A) parece corresponder à tendência observada na relação empírica marginal não-linear entre ρD^2H e ln(BAT).

Para o modelo ANN, existe uma tendência de diminuição no valor estimado da resposta média à medida que ρ aumenta, porém o impacto na predição parece ser modesto para valores superiores à 0,75. Para o modelo SVR, os gráficos PDP para ln(ρD^2H) e ln(D) refletem uma relação linear crescente com a resposta média, ou seja, há uma tendência de aumento progressivo das predições à medida que aumentam os valores das preditoras. Do mesmo modo, para SVR o perfil PDP parece capturar à tendência observada na relação linear empírica da resposta com ln(ρD^2H) e ln(D), e não-linear com ln(D^2), D, D^2 e $1/D^2$.

FIGURA 28 – DEPENDÊNCIA PARCIAL ENTRE A RESPOSTA MÉDIA E PREDITORAS MAIS INFLUENTES PARA OS QUATRO MELHORES MODELOS DE APRENDIZADO DE MÁQUINA INDICADOS NA VALIDAÇÃO CRUZADA.



FONTE: O autor (2020).

NOTA: Elaborado com auxílio do pacote DALEX (BIECEK, 2018).

LEGENDA: PDP = Partial Dependence Plots; D = Diâmetro a 1,30m do solo (cm); H = Altura total (m); ρ = Densidade básica da madeira (g.cm^{-3}); ln = logaritmo neperiano. ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting.

4.1.5 Modelo Clássico de Regressão Linear

Um dos objetivos deste capítulo foi também realizar uma comparação de MAM frente ao Modelo Clássico (ou Gaussiano) de Regressão Linear (MCRL). Conforme discutido na subseção 3.1.2, o conjunto de dados (n = 4004) deste estudo foi compilado por Chave *et al.* (2014), e usado para ajustar modelos lineares através de Mínimos Quadrados Ordinários (MQO). O melhor resultado encontrado supôs a forma funcional com preditor de variável combinada: $ln(BAT) = \beta_0 + \beta_1\ ln(\rho\ D^2H) + \epsilon_i$. Os parâmetros estimados para o modelo foram: $\hat{\beta_0}$ = -2,7628 e $\hat{\beta_1}$ = 0,9759 - (σ = 0,357; $R^2\ adj.$ = 0,9716; AIC = 3130; DF = 4002). Esse modelo foi denominado "Modelo Pantropical" (MP), e também usa o fator de correção de Baskerville (1972) para corrigir as estimativas de biomassa individual.

Aqui, um parêntese apenas para destacar a importância do "Modelo Pantropical" proposto por Chave *et al.* (2014) no cenário mundial, inclusive recomendado pelo "Painel Intergovernamental sobre Mudanças Climáticas" (do inglês, *Intergovernmental Panel on Climate Change* - IPCC) para estimativas de biomassa em florestas tropicais. O MP encontra-se disponível para uso fácil e prático no pacote BIOMASS por meio da função computeAGB() (RÉJOU-MÉCHAIN *et al.*, 2017a,b).

As estimativas dos parâmetros do modelo linear foram realizadas de modo tradicional, sem emprego da validação cruzada. Essa decisão encontra apoio nas evidências do estudo de Antal Kozak e Robert Kozak, no qual reportaram que métodos de validação não fornecem informação adicional em comparação às estatísticas de modelos ajustados a partir de conjuntos de dados completos (KOZAK; KOZAK, 2003). Apesar disso, para uma comparação mais razoável entre as abordagens, o modelo $ln(BAT) = \beta_0 + \beta_1\ ln(\rho\ D^2H) + \epsilon_i$ foi ajustado sob o conjunto de treinamento (n = 3226), usado na construção de modelos de aprendizado de máquina, e o conjunto de teste (n = 778) foi reservado para uma comparação mais apropriada da precisão do modelo linear e MAM na predição de amostras futuras. Para fins didáticos, o modelo ajustado usando o conjunto de treino será chamado de **Modelo Alternativo** - MA .

Ao ajustar o modelo estatístico $ln(BAT) = \beta_0 + \beta_1\ ln(\rho\ D^2H) + \epsilon_i$ sob o conjunto de treino, o desejável é encontrar estimativas de parâmetros e estatísticas de qualidade de ajuste similares à Chave *et al.* (2014). Os parâmetros estimados, erros-padrões, Intervalos de Confiança (ICs) para os parâmetros estimados e as estatísticas de ajuste no conjunto de treino diferiram apenas marginalmente em relação ao ajuste com dados completos (TABELA 11). A semelhança da distribuição residual entre os modelos também foi constatada (FIGURA 29).

Os resultados similares entre os modelos "MP" e "MA", embora pareçam intrigantes, encontram respaldo na clássico estatístico publicado por Anscombe (1973), que provou que quatro conjuntos de dados podem ter as mesmas propriedades estatísticas, inclusive com estimativas idênticas dos parâmetros de regressão, apesar de mostrarem relações bivariadas extremamente diferentes. Esse problema clássico ficou conhecido como "Quarteto de Anscombe". Como já mencionado as propriedades estatísticas dos conjuntos de treino e completo são bastante similares (ver TABELA 9). Portanto, por um lado este fato revela que o processo de amostragem estratificada baseada nos sítios foi eficaz em reproduzir o espaço de características do conjunto de dados completo, e por outro pode explicar a semelhança das estimativas dos modelos "MP" e "MA".

Em se tratando de MCRL, alguns estudos preocupam-se apenas com a estimativa pontual, isto é, especificada uma forma funcional $f(x)$ o objetivo é encontrar estimativas dos parâmetros (β_i) e, em seguida, usar o modelo para predizer $\hat{y}_i$ dado observações reais de preditores, x_i. No entanto, o MCRL também foi desenvolvido sob um sólido arcabouço teórico para realização de inferências estatísticas consistentes. Para tanto, uma série de suposições (normalidade, homocedasticidade e independência) são fixadas sobre o termo

de erro estocástico (ϵ_i) para validar as interpretações inferenciais do modelo. Portanto, caso haja violação de algum dos pressupostos do MCRL, as inferências realizadas com base nos estimadores de MQO tornam-se pouco confiáveis.

TABELA 11 – COMPARAÇÃO ENTRE MODELOS CLÁSSICOS DE REGRESSÃO LINEAR AJUSTADOS AO CONJUNTO DE DADOS COMPLETO (MODELO PANTROPICAL - n=4.004) E AO CONJUNTO DE TREINO (MODELO ALTERNATIVO).

Modelo (n)	Parâmetros estimados	Erro Padrão	*p*-valor	DF	IC (95%) parâmetros estimados		RSE	$RSE(\%)$	r	R^2_a	MAE	AIC
					2,5%	97,5%						
MP (4.004)	$\hat{\beta}_0$ = -2,762824 $\hat{\beta}_1$ = 0,975891	0,0211 0,0026	0,000 0,000	4.002	-2,8042 0,9707	-2,7214 0,9811	0,3575	7,50	0,9857	0,9716	0,2737	3.130,68
Modelo Alternativo (3.236)	$\hat{\beta}_0$ = -2,757009 $\hat{\beta}_1$ = 0,975193	0,0235 0,0029	0,000 0,000	3.224	-2,8031 0,9694	-2,7109 0,9809	0,3588	7,52	0,9857	0,9716	0,2746	2.548,35

FONTE: O autor (2020).

NOTA: Forma funcional especificada: $ln(BAT) = \beta_0 + \beta_1\ ln(\rho\ D^2H) + \epsilon_i$. Todas as métricas foram calculadas na escala da variável resposta do modelo estatístico. O IC é dado por: $\text{IC}^{\beta_i}_{0,95} = \left[\hat{\beta}_i - 1,96 \times EP(\hat{\beta}_i)\ ,\ \hat{\beta}_i + 1,96 \times EP(\hat{\beta}_i)\right]$.

LEGENDA: MP = Modelo Pantropical; n = número de amostras; IC = Intervalo de Confiança; RSE = Residual Standard Error; $RSE(\%)$ = Residual Standard Error, em %; r = Coeficiente de correlação de Pearson; R^2_a = Coeficiente de determinação ajustado; MAE = Mean Absolute Error; DF = Degree of Freedom; AIC = Akaike Information Criterion.

Para avaliar a validade das estimativas de MQO para fins inferenciais (testes de hipóteses e intervalos de confiança), os gráficos de resíduos versus valores preditos e quantil-quantil foram analisados (FIGURA 29), e além disso testes de hipóteses para constatação da normalidade (*Shapiro-Wilk*, α = 0,05) e homocedasticidade (*Breusch-Pagan*, α = 0,05) também foram realizados.

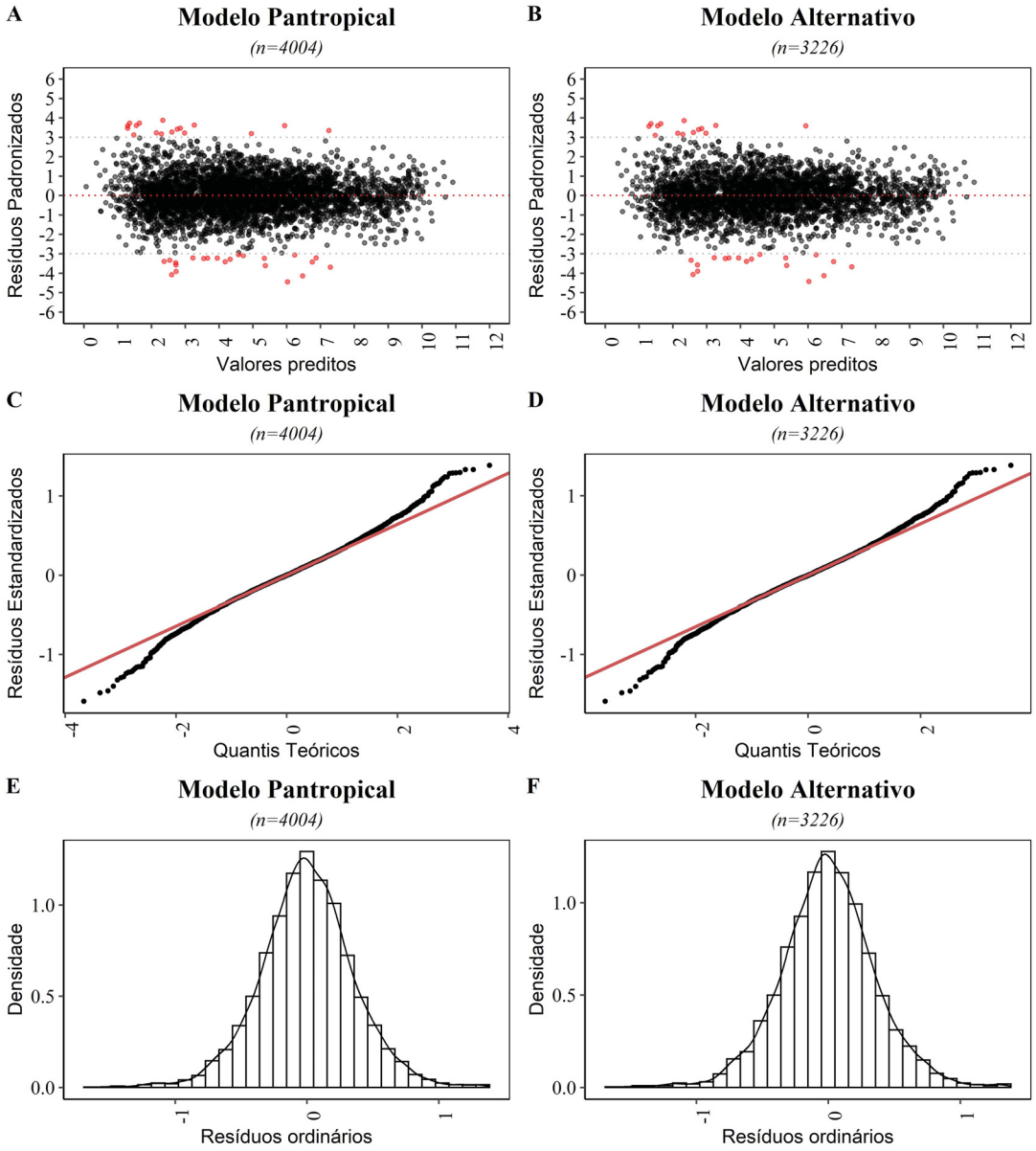
A hipótese de homocedasticidade (ou variação constante) do termo de erro estocástico ϵ_i é, em geral, avaliada através do exame informal dos resíduos $\hat{\epsilon}_i$. Ambos os modelos apresentaram uma quantidade razoável de resíduos fora do intervalo $-3 \leq z_i \leq 3$, indicando a presença de pontos discrepantes. Além disso, através da inspeção visual é razoável admitir uma pequena variação dos resíduos para os diferentes valores estimados $\hat{y_i}$. O teste de hipótese *Breusch-Pagan* ratificou a rejeição da hipótese de homocedasticidade residual: (Modelo Linear (n = 3226): BP = 27,25; *p*-valor = 0,000) e (Modelo Pantropical (n = 4004): BP = 36,475; *p*-valor = 0,000).

Em MCRL, os estimadores de MQO ($\hat{\beta}_0$ e $\hat{\beta}_1$) possuem importantes propriedades estatísticas, ou seja, são lineares, não-viesados e possuem mínima variância na classe de todos os estimadores lineares não-viesados. Devido a isso, os estimadores de MQO são reconhecidos como “Melhores Estimadores Lineares Não-Viesados” (MELNV) (do inglês, *Best Linear Unbiased Estimator* - BLUE). Contudo, na condição de heterocedasticidade, os estimadores de MQO não são mais considerados os “melhores” ou de “variância miníma”. Em tal condição, os estimadores de variância dos estimadores de MQO são viesados e, por conseguinte, os erros-padrões e as inferências do modelo estatístico (intervalos de confiança, testes *t* e *F*) podem ser enganosas (GUJARATI; PORTER, 2011).

Outra suposição para os modelos clássicos de regressão linear é de que o termo de erro estocástico seja normalmente distribuído, com média zero e variância σ^2, $[\epsilon_i \sim N(0, \sigma^2)]$. Assim, violar a suposição de normalidade pode levar também à interpretações inferenciais

imprecisas (JARQUE; BERA, 1980). Outros, porém, ajuízam que a suposição de normalidade é menos importante, em especial, nos casos de grandes amostras (WEISBERG, 2005).

FIGURA 29 – GRÁFICOS DE RESÍDUOS PADRONIZADOS, QUANTIL-QUANTIL E HISTOGRAMA RESIDUAL PARA OS MODELOS LINEARES AJUSTADOS AO CONJUNTO DE DADOS COMPLETO (MODELO PANTROPICAL - n=4.004) E AO CONJUNTO DE TREINO (MODELO ALTERNATIVO - n=3.226).



FONTE: O autor (2020).

Em relação à distribuição residual, os gráficos Quantil-Quantil revelaram resíduos empíricos com desvios nas caudas direita e esquerda da distribuição, sugerindo não aderência à curva Gaussiana. Esses desvios foram atribuídos às árvores de pequeno e médio porte (diâmetro < 74cm e biomassa < 2,4Mg). O teste de hipótese de *Shapiro-Wilk* confirmou a rejeição da hipótese de normalidade da distribuição empírica dos resíduos para ambos os modelos: (Modelo Linear (n = 3.226): W = 0,99267; *p*-valor = 0,000) e (Modelo Pantropical (n = 4.004): W = 0,99284; *p*-valor = 0,000).

Gujarati e Porter (2011) afirma que, mesmo sob a condição de heterocedasticidade e não normalidade, os estimadores de MQO permanecem “não-viesados”, ou seja, os valores médios ou esperados, são iguais aos parâmetros verdadeiros. Portanto, para ambos os modelos (MP e MA), essa assertiva traduz a adequação dos modelos para estimativas pontuais, mas as inferências (intervalos de confiança, testes *t* e *F*) podem ser enganosas. Finalmente, dada similaridade entre os parâmetros, as estatísticas de qualidade e o comportamento das distribuições residuais, é razoável admitir que o modelo “MA” ajustado ao conjunto de treino (n = 3.226) possui poder preditivo similar ao “MP” (n = 4.004).

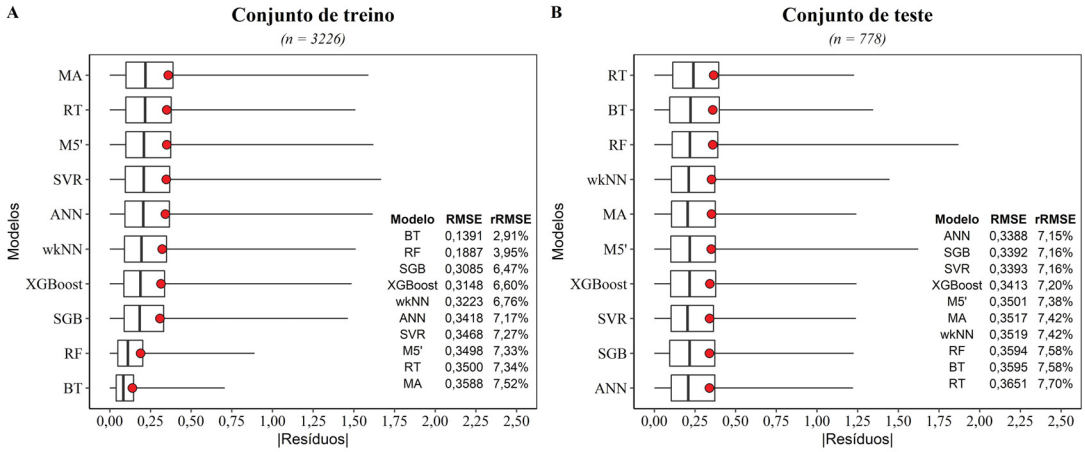
4.1.6 Comparação de abordagens: modelagem tradicional versus aprendizado de máquina

Na subseção 4.1.5 um MCRL foi ajustado usando apenas os dados do conjunto de treinamento (n=3226), designado Modelo Alternativo (MA), e considerado um bom representante do “MP”. A ideia da subseção corrente foi comparar o desempenho médio e a distribuição residual dos modelos de aprendizado de máquina com hiperparâmetros de ajuste ótimo frente ao modelo “MA”, usando um conjunto de dados independente (n=778). Além disso, apenas para fins comparativos o “erro de resubstituição”, calculado sobre o conjunto de treino, foi obtido (FIGURA 30). Em seguida, o desempenho do modelo ANN ajustado à base completa (denominado “9ANN02”), em nível de sítio *j*, é comparado ao MP, conforme abordagem usada em Chave *et al.* (2014).

Em termos gerais, os modelos (algorítmicos e MA) apresentaram semelhança da distribuição residual e do desempenho médio para predizer a variável resposta em amostras futuras, com rRMSE entre 7,15% (ANN) e 7,70% (RT). Portanto, as diferenças parecem ser apenas marginais. Avaliar o desempenho dos modelos de aprendizado de máquina sobre o conjunto de treino não é adequado, pois as estimativas podem ser excessivamente otimista, como para BT e RF. Assim, a avaliação sobre o conjunto independente do ajuste é mais realista e compatível com os indicativos da validação cruzada. Do mesmo modo, o comportamento preditivo semelhante dos modelos foi ratificado através da análise do gráfico da função de distribuição cumulativa empírica inversa (do inglês, *Reversed Empirical Cumulative Distribution Function* - RECDF) para os resíduos absolutos (FIGURA 31).

Para comparar o modelo “9ANN02” (ajustado à base completa) e o MP, as medidas de Coeficiente de Variação (CV) e Viés, em nível de sítio *j*, foram calculadas usando as expressões dadas pelas equações 3.2 e 3.1 da subseção 3.1.2. Também para uma comparação sem tendências a modelagem preditiva feita por Chave *et al.* (2014) foi replicada e, assim, obteve valores estimados de AGB fiéis à análise original. Aqui, porém, o viés médio em nível de sítio *j* encontrado para o MP foi modestamente mais otimista (+5.27%) em comparação ao mencionado em Chave *et al.* (2014) (+5.31%). É provável que pequenas discordâncias nos valores originais das variáveis podem ser a motivação da diferença.

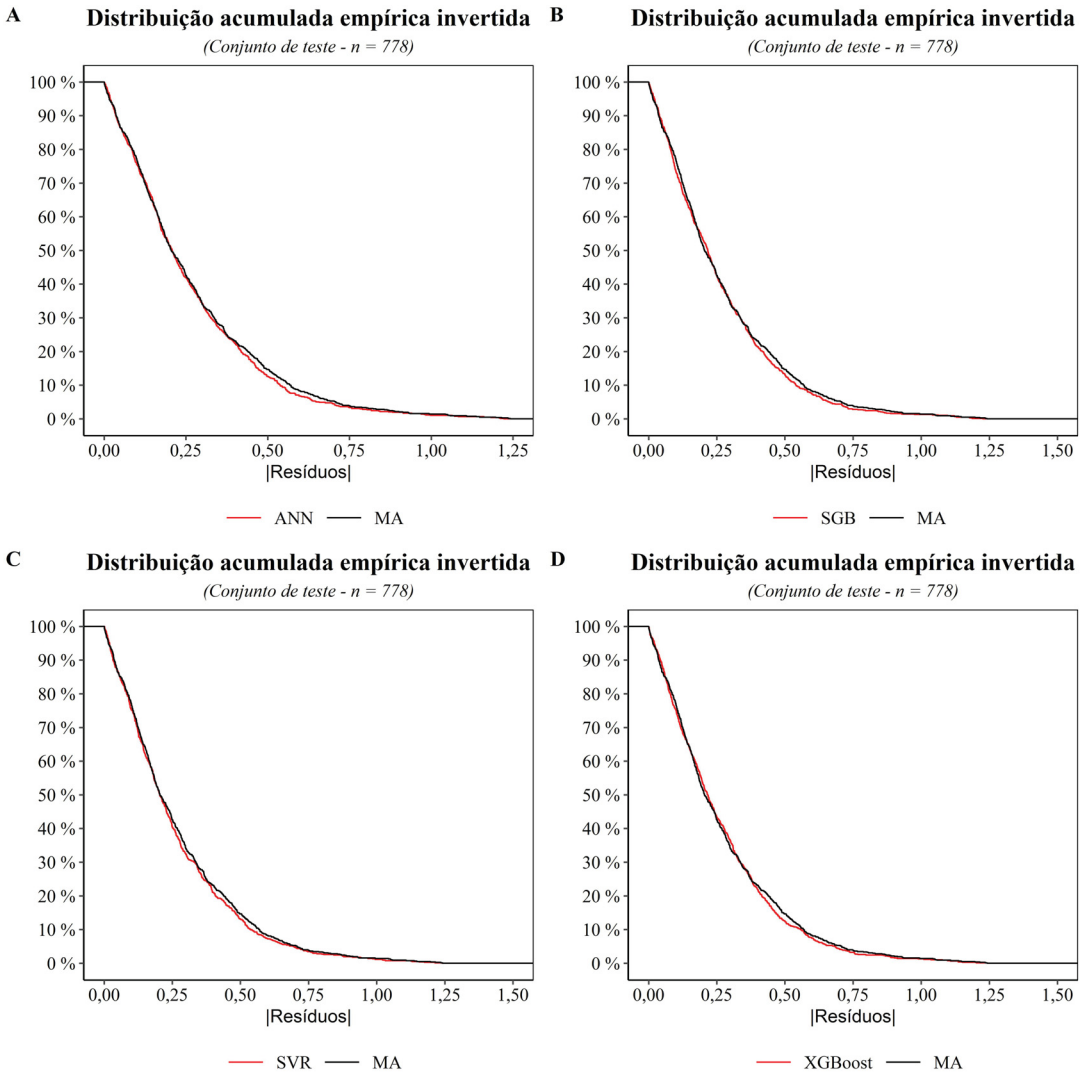
FIGURA 30 – COMPARAÇÃO DE RESÍDUOS ABSOLUTOS NOS CONJUNTOS DE TREINO E TESTE PARA OS MODELOS DE APRENDIZADO DE MÁQUINA E MODELO CLÁSSICO DE REGRESSÃO LINEAR.



FONTE: O autor (2020).
NOTA: Elaborado com auxílio do pacote DALEX (BIECEK, 2018).
LEGENDA: ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor ; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); MA = Modelo alternativo de regressão linear; RMSE = Root Mean Square Error; rRMSE = Relative Root Mean Square Error. O ponto em vermelho representa o RMSE.

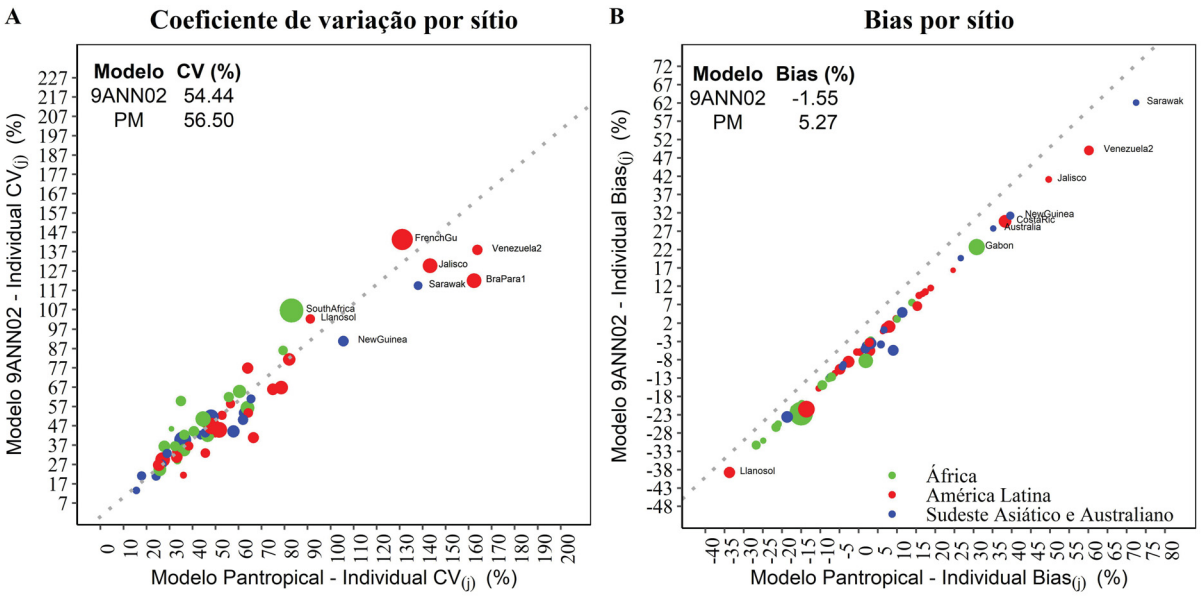
A média de $CV_{(j)}$ em todos os sítios usando o modelo "9ANN02" foi de 2,06% menor do que o MP, evidenciando uma pequena melhoria na predição da biomassa em nível de sítio *j*. Do total de 58 sítios *j*, aproximadamente 57% (33 sítios) apresentaram valores de coeficiente de variação menores quando usado o modelo "9ANN02" para predizer a biomassa individual de árvores (FIGURA 32A). O viés médio usando o modelo "9ANN02" foi de -1,55%, indicando que, em média, as predições em nível de sítio *j* estão mais próximas da biomassa empírica do que aquelas providas pelo MP (FIGURA 32B).

FIGURA 31 – COMPARAÇÃO DA DISTRIBUIÇÃO CUMULATIVA EMPÍRICA INVERTIDA PARA OS RESÍDUOS ABSOLUTOS ENTRE QUATRO MODELOS DE APRENDIZADO DE MÁQUINA E O MODELO ALTERNATIVO DE REGRESSÃO LINEAR



FONTE: O autor (2020).
NOTA: Elaborado com auxílio do pacote DALEX (BIECEK, 2018).
LEGENDA: ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; MA = Modelo alternativo de regressão linear.

FIGURA 32 – COMPARAÇÃO DO DESEMPENHO, EM NÍVEL DE SÍTIO, ENTRE O MELHOR MODELO ALGORÍTMICO (ANN; SIZE=9, DECAY=0,2) AJUSTADO À BASE COMPLETA E O MODELO PANTROPICAL.



FONTE: O autor (2020).

LEGENDA: **A)** Relação entre o coeficiente de variação (CV) em nível de sítio para o modelo “9ANN02” e MP. A dimensão de cada ponto é proporcional ao tamanho da amostra (n) em cada sítio *j*. Os sítios com CV maior do que 100% em qualquer direção foram rotulados. **B)** Relação entre o viés individual em cada sítio *j* para o modelo “9ANN02” e MP. A dimensão de cada ponto é proporcional à quantidade de biomassa aérea total em cada sítio *j*. Os sítios com viés maior do que 30% em qualquer direção foram rotulados.

4.1.7 **MLMBio** - Aplicação na web com modelos de aprendizado de máquina para predição da biomassa aérea total de árvores em florestas tropicais

MLMBio é uma aplicação web com modelos de aprendizado de máquina para predição da biomassa aérea total de árvores em florestas tropicais desenvolvida usando o framework shiny. A aplicação web foi desenvolvida para disponibilizar de modo fácil e prático os dois modelos mais acurados encontrados para estimar a BAT: Modelo ANN (size = 9; decay = 0,2) e Modelo SGB (interaction.depth = 5; shrinkage = 0,01; n.trees = 1500; n.minobsinnode = 5). Os modelos foram ajustados ao conjunto de dados completo (n = 4004) e disponibilizados na aplicação web no seguinte endereço: https://deivisonsouza.shinyapps.io/MLBiomass/. O modelo pantropical desenvolvido por Chave *et al.* (2014) também está disponível para comparação com os MAM.

Na modelagem da biomassa, os MAM foram treinados usando as seguintes nomenclaturas para as variáveis: **D** = diâmetro 1,30m do solo (centímetros); **H** = altura total (metros); **WSG** = densidade básica da madeira (g.cm^{-3}); e **AGB** = Biomassa aérea total (kg). Portanto, a mesma nomenclatura de variáveis deve ser usada para uma nova base de dados. Assim, os modelos construídos encontrarão as variáveis no conjunto de dados e, portanto, as predições serão realizadas. Também é importante ater-se às unidades de medidas usadas para cada variável.

A aplicação web dispõe de um "botão" para upload de arquivos do tipo .csv com até 100 megabytes. Além disso, a interface de usuário possui oito menus de fácil interatividade que auxiliam na navegação do site e extração de informações do conjunto de dados: 1) About the app; 2) About file; 3) Dataset; 4) Explore; 5) Prediction; 6) Compare; 7) Metrics; e 8) Reference. A segui as principais funcionalidades dos menus são descritas. Um video interativo mostrando o funcionamento da aplicação foi desenvolvido (FIGURA 33).

1. **About the app**: apresenta um breve relato sobre a aplicação desenvolvida. Informa sobre o conjunto de dados usado na modelagem preditiva. Relata a linguagem de programação, os algoritmos e pacotes usados para aprendizado dos modelos preditivos. Destaca os MAM encapsulados e como as variáveis devem ser nomeadas para que os modelos possam realizar corretamente as predições.

2. **About file**: apresenta uma descrição básica do conjunto de dados carregados. Revela o tipo de variáveis e a dimensão do conjunto de dados (quantidade de observações e número de variáveis).

3. **Dataset**: revela o conjunto de dados carregado e possibilita salvar em diferentes extensões (.csv, .xlsx, .pdf).

4. **Explore**: Neste menu, quatro submenus estão disponíveis: **Summary**, **Histogram**, **Scaterplot** e **BoxPlot**. No submenu **Summary**, o usuário tem acesso a uma rápida estatística descritiva para as variáveis numéricas. Em **Histogram**, histogramas de frequência iterativos, para cada variável, podem ser criados. Ao passar o curso sobre o(s) gráfico(s) construídos, o usuário pode consultar para cada classe: i) ponto médio, limites inferior e superior e frequência absoluta. No submenu **Scaterplot**, gráficos de dispersão podem ser criados para mostrar a relação marginal entre as variáveis. Um análise de correlação de Pearson é realizada para a relação específica. Do mesmo modo, ao passar o curso sobre o(s) gráfico(s) construído(s) informações de cada ponto são reveladas. No submenu **BoxPlot**, gráficos boxplots podem ser construídos, e as informações de mínimo, máximo, primeiro e terceiro quartil, mediana e potenciais *outliers* são interativamente visualizadas. Todos os gráficos gerados podem ser salvos no formato .png, e o sumário estatístico pode ser salvo em .csv e .xlsx.

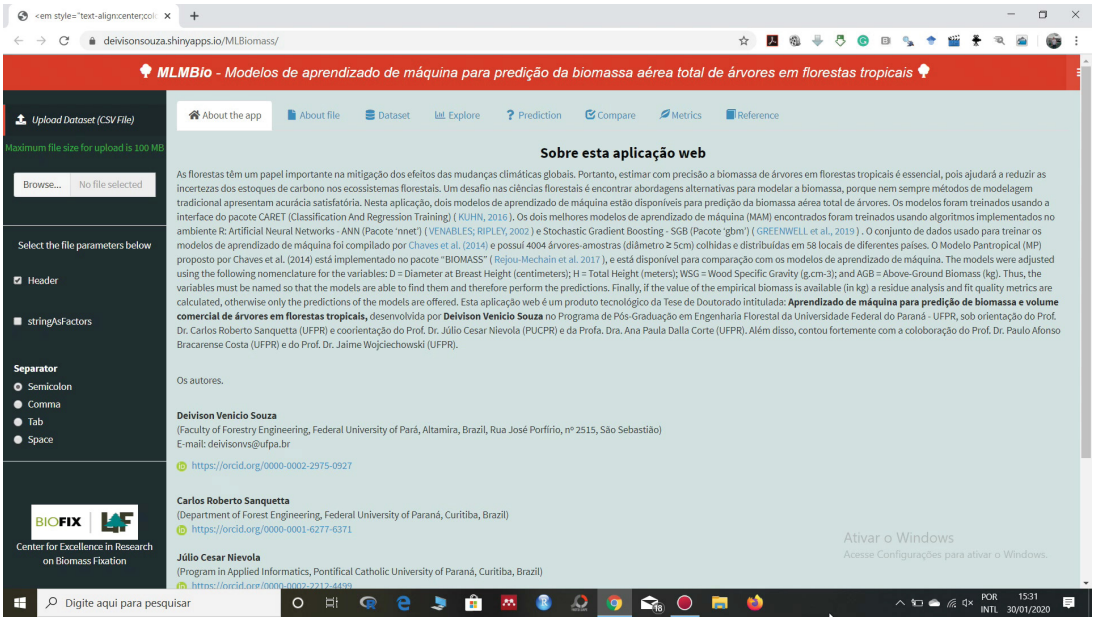
5. **Prediction**: Neste menu, os MAM e o modelo pantropical são usados para realizar as predições de BAT. Na prática, duas situações são possíveis: 1) quando a biomassa empírica está disponível; e 2) quando a biomassa empírica não está disponível. Na primeira situação, um quadro de dados é gerado com as variáveis originais (**D**, **H**, **WSG** e **AGB**), três colunas com predições (**Pred_PM**, **Pred_ANN** e **Pred_SGB**) e três colunas com os resíduos ordinários (**res_PM**, **res_ANN** e **res_SGB**). Em que, **Pred_PM**: predição da BAT usando o modelo pantropical de Chave *et al.* (2014); **Pred_ANN**: predição da BAT usando o modelo ANN (size = 9; decay = 0,2); **Pred_SGB**: predição da BAT usando o modelo SGB (interaction.depth = 5; shrinkage = 0,01; n.trees = 1500; n.minobsinnode = 5); **res_PM**: resíduos ordinários para o modelo pantropical; **res_ANN**: resíduos ordinários para o modelo ANN; **res_SGB**: resíduos ordinários para o modelo SGB. Na segunda situação, uma vez que a biomassa empírica não está disponível, não é possível realizar o cálculo de resíduos.

6. **Compare**: Neste menu, gráfico(s) de valores observados como função dos preditos podem ser visualizados. Diversas métricas de desempenho são calculadas (MSE, RMSE, RMSE%, r, R^2, Bias, Bias%, MAE, MAPE). Além disso, um quadro interativo revela os maiores resíduos, para cada modelo. Os gráficos e tabela com resíduos ranqueados, em ordem decrescente, estão disponíveis para download.

7. **Metrics**: apresenta as expressões matemáticas das métricas calculadas no menu **Compare**.

8. **Reference**: lista os pacotes da linguagem R usados para a construção da aplicação web.

FIGURA 33 – ANIMAÇÃO DEMONSTRANDO O FUNCIONAMENTO DA APLICAÇÃO WEB MLMBIO.



FONTE: O autor (2020).
NOTA: O arquivo com a animação pode ser acessado no endereço: https://github.com/DeivisonSouza/PhD-thesis.

4.1.8 Conclusão

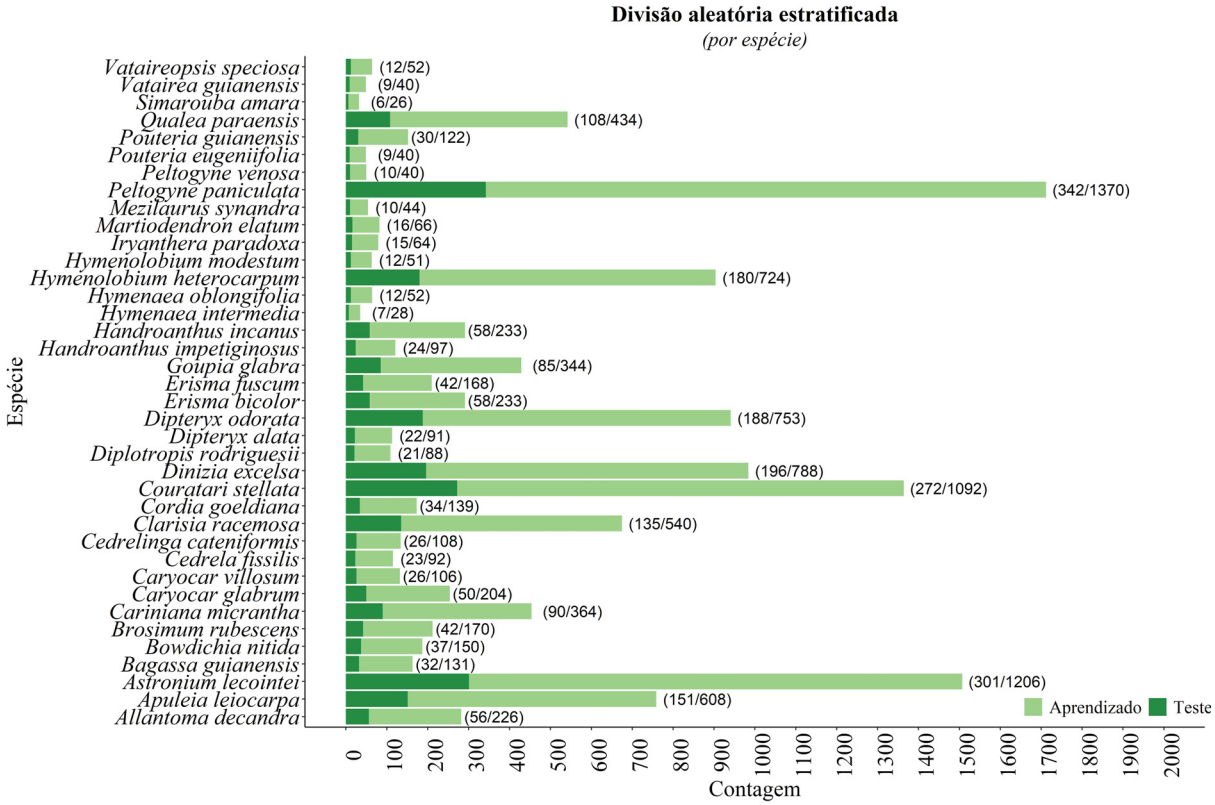
- Apesar do ganho em precisão dos modelos de aprendizado de máquina frente à regressão tradicional ser julgado mínimo (se algum), a abordagem algorítmica pode ser considerada uma alternativa potencial para obtenção de modelos genéricos de predição de biomassa individual de árvores em florestas naturais inequiâneas, até porque modelos genéricos precisos nem sempre são encontrados com uso da técnica de regressão clássica, ou mesmo com outras abordagens paramétricas mais flexíveis.
- A tarefa de engenharia de recursos é fator crucial na construção de modelos de aprendizado de máquina, pois os preditores apresentam desigual importância para os diferentes algoritmos e, isso foi provado pela estimativa de importância de variáveis. Portanto, se realizada adequadamente melhorias de desempenho podem ser alcançadas, sendo o conhecimento da área de domínio um aspecto diferencial no processo de construção do espaço de preditores.
- É possível interpretar modelos de aprendizado de máquina, e inúmeras técnicas têm sido desenvolvidas, como a estimativa de importância relativa de preditoras e dependência parcial entre preditor-resposta no modelos preditivos construído. O desenvolvimento de métodos interpretativos têm desmistificado a metáfora da “caixa preta”, e muito valor tem agregado às áreas de conhecimentos cuja interpretabilidade do modelo é fundamental para as tomadas de decisão.

4.2 **ESTUDO DE CASO 2:** MODELOS TRADICIONAIS E APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DE VOLUME COMERCIAL DE ESPÉCIES MANEJADAS NA AMAZÔNIA BRASILEIRA

4.2.1 Análise exploratória e divisão de dados

O conjunto de dados original (n = 13.831) possui três variáveis dendrométricas (volume comercial, diâmetro e altura) medidas de árvores de 38 espécies florestais manejadas na Amazônia brasileira. Para fins de modelagem preditiva, foram admitidas apenas as espécies com no mínimo 30 indivíduos na amostra completa. O conjunto completo foi dividido em bases de treinamento (80%; n = 11.084) e teste (20%; n = 2.747), usando de amostragem aleatória estratificada em nível de espécie (FIGURA 34).

FIGURA 34 – CONJUNTO DE DADOS DE TREINAMENTO E TESTE APÓS DIVISÃO ALEATÓRIA ESTRATIFICADA EM NÍVEL DE ESPÉCIE.



FONTE: O autor (2020).
NOTA: Rótulos por espécie: (N_{teste}/N_{treino}). N = número de amostras.

As estatísticas descritivas amostrais de cada variável (média, desvio padrão e forma das distribuições) dos conjuntos de treinamento e teste foram equiparáveis ao conjunto de dados completo (TABELA 12). Portanto, é razoável admitir que os padrões contidos nos dados completos foram adequadamente cobertos nos conjuntos de treino e teste. O conjunto de treinamento abrangeu os mínimos e máximos das distribuições empíricas das variáveis, com exceção do mínimo da variável volume comercial.

No conjunto completo, o diâmetro das árvores apresentou Coeficiente de Variação (CV) igual a 26,09%, com mínimo e máximo de 50cm e 245cm, respectivamente. A variável

altura mostrou menor variância (CV = 20,41%). A variável resposta (volume, em m³) exibiu elevada dispersão (CV = 62,75%). Para as variáveis ”diâmetro“ e ”altura comercial“, a variância entre indivíduos da mesma espécies apresentou valores inferiores à 30% (TABELA 13).

Em geral, as variáveis biométricas apresentaram distribuições com assimetria positiva (g_1 > 0) e leptocúrticas (g_2 > 0). Distribuições com assimetria positiva possuem uma cauda mais alongada para a direita ou, intuitivamente, uma distribuição com maior concentração de dados à esquerda. Uma curva leptocúrtica é mais alongada do que a distribuição Gaussiana (0,263). A variável volume apresentou maior grau de assimetria e curtose.

TABELA 12 – COMPARAÇÃO DAS PROPRIEDADES ESTATÍSTICAS DO CONJUNTO ORIGINAL, TREINAMENTO E TESTE PARA DADOS DE VOLUME COMERCIAL DE ESPÉCIES AMAZÔNICAS.

Variáveis	**Dados completos**						
	n	Mínimo	Máximo	Média	DP	Assimetria	Curtose
d		50	245	84,09	21,94	1,44	2,86
h	13.831	6	35	17	3,47	0,37	0,22
v		1,11	49,41	7,41	4,65	2,01	6,01
	Dados de treinamento						
d		50	245	84,15	22,09	1,45	2,88
h	11.084	6	35	17	3,46	0,36	0,18
v		1,23	49,41	7,43	4,67	2,01	6,03
	Dados de teste						
d		50,29	207,54	83,84	21,34	1,42	2,75
h	2.747	7	34	16,98	3,5	0,41	0,38
v		1,11	38,53	7,34	4,58	2,01	5,9

FONTE: O autor (2020).
LEGENDA: n = número de observações; v = volume comercial (m³); d = diâmetro a 1,30m do solo (cm); h = altura comercial (m); DP = Desvio padrão

A elevada heterogeneidade de medidas biométricas de árvores é uma condição intrínseca em florestas naturais inequiâneas. Alguns aspectos, como diversidade de espécies e idades, condições edafoclimáticas e o estabelecimento de competição intra e interespecífica por recursos, são fatores influentes no desenvolvimento das árvores. Cysneiros *et al.* (2017), por exemplo, reportaram valores de 72,19% e 68,74% para a variável ”volume comercial“ sem e com remoção de *outliers*, respectivamente, para dados de romaneio de 32 espécies florestais manejadas da Amazônia brasileira.

O coeficiente de correlação de Pearson (r) foi usado para medir o grau de associação linear bivariada e a direção da relação (positiva ou negativa) entre as variáveis dos modelos estatísticos. Na maioria das relações bivariadas entre variáveis dependentes (v e $ln(v)$) e independentes as correlações foram superiores a 86%. As variáveis de interação d^2h e $ln(d^2H)$ mostraram maior correlação linear com v e $ln(v)$, respectivamente. Por outro lado, a variável altura (escala original ou transformada) apresentou uma relação linear positiva fraca ($0,3 \leq r \leq 0,42$) com as variáveis respostas. Todas as correlações foram significativas através do teste *t*-Student ao nível de 1% de probabilidade ($p \leq 0,01$) (FIGURA 35).

TABELA 13 – ESTATÍSTICA DESCRITIVA DAS VARIÁVEIS DENDROMÉTRICAS, POR ESPÉCIE, PARA O CONJUNTO DE DADOS COMPLETO.

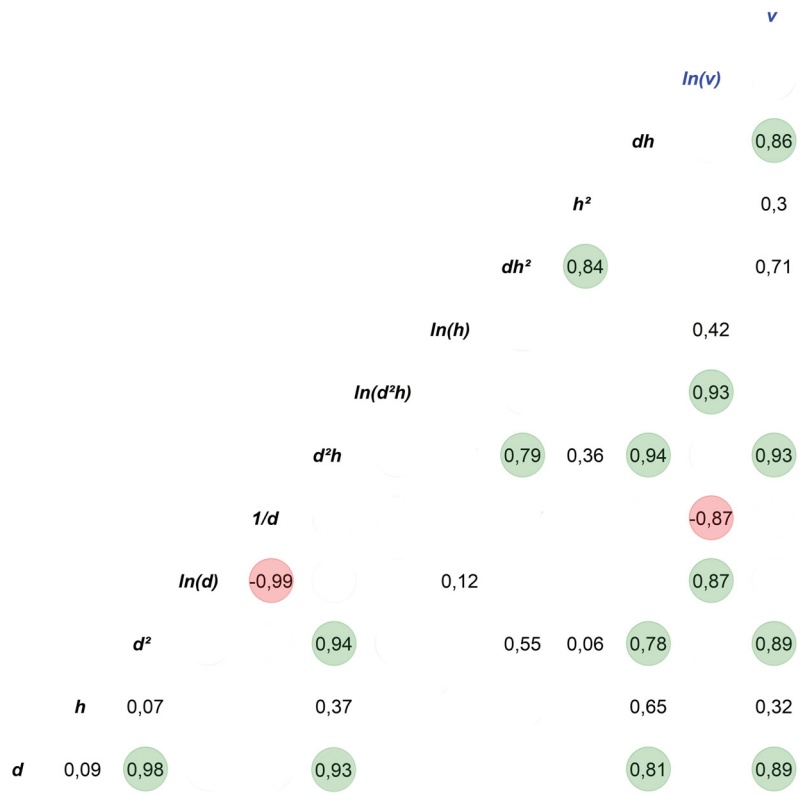
Nome Científico	n	Diâmetro (cm)				Altura (m)				Volume (m³)			
		Mín.	Máx.	Média	CV(%)	Mín.	Máx.	Média	CV(%)	Mín.	Máx.	Média	CV(%)
Allantoma decandra	282	50,00	147,70	84,75	21,95	10,00	28,00	17,73	14,69	1,89	20,11	7,49	47,20
Apuleia leiocarpa	759	60,16	173,48	95,19	21,44	8,00	27,00	16,83	12,86	2,04	24,43	8,78	46,47
Astronium lecointei	1507	50,29	133,69	79,74	16,73	10,00	35,00	21,28	13,75	2,61	21,65	8,22	37,14
Bagassa guianensis	163	56,98	148,97	93,22	21,11	10,00	23,00	16,84	12,84	2,27	20,51	8,43	37,70
Bowdichia nitida	187	50,29	112,68	68,95	17,02	11,00	25,00	17,46	14,26	2,32	12,96	5,14	41,23
Brosimum rubescens	212	50,61	146,42	80,15	19,40	10,00	30,00	16,04	15,72	2,20	18,14	6,09	45,59
Cariniana micrantha	454	50,93	195,76	117,03	23,13	12,00	27,00	18,53	13,81	2,44	37,62	14,78	47,93
Caryocar glabrum	254	51,25	150,00	84,49	21,82	8,00	22,00	13,98	19,06	2,07	19,49	5,97	52,42
Caryocar villosum	132	57,30	184,62	98,01	22,55	8,00	20,00	14,52	16,63	1,85	27,97	8,57	50,88
Cedrela fissilis	115	52,20	165,84	86,14	24,58	9,00	19,00	14,19	16,19	1,89	18,62	5,93	54,15
Cedrelinga cateniformis	134	51,57	205,00	104,60	28,36	10,00	24,00	16,42	14,58	2,07	38,55	10,81	67,35
Clarisia racemosa	675	50,29	114,59	70,79	13,19	8,00	22,00	15,03	14,93	1,79	12,39	4,42	33,01
Cordia goeldiana	173	52,20	120,96	73,28	16,76	14,00	25,00	20,39	13,62	2,54	14,70	6,47	36,60
Couratari stellata	1364	54,75	190,99	93,18	21,40	10,00	30,00	20,14	13,12	2,45	34,94	10,68	46,21
Dinizia excelsa	984	54,43	245,00	107,34	28,10	7,00	23,00	15,44	15,72	1,71	49,41	11,18	57,86
Diplotropis rodriguesii	109	50,61	91,99	63,91	12,29	13,00	25,00	17,36	13,90	2,25	8,97	4,29	29,42
Dipteryx alata	113	50,93	116,18	72,98	17,77	10,00	20,00	15,88	12,61	2,10	14,13	5,15	41,61
Dipteryx odorata	941	50,29	171,89	75,79	21,28	6,00	25,00	14,84	15,58	1,11	25,62	5,17	52,26
Erisma bicolor	291	50,29	150,00	76,83	22,28	8,00	25,00	15,97	15,01	1,94	21,72	5,63	52,22
Erisma fuscum	210	53,16	134,00	76,02	18,54	12,00	27,00	16,44	13,87	2,68	15,88	5,73	38,45
Goupia glabra	429	53,00	159,47	84,76	20,22	6,00	20,00	13,98	18,04	1,79	17,83	5,83	47,43
Handroanthus impetiginosus	121	50,93	158,52	82,11	22,70	13,00	26,00	19,54	12,56	2,32	20,25	8,13	47,21
Handroanthus incanus	291	50,61	128,60	75,10	20,98	6,00	28,00	19,47	17,03	2,13	22,37	6,51	48,39
Hymenaea intermedia	35	54,00	132,00	75,30	21,45	12,00	26,00	17,46	19,07	2,63	17,54	6,21	43,67
Hymenaea oblongifolia	64	51,57	118,09	74,68	17,88	12,00	25,00	16,97	13,45	1,98	15,10	6,58	41,44
Hymenolobium heterocarpum	904	50,29	200,54	89,33	27,25	6,00	25,00	16,75	14,04	1,84	43,87	8,64	61,74
Hymenolobium modestum	63	51,88	133,69	77,47	24,43	8,00	23,00	14,67	19,28	1,85	21,80	5,46	63,16
Iryanthera paradoxa	79	51,25	119,68	74,38	14,75	12,00	25,00	18,59	17,66	2,70	11,78	6,18	28,18
Martiodendron elatum	82	57,30	133,69	82,67	20,52	12,00	25,00	16,60	15,70	2,15	17,18	6,31	46,82
Mezilaurus synandra	54	60,48	169,34	76,01	22,65	11,00	22,00	15,91	15,41	2,29	17,58	5,42	49,80
Peltogyne paniculata	1712	50,00	124,14	74,45	13,88	7,00	25,00	14,40	16,89	1,35	13,22	4,44	32,63
Peltogyne venosa	50	57,00	114,59	76,31	15,84	12,00	23,00	17,40	11,73	1,88	10,42	5,70	33,43
Pouteria eugeniifolia	49	51,57	113,00	69,99	17,47	10,00	20,00	13,76	15,53	1,53	12,01	4,18	48,45
Pouteria guianensis	152	50,61	104,09	65,03	11,69	8,00	22,00	15,77	13,84	1,85	7,08	3,95	24,69
Qualea paraensis	542	50,93	124,14	72,44	16,76	11,00	27,00	17,89	14,43	1,88	16,46	5,50	37,66
Simarouba amara	32	50,29	77,99	59,67	11,91	12,00	23,00	16,69	14,87	1,98	5,51	3,30	26,56
Vatairea guianensis	49	53,16	127,01	76,46	22,20	12,00	24,00	15,24	14,45	1,68	18,29	5,27	49,86
Vataireopsis speciosa	64	50,61	111,41	72,56	17,19	12,00	20,00	15,70	11,38	2,18	13,99	5,21	44,99

FONTE: O autor (2020).

LEGENDA: n = número de observações; Mín = Mínimo; Máx = Máximo, CV = coeficiente de variação.

As correlações entre variáveis independentes foram superiores a 0,75, na maioria dos casos, no conjunto de treinamento. Em particular, essa condição foi encontrada nos modelos de *Hohenadl-Krenn*, *Brenac*, *Näslund* e *Meyer*. Em modelos múltiplos, a inserção de variáveis independentes com correlações bivariadas superiores a 0,75 podem implicar em problemas de multicolinearidade (MAROCO, 2010). Adicionalmente, Gujarati e Porter (2011) advertem que a existência de correlações altas entre regressores pode ser uma condição suficiente, mas não necessária, para a existência de multicolinearidade. Portanto, Sileshi (2014) recomenda o uso da estatística VIF_j para detectar problemas de multicolinearidade, evitando suposições errôneas, sobretudo para modelos múltiplos com mais de dois regressores.

FIGURA 35 – MATRIZ DE CORRELAÇÃO LINEAR DE PEARSON ENTRE AS VARIÁVEIS DEPENDENTES E INDEPENDENTES DE MODELOS VOLUMÉTRICOS TRADICIONAIS, EM FLORESTA MANEJADA NA AMAZÔNIA BRASILEIRA.



FONTE: O autor (2020).

NOTA: Círculo verde = correlações positivas superiores à 0,75; Círculo vermelho = correlações negativas superiores à 0,75; As variáveis estão na diagonal da matriz (cor azul: variáveis respostas admitidas). Elaborado com auxílio do pacote “GGally” (SCHLOERKE *et al.*, 2018)

LEGENDA: v = volume (m³); d = diâmetro a 1,30m do solo (cm); h = altura comercial (m); ln = logaritmo neperiano.

4.2.2 Modelos tradicionais genéricos: estimação pontual, diagnóstico e inferência

Os modelos volumétricos ajustados explicaram entre 78% e 87% das variações ocorridas na variável resposta (TABELA 14). Em geral, os modelos de dupla entrada, que incorporam as variáveis altura e diâmetro, apresentaram as melhores estatísticas de qualidade de ajuste. Os escores da estatística $PRESS$ indicaram que os modelos logarítmicos de Spurr e Schumacher-Hall possuem similar capacidade de predizer amostras futuras. Quando apenas a variável diâmetro estiver disponível os modelos de Berkhout, Kopezky-Gerhardt e Husch são as melhores opções e parecem ter similar capacidade preditiva.

TABELA 14 – MODELOS TRADICIONAIS GENÉRICOS AJUSTADOS PARA PREDIZER O VOLUME COMERCIAL DE ESPÉCIES MANEJADAS DA AMAZÔNIA BRASILEIRA.

Modelo	Parâmetros	IC (95%) - parâmetros		R_a^2	RSE	$RSE(\%)$	$PRESS$	AIC	VIF
		2,75%	97,5%						
Bk	$\hat{\beta}_0$= -8,3156	-8,474859	-8,156437	0,7838	2,1713	29,22	52285,35	48646,07	
	$\hat{\beta}_1$ = 0,1871	0,185305	0,188965						
K-G	$\hat{\beta}_0$= 0,4715001	0,392554	0,550446	0,7848	2,1664	29,15	52062,17	48595,89	
	$\hat{\beta}_1$= 0,0009195	0,000911	0,000929						
H-K	$\hat{\beta}_0$= -3,8518802	-4,331921	-3,371840	0,7908	2,1358	28,74	50629,31	48282,26	30,57
	$\hat{\beta}_1$ = 0,0908421	0,080889	0,100795						
	$\hat{\beta}_2$ = 0,0004808	0,000432	0,000530						
H	$\hat{\beta}_0$= -6,942	-7,033052	-6,850278	0,7809	2,1856	29,41	52974,80	2069,65	
	$\hat{\beta}_1$ = 1,997	1,975933	2,017388						
B	$\hat{\beta}_0$= -4,859	-5,588178	-4,130225	0,7889	2,1453	28,87	51062,73	2039,84	41,54
	$\hat{\beta}_1$ = 1,617	1,483752	1,750580						
	$\hat{\beta}_2$ = -32,726	-44,092327	-21,360500						
S	$\hat{\beta}_0$ = 5,874e-01	0,526739	0,648011	0,8608	1,7419	23,44	33662,49	43761,73	
	$\hat{\beta}_1$ = 5,273e-05	0,000052	0,000053						
S(ln)	$\hat{\beta}_0$ = -8,9924	-9,073272	-8,911446	0,8624	1,7323	23,31	33275.52	-3916,94	
	$\hat{\beta}_1$ = 0,9332	0,926203	0,940117						
S-H	$\hat{\beta}_0$ = -8,9190	-9,000629	-8,837339	0,8643	1,7200	23,15	32808.61	-4026,74	1,01
	$\hat{\beta}_1$ = 1,9082	1,892297	1,924017						
	$\hat{\beta}_2$ = 0,8416	0,823286	0,859880						
N	$\hat{\beta}_0$ = 1,431e-01	-0,022813	0,308966	0,8634	1,7257	23,22	33091,35	43557,56	
	$\hat{\beta}_1$ = 2,861e-04	0,000235	0,000337						50,24
	$\hat{\beta}_2$ = 3,255e-05	0,000028	0,000037						131,35
	$\hat{\beta}_3$ = 5,081e-05	0,000028	0,000074						84,43
	$\hat{\beta}_4$ = -1,331e-03	-0,002616	-0,000046						24,22
M	$\hat{\beta}_0$ = 4,116e+00	2,095967	6,136738	0,8653	1,7140	23,07	32669,28	43408,05	
	$\hat{\beta}_1$ = -1,021e-01	-0,144411	-0,059843						856,68
	$\hat{\beta}_2$= 7,777e-04	0,000567	0,000988						879,72
	$\hat{\beta}_3$ = 8,704e-03	0,006175	0,011232						1533,97
	$\hat{\beta}_4$ = -5,268e-06	-0,000018	0,000007						1058,88
	$\hat{\beta}_5$ = -3,417e-01	-0,462513	-0,220820						171,32

FONTE: O autor (2020).

NOTA: Valor predito foi corrigido pelo fator apresentado em Baskerville (1972). As métricas foram calculadas considerando as estimativas na escala natural da variável resposta. O IC é dado por: $IC_{0,95}^{\beta_i} = \left[\hat{\beta}_i - 1,96 \times EP(\hat{\beta}_i)\,,\, \hat{\beta}_i + 1,96 \times EP(\hat{\beta}_i)\right]$.

LEGENDA: Bk = Berkhout; K-G = Kopezky-Gerhardt; H-K = Hohenadl-Krennt; H = Husch; B = Brenac; S = Spurr; S(ln) = Spurr (ln); S-H = Schumacher-Hall; N = Näslund; M = Meyer. $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\beta}_3$, $\hat{\beta}_4$ e $\hat{\beta}_5$ = parâmetros estimados dos modelos de regressão; ns = coeficiente não significativo (α = 0,05); IC = intervalo de confiança dos parâmetros estimados; R_a^2 = coeficiente de determinação ajustado; RSE = Residual Standard Error; $PRESS$ = Prediction Residual Error Sum of Squares; AIC = Critério de Informação de Akaike; VIF = Fator de Inflação de Variância.

Os modelos de Hohenadl-Krennt, Brenac, Näslund e Meyer apresentaram variáveis altamente colineares. Esta condição é esperada devido à presença de regressores com alta correlação linear nas formas funcionais. Schumacher-Hall foi o único modelo linear múltiplo

sem indicativo de problemas de multicolinearidade (VIF < 5), com $ln(d)$ e $ln(h)$ pouco correlacionadas linearmente (r = 0,12). Também, Näslund e Meyer incorporam mais parâmetros, porém sem melhorias expressivas nas estatísticas de qualidade do ajustamento, provavelmente devido a existência de variáveis altamente correlacionadas nas estruturas dos modelos.

Em termos teóricos, Gujarati e Porter (2011) destacam que mesmo na presença de alta multicolinearidade os estimadores de Mínimos Quadrados Ordinários (MQO) ainda mantêm a propriedade de "Melhores Estimadores Lineares Não-Viesados" (MELNV). Apesar disso, algumas consequências práticas são: i) variância inflada dos estimadores de MQO, que prejudica a estimação dos erros padrões das estimativas dos parâmetros. Por conseguinte, os intervalos de confiança associados aos coeficientes tendem a ser mais amplos (menos informativos); ii) a razão *t* de um ou mais coeficientes tende a ser estatisticamente insignificante, mesmo para modelos com alto R^2 (GUJARATI; PORTER, 2011); e iii) estimativas não coerentes dos parâmetros da regressão, ou seja, coeficientes com sinais negativos para efeitos positivos esperados e vice-versa (SILESHI, 2014).

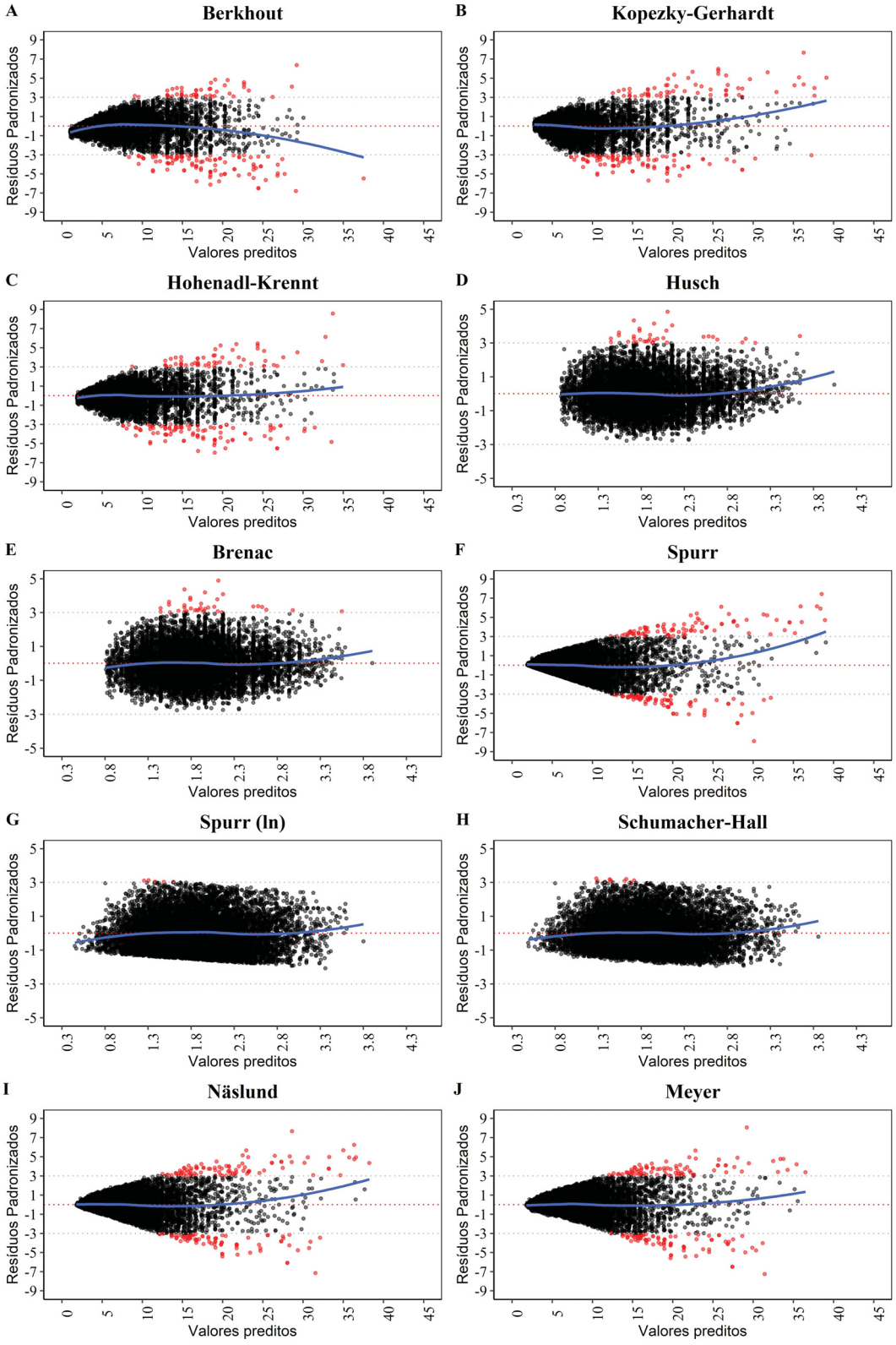
Em modelos clássicos de regressão linear, a confiabilidade da inferência estatística sobre os parâmetros populacionais (testes de hipóteses e intervalos de confiança) depende do atendimento às suposições (normalidade, homocedasticidade e independência) impostas sobre o termo de erro estocástico, ϵ_i. Nesta perspectiva, gráficos dos resíduos padronizados dos MQO ($\hat{\epsilon}_i$) contra os valores estimados da variável dependente, $\hat{y}_i$ (FIGURA 36) e também os diagramas Quantil-Quantil Normal (Q-Qplot) (FIGURA 37) são bastante informativos. As violações dos pressupostos do modelo podem ocorrer apesar dos resíduos parecerem bem comportados, pois existe subjetividade na análise. Portanto, testes de hipóteses formais também são recomendados.

Neste estudo, usar RL modelar o volume na escala natural, sem qualquer transformação, não constituiu a abordagem mais adequada, pois foi constatado um aumento da variância residual com o acréscimo dos valores estimados da variável dependente $\hat{y}_i$ (forma de megafone). Quando a suposição de variância igual é violada um procedimento comum é realizar uma transformação da variável resposta y, ou da(s) preditora(s) x, ou de ambas (MORETTIN; BUSSAB, 2017). Ainda, quando existe assimetria à direita é útil realizar uma transformação logarítmica da variável resposta (CASSON; FARMER, 2014). Aqui, em particular, esta condição (assimetria positiva) foi encontrada para o volume comercial (ver TABELA 13), e constitui uma tendência da variável volume de madeira em florestas naturais inequiâneas (SILVA; SANTANA, 2015).

Naturalmente, alguns modelos tradicionais (Husch, Brenac, Spurr (ln) e Schumacher-Hall) incorporam a abordagem de transformações logarítmicas da variável resposta e regressor(es). O teste de *Breusch-Pagan* (BP) indicou rejeição da hipótese de homocedasticidade do termo de erro estocástico para todos os modelos. Apesar disso, é razoável admitir que o modelo de Husch esteve mais próximo da não rejeição da hipótese de homocedasticidade (B-P = 7,0929; p = 0,0077), devido ao p-valor mais próximo ao nível de significância estabelecido (α = 0,05). Ainda, para os modelos de Spurr (ln) e Schumacher-Hall é perceptível a existência de padrão residual heterocedástico, sem disposição aleatória em torno da média zero.

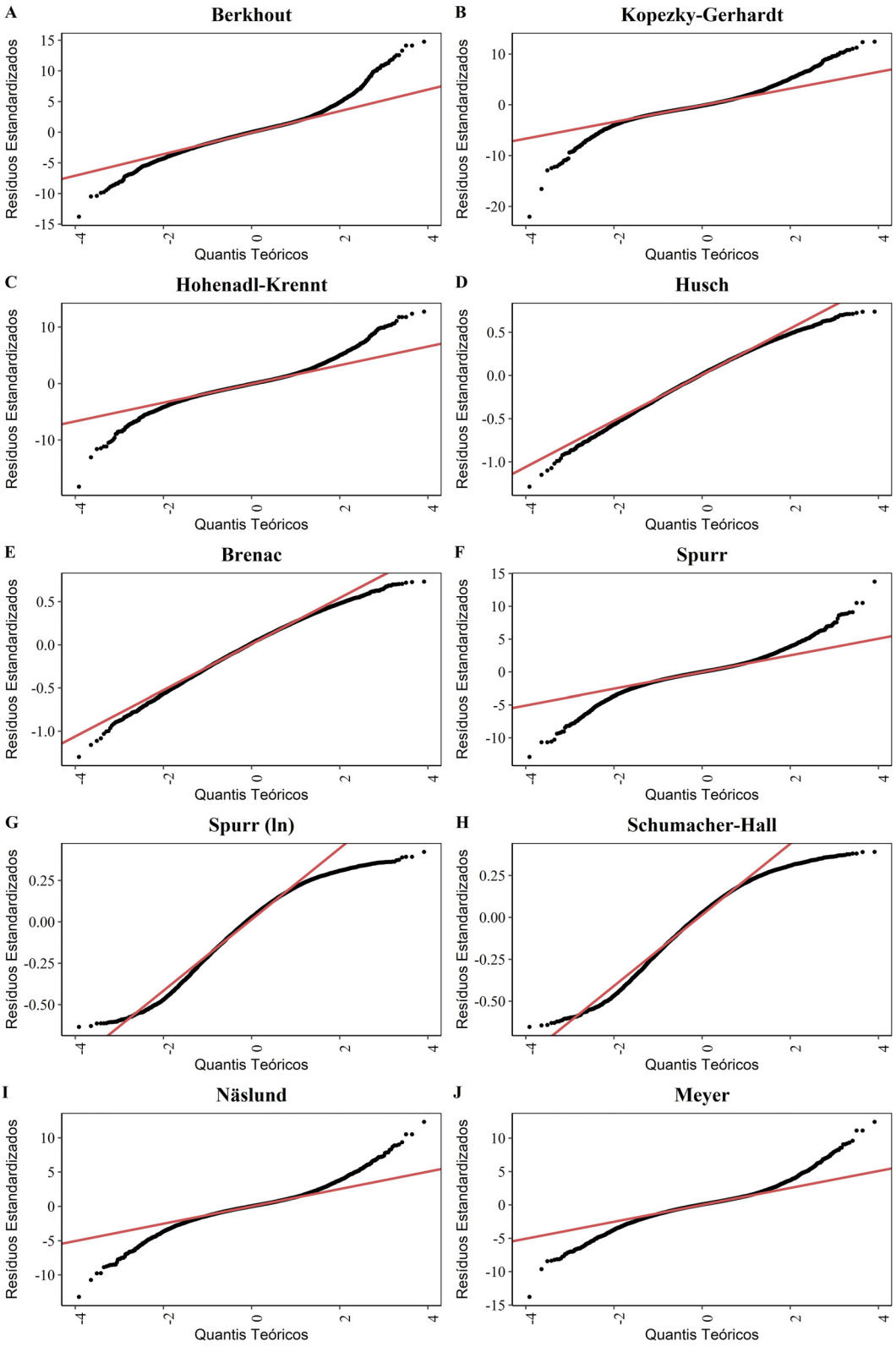
Na presença de heterocedasticidade os estimadores de MQO dos parâmetros lineares da estrutura de regressão conservam as propriedades estatísticas de "não tendenciosidade" (a média se iguala ao parâmetro verdadeiro) e "consistência" (os estimadores convergem para os parâmetros verdadeiros à medida que o tamanho da amostra aumenta indefinidamente), mas deixam de ser "eficientes" (ou melhores) mesmo assintoticamente, ou seja, não têm variância mínima na classe dos estimadores não viciados. Portanto, sob tal condição as variâncias dos estimadores de MQO são viesadas e, por conseguinte, os erros-padrões estimados para construção de intervalos de confiança e os testes de hipóteses (*t* e *F*) não são válidos (GUJARATI; PORTER, 2011).

FIGURA 36 – GRÁFICOS DE RESÍDUOS PADRONIZADOS VERSUS VALORES PREDITOS DOS MODELOS TRADICIONAIS GENÉRICOS AJUSTADOS PARA PREDIZER O VOLUME COMERCIAL DE ESPÉCIES MANEJADAS DA AMAZÔNIA BRASILEIRA.



FONTE: O autor (2020).

FIGURA 37 – GRÁFICOS QUANTIL-QUANTIL NORMAL DOS MODELOS TRADICIONAIS GENÉRICOS AJUSTADOS PARA PREDIZER O VOLUME COMERCIAL DE ESPÉCIES MANEJADAS DA AMAZÔNIA BRASILEIRA.



FONTE: O autor (2020).

Sob a hipótese de normalidade ($\epsilon_i \sim N(0, \sigma^2)$), os estimadores de MQO são MELNV (eficientes e consistentes) (GUJARATI; PORTER, 2011). As transformações logarítmicas garantiram modelos com distribuição residual empírica mais simétrica. Apesar disso, para todos os modelos ajustados, o teste de *Jarque–Bera* (α=0,05) indicou rejeição da hipótese de normalidade. Em geral, os diagramas Q-Q normal exibiram resíduos empíricos com desvios nas caudas direita e esquerda. Modelos não logarítmicos mostraram bastante quantidades de resíduos fora do intervalo $-3 \leq z_i \leq 3$, indicando a presença excessiva de pontos discrepantes.

Em termos teóricos, para pequenas amostras (n < 100) a hipótese de normalidade assume papel fundamental na derivação das distribuições de probabilidade dos estimadores de MQO, e permite o uso confiável de testes estatísticos (*t*, *F*, χ^2) usuais em MCRL. Por outro lado, se o tamanho da amostra for suficientemente grande a hipótese de normalidade pode ser relaxada (GUJARATI; PORTER, 2011). Portanto, devido o tamanho da amostra deste estudo ser considerada grande (n = 11084), é razoável admitir que a rejeição da hipótese de normalidade do termo de erro estocástico não invalidará os procedimentos inferenciais usuais, desde que a suposição de homocedasticidade não seja rejeitada. Também o teste de *Durbin-Watson* (α = 0,05) rejeitou a hipótese de autocorrelação residual para os modelos ajustados.

O desempenho dos modelos de regressão linear ajustados também foi avaliado sobre uma amostra independente do ajuste (n = 2747), mesma usada para avaliar os MAM após seleção de hiperparâmetros ótimos. Em termos gerais, os modelos tiveram adequada capacidade de generalização, compatíveis com os indicativos do erro de resubstituição (TABELA 15). Os resíduos no conjunto de teste também apresentaram padrão similar aos encontrados sobre o conjunto de treinamento (FIGURA 38).

TABELA 15 – DESEMPENHO DOS MODELOS TRADICIONAIS GENÉRICOS NO CONJUNTO DE TESTE (n = 2747).

Modelo	Bk	K-G	H-K	H	B	S	S(ln)	S-H	N	M
r	0,8856	0,8885	0,8905	0,8885	0,8905	0,9285	0,9294	0,9305	0,9303	0,9313
R_a^2	0,7840	0,7886	0,7925	0,7881	0,7928	0,8621	0,8638	0,8657	0,8653	0,8671
RSE (%)	28,96	28,64	28,38	28,68	28,36	23,14	22,99	22,83	22,87	22,71

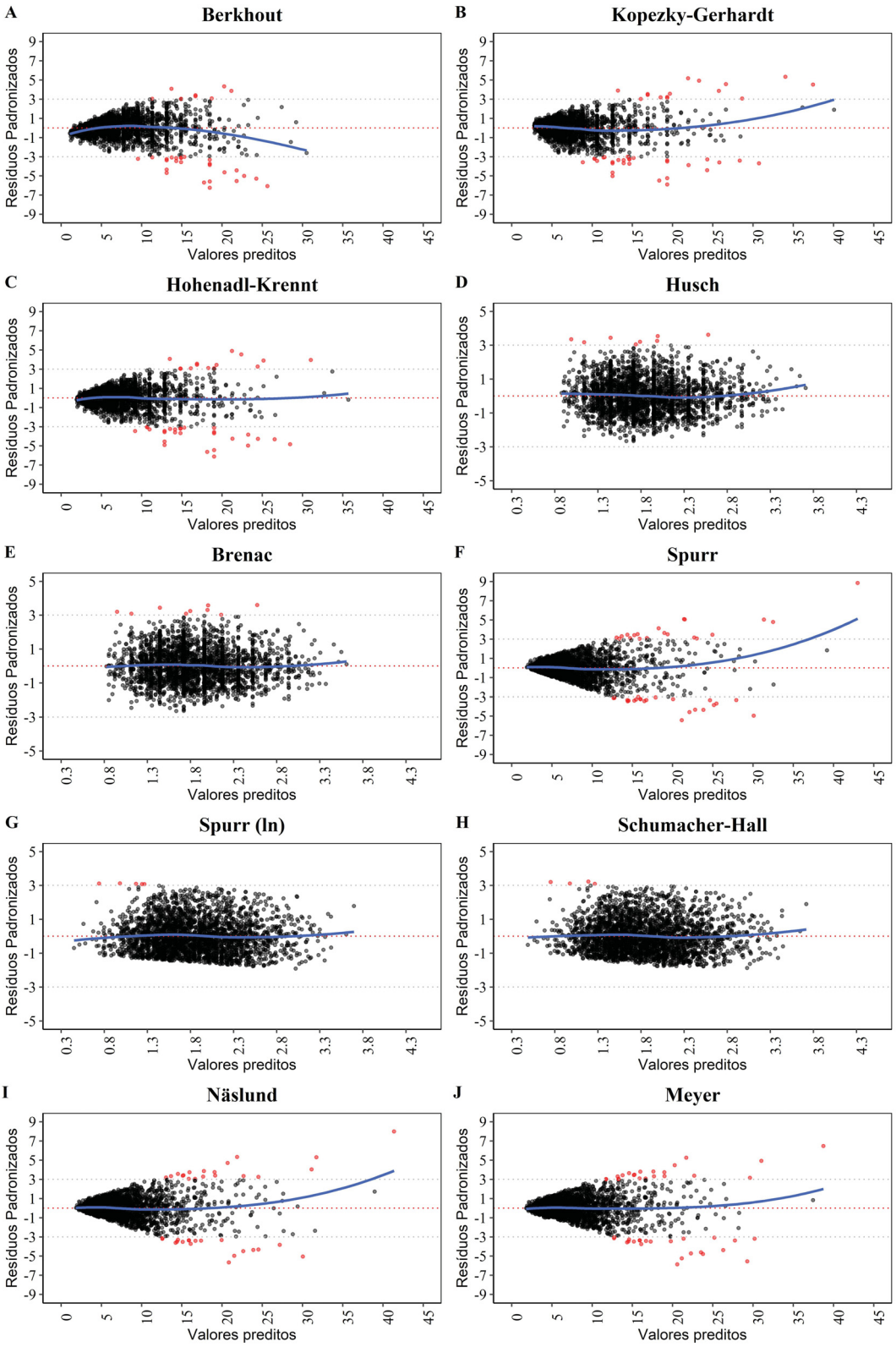
FONTE: O autor (2020).

NOTA: Valor predito foi corrigido pelo fator apresentado em Baskerville (1972). As métricas foram calculadas considerando as estimativas na escala natural da variável resposta.

LEGENDA: Bk = Berkhout; K-G = Kopezky-Gerhardt; H-K = Hohenadl-Krennt; H = Husch; B = Brenac; S = Spurr; S(ln) = Spurr (ln); S-H = Schumacher-Hall; N = Näslund; M = Meyer. r = coeficiente de correlação; R_a^2 = coeficiente de determinação ajustado; RSE = Residual Standard Error.

Em se tratando de florestas naturais inequiâneas, a escolha de modelos volumétricos tradicionais deve ponderar, para além das estatísticas de qualidade de ajuste, correlações entre preditores e atendimento às hipóteses subjacentes do MCRL (em especial, a homocedasticidade), a facilidade de coleta de dados de regressores, com rigorosa atenção à qualidade e precisão das medidas tomadas. Neste ponto, a dificuldade de determinar com precisão a altura de árvores em florestas nativas deve ser refletida. Portanto, se as medidas de altura não puderem ser tomadas com qualidade, parece ser razoável considerar o uso de modelos que incorporem apenas a variável diâmetro.

FIGURA 38 – GRÁFICOS DE RESÍDUOS PADRONIZADOS VERSUS VALORES PREDITOS PARA O CONJUNTO DE TESTE (n = 2747).



FONTE: O autor (2020).

Neste estudo, em particular, a inserção da variável altura apesar de melhorar as estatísticas de qualidade de ajuste, parece incluir um padrão sistemático aos resíduos dos

modelos de Spurr (ln) e Schumacher-Hall. Assim, é razoável admitir que provavelmente a variável altura não foi medida com precisão. Em termos práticos, isso pode ser admitido devido a variável assumir apenas valores inteiros, apesar de sua natureza contínua.

Finalmente, em razão da violação da hipótese de homocedasticidade assumiu-se que as variâncias dos estimadores de MQO são tendenciosas e, portanto, as interpretações inferenciais (testes de hipóteses e intervalos de confianças) não são confiáveis. Apesar disso, a propriedade estatística de não tendenciosidade dos estimadores de MQO é conservada. Assim, se o interesse é a estimativa pontual o modelo logarítmico de Husch, que inclui apenas a variável diâmetro, foi admitido como mais adequado, sobretudo, devido aos resíduos de MQO mais comportados.

4.2.3 Modelos de aprendizado de máquina: configuração, seleção e comparação

Na construção dos modelos de aprendizado de máquina para estimativa do volume comercial com casca foram consideradas duas estratégias de modelagem preditiva: **Abordagem 1**: usar todas as preditoras (p = 11) (ver subseção 3.3.1) como entradas no processo de construção dos modelos; **Abordagem 2**: usar apenas preditoras que continham o diâmetro incluso (p = 4) como entradas no processo de construção dos modelos. Assim, a ideia básica foi estabelecer dois espaços de características para o processo de treinamento dos modelos: um dependente apenas do diâmetro, e outro sujeito à disponibilidade da altura e diâmetro da árvore.

A abordagem de considerar variáveis dependentes do diâmetro e altura no espaço de entrada garantiu a construção de modelos de aprendizado de máquina mais acurados, com redução de aproximadamente 6% na estimativa do RMSE. Para a abordagem 1, o RMSE variou entre 1,69 (22,83%; SVR) e 1,82 (24,47%; BT) (TABELA 16). Por outro lado, os modelos construídos usando apenas variáveis dependentes do diâmetro mostraram RMSE entre 2,14 (28,77%; SVR) e 2,22 (29,91%; BT) (TABELA 17). Ainda, para ambas as abordagens, os modelos com ótimo ajuste de hiperparâmetros mostraram estimativas de desempenho bastante próximas, com diferença de apenas 1,64% e 1,14% entre o primeiro e último modelo no ranque de RMSE, para as abordagens 1 e 2, respectivamente.

Em termos gerais, o emprego conjunto dos métodos “Center and Scale” e “BoxCox” mostraram-se mais adequados para o pré-processamento das preditoras disponíveis. Outras técnicas como “spatialSign” e “YeoJohnson” não foram adequadas. Para os algoritmos RT, M5’ e Floresta Aleatória nenhum pré-processamento foi realizado. Os CVs das métricas RMSE e MAE na reamostragem foram inferiores a 5% para todos os modelos de aprendizado de máquina, evidenciando a estabilidade dos modelos para predizer a resposta média para amostras futuras. Para alguns modelos, medidas de desempenho discrepantes (Q1 – 1.5*IQR e Q3 + 1,5*IQR) foram identificadas (FIGURA 39).

TABELA 16 – CONFIGURAÇÃO ÓTIMA DE HIPERPARÂMETROS E ESTIMATIVA DE DESEMPENHO MÉDIO USANDO TODAS AS PREDITORAS (p = 11) COMO ENTRADAS NO PROCESSO DE CONSTRUÇÃO DOS MODELOS.

Modelo	**Pré-Processamento**	**Hiperparâmetro**	**Conjunto de validação (folds = 10)**					
			$RMSE$	$rRMSE$	r	R^2	MAE	$Bias(\%)$
SVR (Radial)	Center and Scale; BoxCox	**sigma** = 0,003 **C** = 128	1,6965 (0,0639)	22,83 (0,86)	0,9315 (0,0067)	0,8678 (0,0125)	1,1995 (0,0266)	0,2253 (0,6868)
ANN	Center and Scale; BoxCox	**size** = 7 **decay** = 0,003	1,7027 (0,0636)	22,91 (0,84)	0,9318 (0,0069)	0,8681 (0,0128)	1,2061 (0,0293)	-1,9750 (0,6678)
M5'	-	**pruned** = Yes **smoothed** = No	1,7101 (0,0651)	23,01 (0,87)	0.9311 (0,0071)	0,8669 (0,0132)	1,2119 (0,0284)	-1,9855 (0,7034)
XGBoost	Center and Scale; BoxCox	**rounds** = 550 **eta** = 0,05 **max_depth** = 1 **colsample_bytree** = 1 **min_child_weight** = 30 **subsample** = 0,9	1,7107 (0,0572)	23,02 (0,79)	0,9309 (0,0066)	0,8667 (0,0124)	1,2076 (0,0304)	-1,9799 (0,5682)
SGB	Center and Scale; BoxCox	**interaction.depth** = 1 **shrinkage** = 0,021 **n.trees** = 700 **n.minobsinnode** = 5	1,7119 (0,0623)	23,04 (0,84)	0,9309 (0,0066)	0,8668 (0,0123)	1,2069 (0,0311)	-2,0516 (0,5839)
*wk*NN	Center and Scale; BoxCox	**k** = 24 **kernel** = rectangular **d** = 1	1,7631 (0,0655)	23,73 (0,88)	0,9266 (0,0072)	0,8586 (0,0133)	1,2313 (0,0346)	-2,1041 (0,5807)
RF	-	**mtry** = 1 **ntree** = 100	1,7672 (0,0604)	23,78 (0,86)	0,9259 (0,0076)	0,8575 (0,0140)	1,2306 (0,0315)	-2,003 (0,6134)
RT	-	**cp** = 0,0003	1.7675 (0,0554)	23,79 (0,75)	0,9259 (0,0066)	0,8574 (0,0123)	1,2337 (0,0294)	-1,9583 (0,6301)
BT	-	**nbagg** = 1000	1,8179 (0,0623)	24,47 (0,8872)	0,9212 (0,0084)	0,8488 (0,0154)	1,2599 (0,0357)	-1,4867 (0,6195)

FONTE: O autor (2020).

NOTA: Todas as métricas foram calculadas na escala da variável resposta do modelo de aprendizado. *As árvores foram crescidas usando a configuração: rpart.control(minsplit = 2, cp = 0).

LEGENDA: ANN = Artificial Neural Networks (rede MLP); SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); $RMSE$ = Root Mean Square Error; $rRMSE$ = Relative Root Mean Square Error; r = Coeficiente de correlação de Pearson; R^2 = Coeficiente de determinação; MAE = Mean Absolute Error; Entre parênteses está o desvio padrão das métricas na reamostragem.

TABELA 17 – CONFIGURAÇÃO ÓTIMA DE HIPERPARÂMETROS E ESTIMATIVA DE DESEMPENHO MÉDIO USANDO APENAS PREDITORAS QUE CONTINHAM O DIÂMETRO INCLUSO (p = 4) COMO ENTRADAS NO PROCESSO DE CONSTRUÇÃO DOS MODELOS.

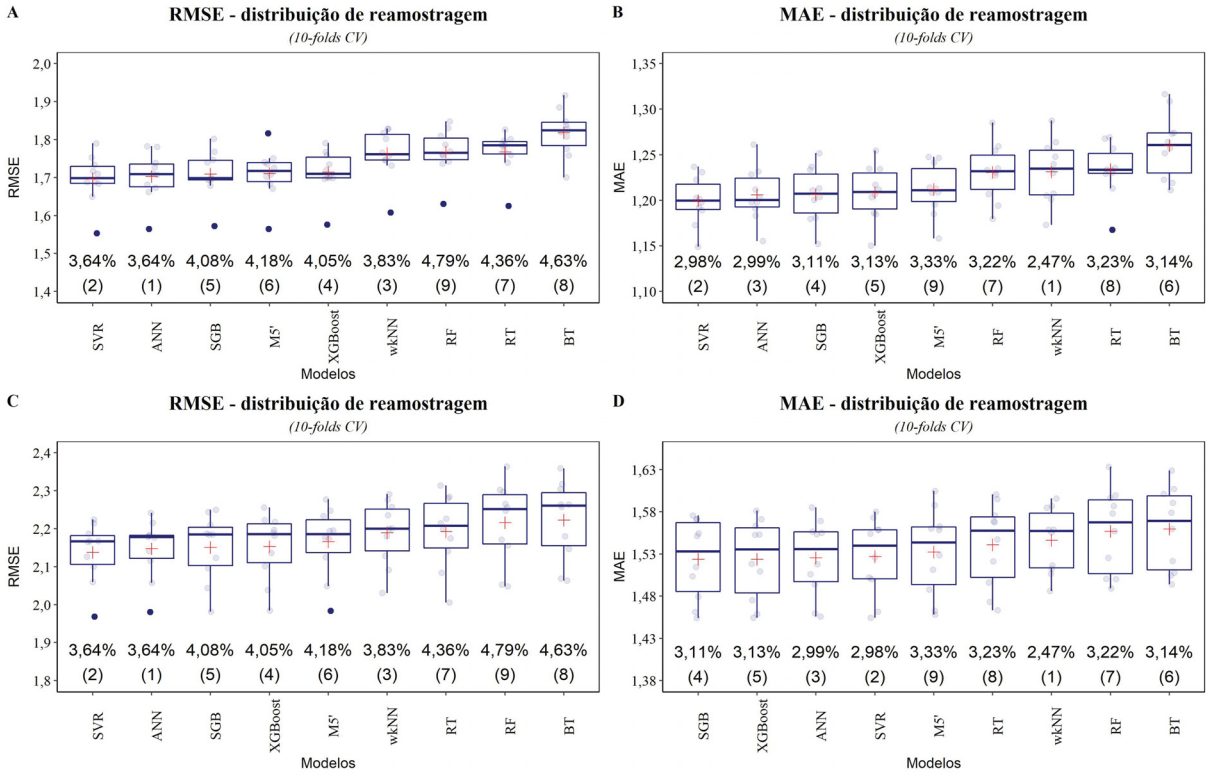
Modelo	**Pré-Processamento**	**Hiperparâmetro**	**Conjunto de validação (folds = 10)**					
			$RMSE$	$rRMSE$	r	R^2	MAE	$Bias(\%)$
SVR (Radial)	Center and Scale; BoxCox	**sigma** = 0,001 **C** = 128	2,1379 (0,0779)	28,77 (0,99)	0,8894 (0,0087)	0,7911 (0,0154)	1,5269 (0,0454)	-1,9132 (0,5118)
ANN	Center and Scale; BoxCox	**size** = 11 **decay** = 0,002	2,1472 (0,0781)	28,89 (0,99)	0,8897 (0,0087)	0,7916 (0,0154)	1,5253 (0,0456)	-3,3164 (0,5347)
SGB	Center and Scale; BoxCox	**interaction.depth** = 2 **shrinkage** = 0,091 **n.trees** = 200 **n.minobsinnode** = 20	2,1504 (0,0878)	28,94 (1,16)	0,8891 (0,0099)	0,7905 (0,0175)	1,5238 (0,0473)	-3,2921 (0,5041)
XGBoost	Center and Scale; BoxCox	**rounds** = 800 **eta** = 0,02 **max_depth** = 1 **colsample_bytree** = 0,7 **min_child_weight** = 40 **subsample** = 0,8	2,1525 (0,0871)	28,97 (1,13)	0,8889 (0,0094)	0,7903 (0,0166)	1,5238 (0,0477)	-3,2967 (0,5029)
M5'	-	**pruned** = Yes **smoothed** = No	2,1656 (0,0906)	29,14 (1,13)	0,8882 (0,0085)	0,7889 (0,0151)	1,5323 (0,0511)	-3,3885 (0,5160)
*wk*NN	Center and Scale; BoxCox	**k** = 24 **kernel** = rank **d** = 3	2,1892 (0,0839)	29,46 (1,11)	0,8847 (0,0098)	0,7828 (0,0173)	1,5463 (0,0383)	-3,3426 (0,4531)
RT	-	**cp** = 0.00048	2,1925 (0,0956)	29,50 (1,21)	0,8846 (0,0087)	0,7827 (0,0153)	1,5409 (0,0497)	-3,3876 (0,4992)
RF	-	**mtry** = 2 **ntree** = 450	2,2161 (0,1062)	29,82 (1,40)	0,8811 (0,0116)	0,7765 (0,0205)	1,5567 (0,0501)	-3,1176 (0,5944)
BT	Center and Scale; BoxCox	**nbagg** = 50	2,2225 (0,1029)	29,91 (1,3578)	0,8804 (0,0114)	0,7752 (0,0202)	1,5595 (0,0490)	-3,0643 (0,6149)

FONTE: O autor (2020).

NOTA: Todas as métricas foram calculadas na escala da variável resposta do modelo de aprendizado. *As árvores foram crescidas usando a configuração: rpart.control(minsplit = 2, cp = 0).

LEGENDA: ANN = Artificial Neural Networks (rede MLP); SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); $RMSE$ = Root Mean Square Error; $rRMSE$ = Relative Root Mean Square Error; r = Coeficiente de correlação de Pearson; R^2 = Coeficiente de determinação; MAE = Mean Absolute Error; Entre parênteses está o desvio padrão das métricas na reamostragem.

FIGURA 39 – DISTRIBUIÇÃO DAS ESTIMATIVAS DE DESEMPENHO (RMSE E MAE) NA VALIDAÇÃO CRUZADA PARA OS MODELOS DE CONFIGURAÇÃO ÓTIMA.

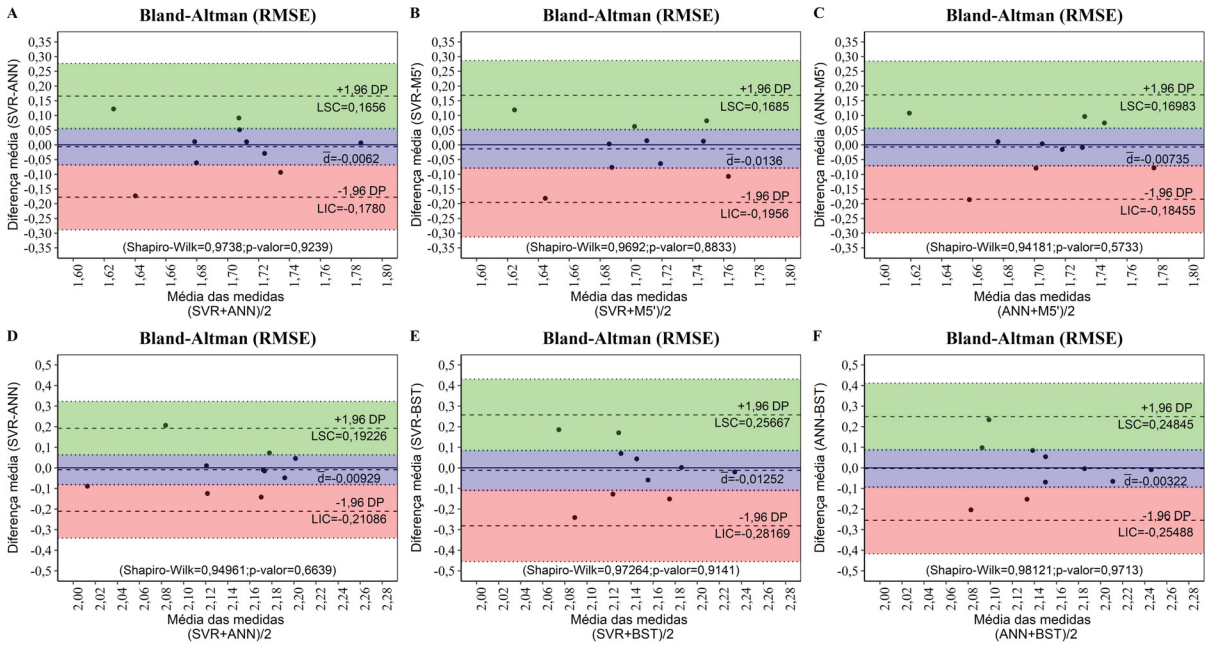


FONTE: O autor (2020).

LEGENDA: Barras na vertical (cor azul) representam: Q1 – 1.5*IQR (1º quartil menos 1,5 vezes o intervalo interquartil) e Q3 + 1,5*IQR (3º quartil mais 1,5 vezes o intervalo interquartil); Cruz em vermelho representa a média de desempenho dos modelos na reamostragem. O coeficiente de variação (CV) na reamostragem está expresso abaixo das caixas boxplot. **A** e **B** = modelos construídos usando todas as preditoras no espaço de recursos; **C** e **D** = modelos construídos usando apenas preditoras dependentes do diâmetro da árvore no espaço de recursos. ANN = Artificial Neural Networks (rede MLP); SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); RMSE = Root Mean Square Error.

O gráfico de Bland-Altman foi usado para avaliar o grau de concordância entre as estimativas de desempenho (RMSE) de modelos de aprendizado de máquina. Para cada abordagem, o método foi implementado apenas para os três melhores modelos indicados na reamostragem. Para todas as comparações, as estimativas de viés não foram consideradas significativas, e a linha de igualdade (zero) esteve dentro dos intervalos de confiança da diferença média. Além disso, a maioria das diferenças entre modelos estiveram dentro dos limites de concordância ($\bar{d} \pm 1,96 * DP$) e, portanto, a suposição de normalidade das distribuições das diferenças foi admitida através do teste *Shapiro-Wilk* (α = 0,05) (FIGURA 40). Portanto, existem evidências para admitir que os modelos SVR, ANN e M5' (abordagem 1) e SVR, ANN e BST (abordagem 2) possuem desempenho médio concordante e, portanto, é razoável admitir uma precisão similar dos modelos para predição de amostras futuras.

FIGURA 40 – ANÁLISE DE CONCORDÂNCIA DE BLAND-ALTMAN BASEADO NA MEDIDA RMSE ENTRE OS TRÊS MELHORES MODELOS INDICADOS NA VALIDAÇÃO CRUZADA.



FONTE: O autor (2020).

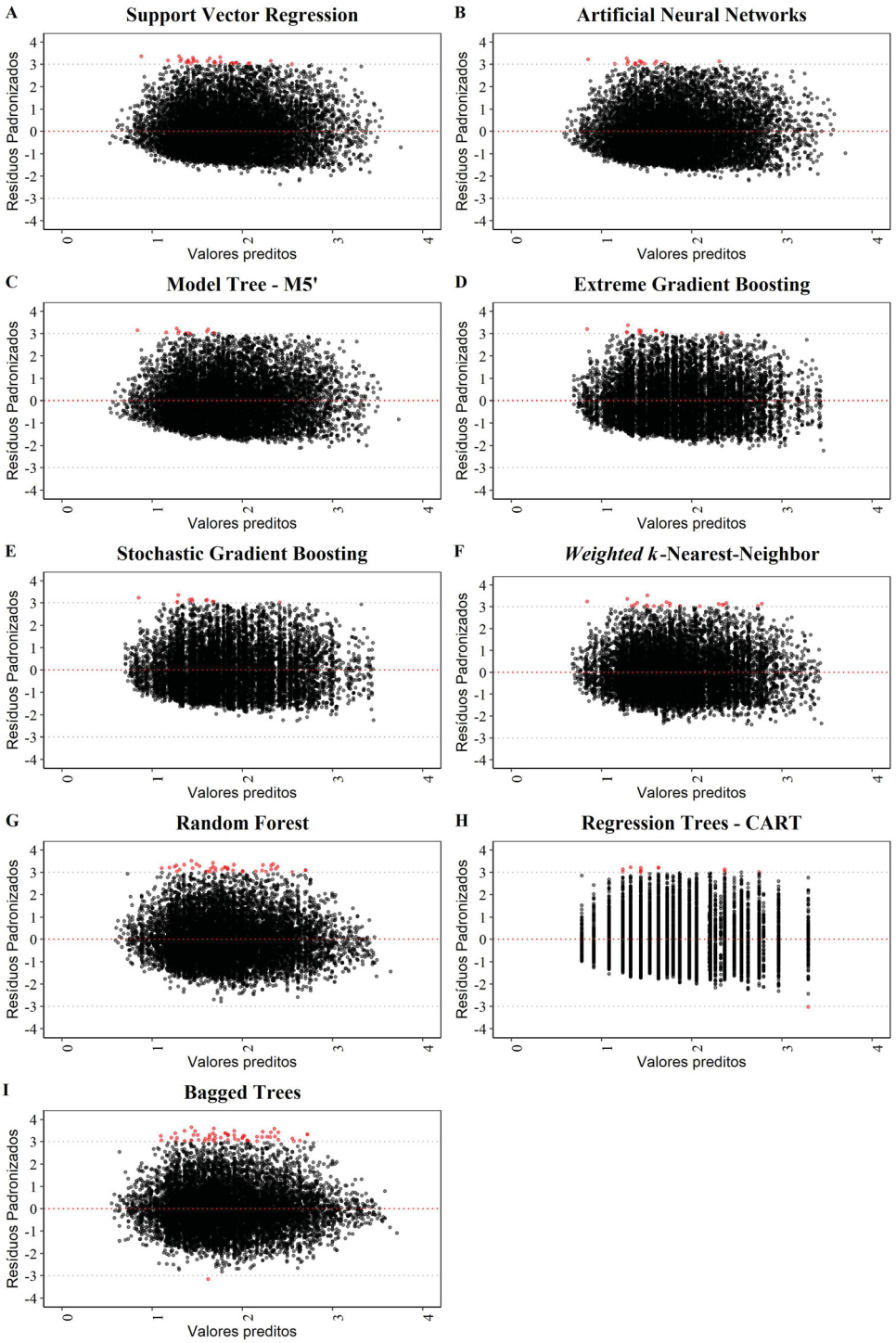
NOTA: O gráfico de Bland-Altman foi elaborado com auxílio do pacote “blandr” (DATTA, 2017)

LEGENDA: **A**, **B** e **C** = modelos construídos usando todas as preditoras no espaço de recursos; **D**, **E** e **F** = modelos construídos usando apenas preditoras dependentes do diâmetro da árvore no espaço de recursos. $\bar{d}$ = diferença média entre dois modelos (linha pontilhada horizontal central); DP = desvio padrão; LIC = limite de concordância inferior; LSC = limite de concordância superior; regiões hachuradas = intervalo de confiança de 95% para os LIC (vermelha), LSC (verde) e diferença média (azul). ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; M5’ = Model Tree. RMSE = Root Mean Square Error; MAE = Mean Absolute Error.

Em relação à distribuição residual, apenas poucos pontos amostrais estiveram situados fora do intervalo $-3 \leq z_i \leq 3$, seja para os modelos de configuração ótima construídos usando um espaço de recurso dependente das variáveis altura e diâmetro (FIGURA 41), seja para os modelos aprendidos usando um espaço de características dependente apenas do diâmetro (FIGURA 42).

Na abordagem 1, os modelos construídos apresentaram resíduos não uniformemente distribuídos em relação à média zero, com indícios de heterocedasticidade residual. Apesar disso, a heterocedasticidade nos modelos de aprendizado de máquina parece ser menos severa do que aquela constatada para os modelos de Spurr (ln) (FIGURA 36G) e Schumacher-Hall (FIGURA 36H). Na abordagem 2, os modelos mostraram resíduos mais comportados em comparação à abordagem 1, e também com padrão similar aos resíduos de MQO dos modelos logarítmicos de simples entrada (FIGURA 36D - Husch e FIGURA 36E - Brenac).

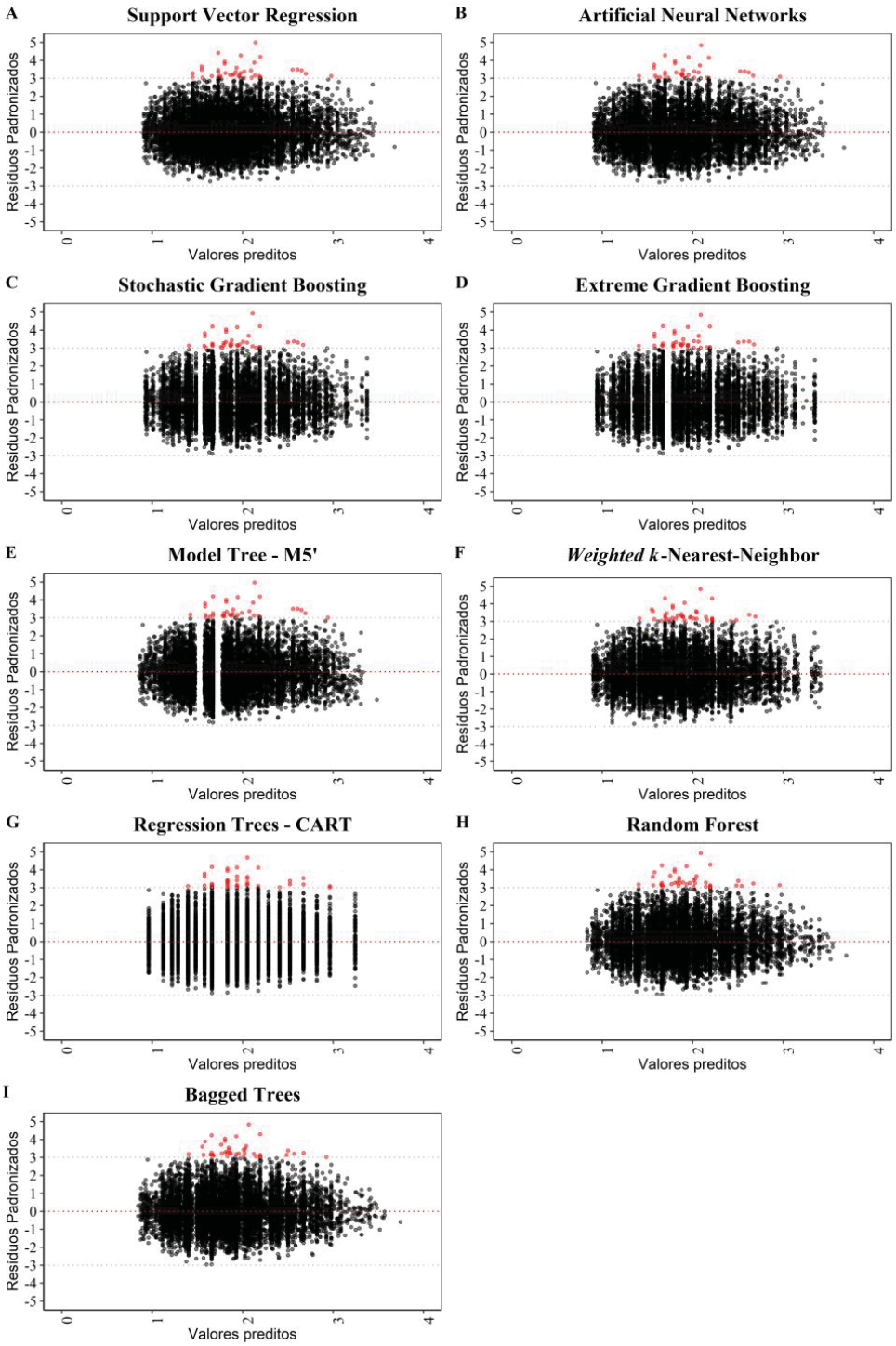
FIGURA 41 – RESÍDUOS PADRONIZADOS NO CONJUNTO DE TREINAMENTO PARA OS MODELOS DE CONFIGURAÇÃO ÓTIMA APRENDIDOS USANDO UM ESPAÇO DE RECURSOS DEPENDENTE DO DIÂMETRO E ALTURA DA ÁRVORE.



FONTE: O autor (2020).

NOTA: Pontos em vermelho indicam resíduos com valores fora do intervalo $-3 \leq z_i \leq 3$.

FIGURA 42 – RESÍDUOS PADRONIZADOS NO CONJUNTO DE TREINAMENTO PARA OS MODELOS DE CONFIGURAÇÃO ÓTIMA APRENDIDOS USANDO UM ESPAÇO DE RECURSOS DEPENDENTE APENAS DO DIÂMETRO DA ÁRVORE.



FONTE: O autor (2020).

NOTA: Pontos em vermelho indicam resíduos com valores fora do intervalo $-3 \leq z_i \leq 3$.

4.2.4 Interpretando modelos de aprendizado de máquina: importância e relação de variáveis

O uso de técnicas de aprendizado de máquina têm aumentado progressivamente em diversos campos de conhecimento, devido a alta capacidade de modelar problemas complexos. Porém, os modelos aprendidos são considerados “caixa preta” (do inglês, *Black-Box*), em razão da dificuldade de interpretabilidade da função modelada. Assim, a proposta da seção corrente é empregar abordagens que permitam obter algum nível de interpretabilidade para os modelos de aprendizado de máquina de melhor desempenho. Em particular, duas abordagens foram empregadas: 1) gráficos de importância de variáveis preditoras; e 2) gráficos de dependência parcial (do inglês, *Partial Dependence Plots* - PDP) (FRIEDMAN, 2001).

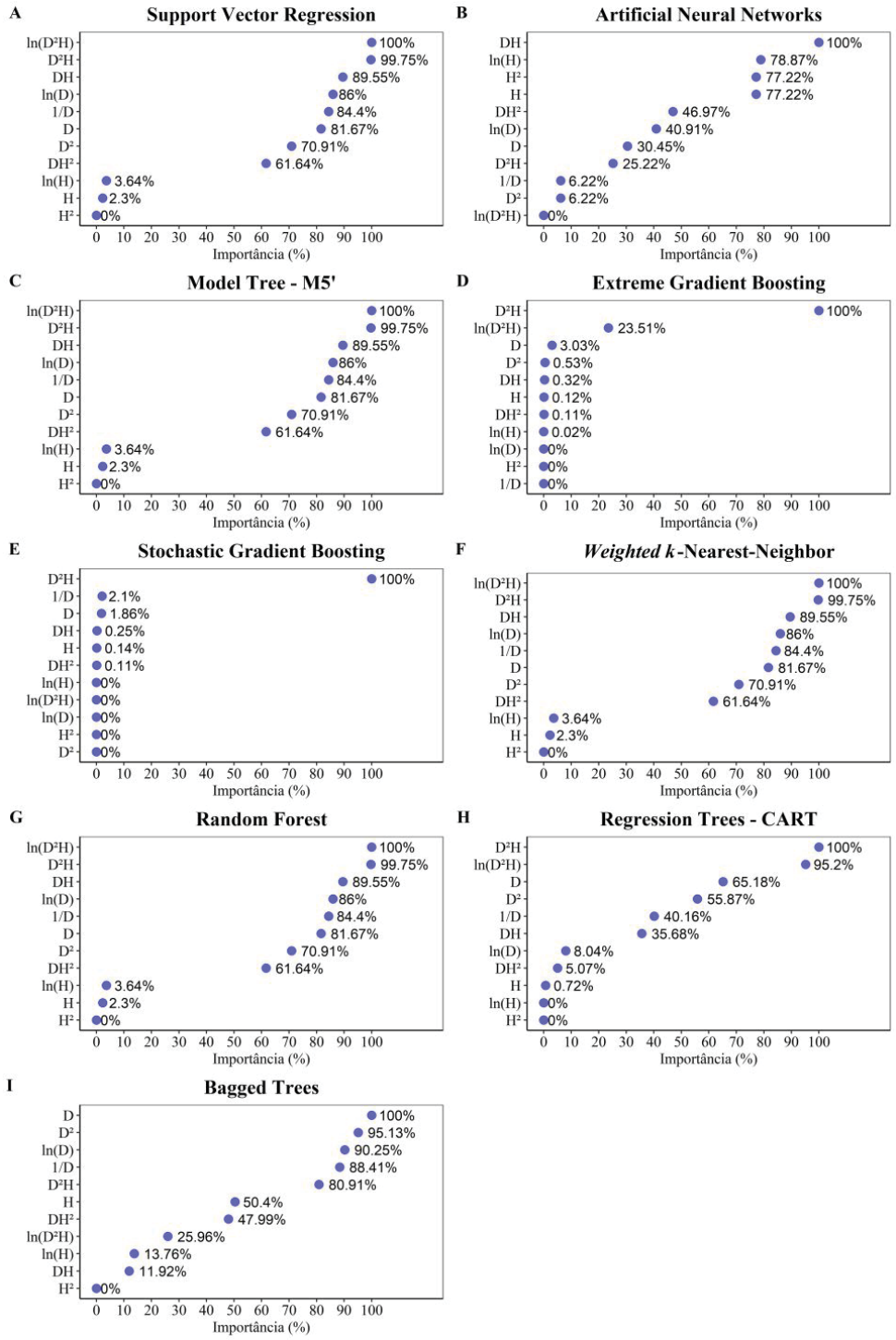
Na abordagem 1, quando usado um espaço de recursos dependente de diâmetro e altura da árvore (FIGURA 43), as preditoras de interação ln(D^2H) e D^2H apresentaram maior estimativa de importância relativa marginal com a resposta, na maioria dos casos. Apesar disso, a variável ln(D^2H) apresentou menor importância marginal para o modelo ANN (rede MLP), e também pouca influência para os modelos SGB e BT. Ainda, o diâmetro da árvore (D) e suas transformações (D^2, ln(D), 1/D) foram indicados como mais importantes para o modelo BT.

Na abordagem 2, quando usado um espaço de recursos dependente apenas do diâmetro da árvore (FIGURA 44), o logaritmo do diâmetro da árvore apresentou maior importância para os modelos SVR, ANN, M5’, wkNN e RF. Porém, a variável 1/D mostrou maior importância marginal para os modelos SGB, RT e BT, e o diâmetro (D) foi mais importante para o modelo XGBoost.

Gráficos PDP bidimensionais foram construídos para examinar a dependência entre a resposta média e os valores de preditoras mais influentes (limiar = 90%) para os quatro melhores modelos indicados na validação cruzada (FIGURA 45). Em cada trama PDP foram inseridos subgráficos bidimensionais (resposta versus preditor), com uma linha tendência (suavização) para a relação empírica no conjunto de treinamento. Na Figura 45, **A**, **B** e **C**: gráficos PDP para a abordagem 1; e **D**, **E** e **F**: gráficos PDP para a abordagem 2.

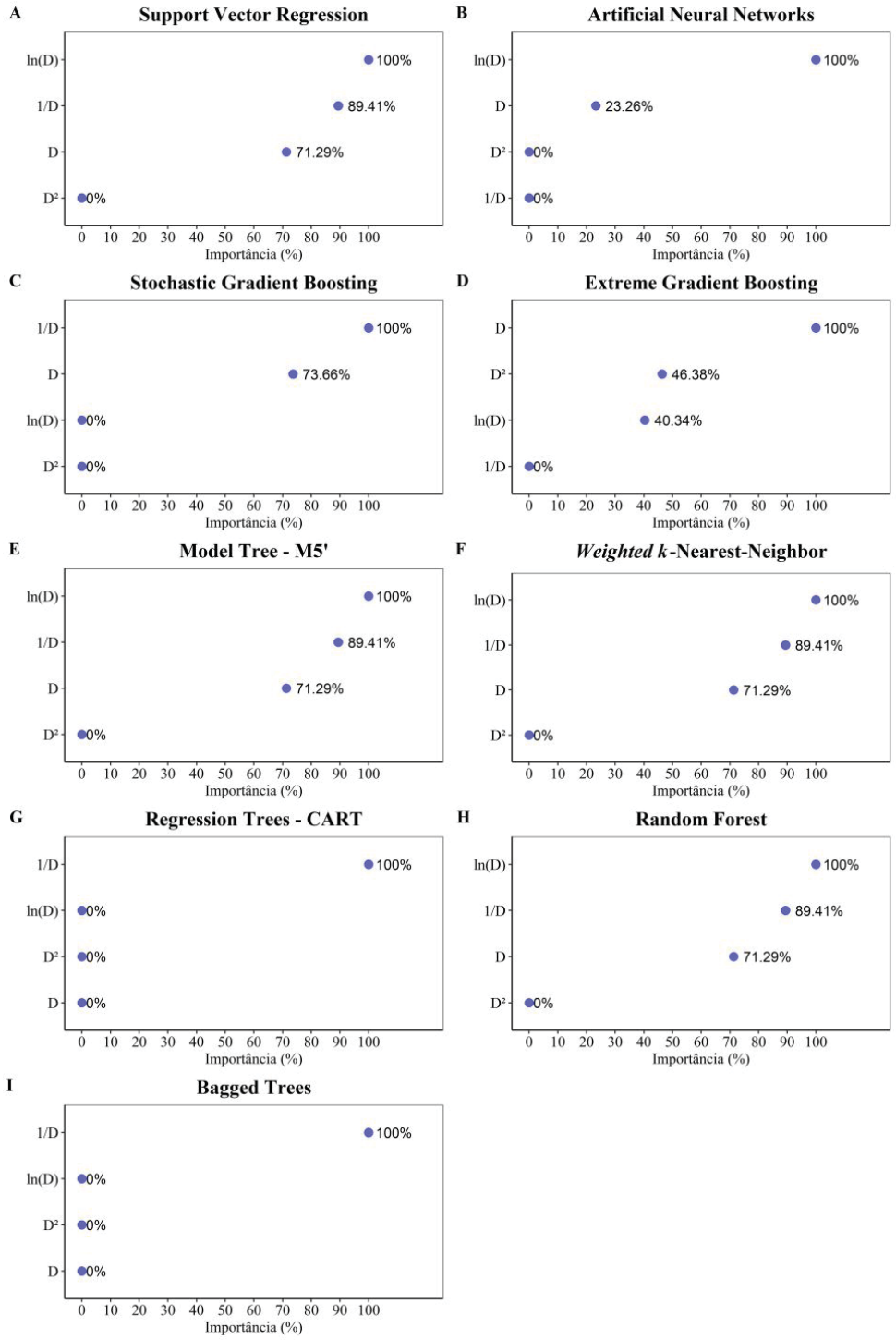
Em termos gerais, os modelos parecem aprender melhor a tendência da relação empírica marginal entre variável resposta e um preditor (ver subgráficos), em especial, para as suas respectivas variáveis mais importantes. Por exemplo, na abordagem 1, para os modelos XGBoost e ANN, os perfis PDP para D^2H e DH, respectivamente, refletem uma relação não-linear e crescente com a resposta média, ou seja, há uma tendência de aumento da resposta média à medida que os valores das preditoras crescem.

FIGURA 43 – IMPORTÂNCIA RELATIVA DE PREDITORAS PARA OS MODELOS DE CONFIGURAÇÃO ÓTIMA APRENDIDOS USANDO UM ESPAÇO DE RECURSOS DEPENDENTE DO DIÂMETRO E ALTURA DA ÁRVORE (p = 11).



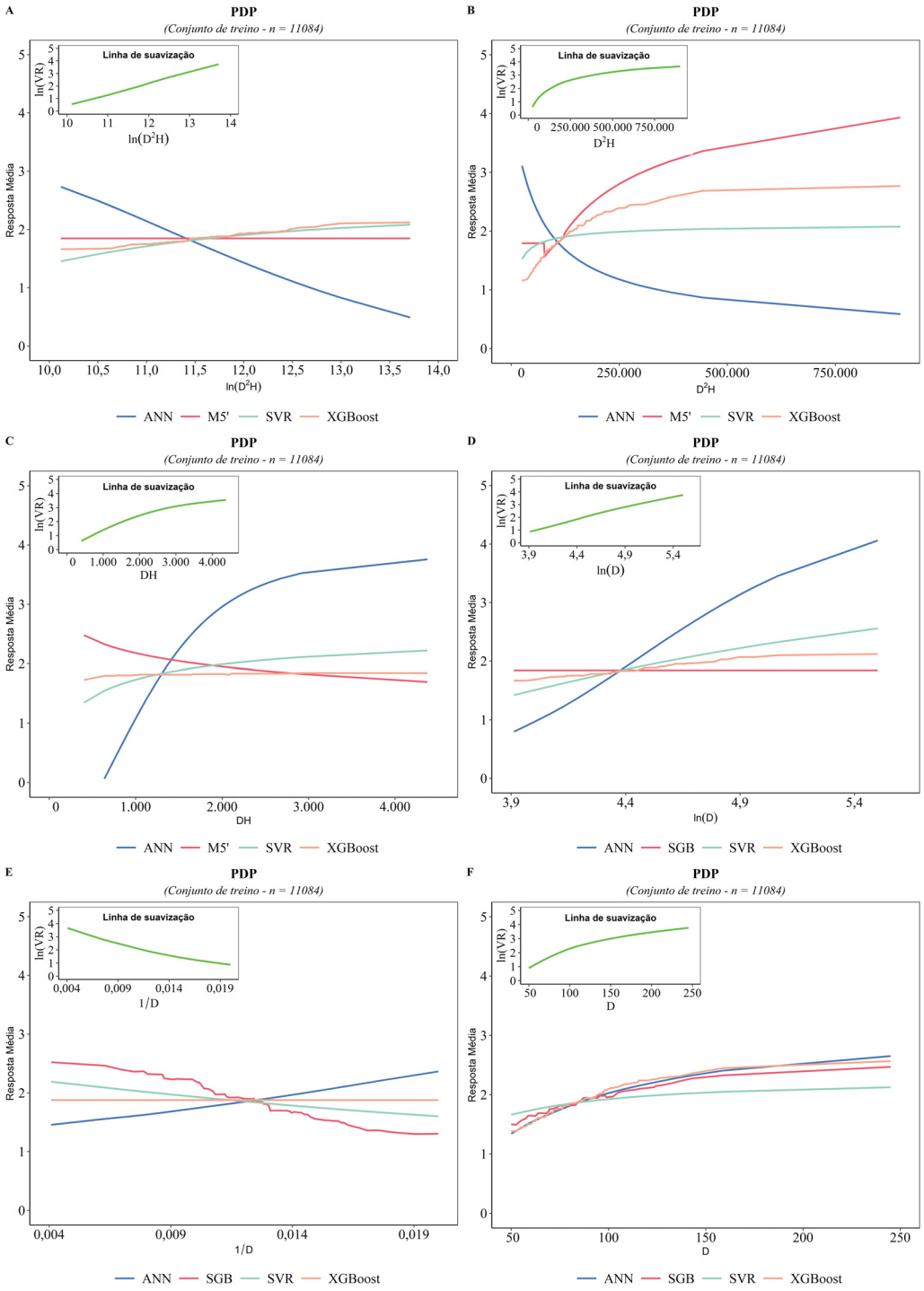
FONTE: O autor (2020).
NOTA: Elaborado com auxílio do pacote CARET (KUHN *et al.*, 2016).
LEGENDA: D = diâmetro a 1,30m do solo (cm); H = altura total (m); ln = logaritmo neperiano.

FIGURA 44 – IMPORTÂNCIA RELATIVA DE PREDITORAS PARA OS MODELOS DE CONFIGURAÇÃO ÓTIMA APRENDIDOS USANDO UM ESPAÇO DE RECURSOS DEPENDENTE APENAS DO DIÂMETRO DA ÁRVORE (p = 4).



FONTE: O autor (2020).
NOTA: Elaborado com auxílio do pacote CARET (KUHN *et al.*, 2016).
LEGENDA: D = diâmetro a 1,30m do solo (cm); H = altura total (m); ln = logaritmo neperiano.

FIGURA 45 – DEPENDÊNCIA PARCIAL ENTRE A RESPOSTA MÉDIA E PREDITORAS MAIS INFLUENTES PARA OS QUATRO MELHORES MODELOS DE APRENDIZADO DE MÁQUINA INDICADOS NA VALIDAÇÃO CRUZADA.



FONTE: O autor (2020).

NOTA: Elaborado com auxílio do pacote DALEX (BIECEK, 2018).

LEGENDA: PDP = Partial Dependence Plots; D = diâmetro a 1,30m do solo (cm); H = altura total (m); ln = logaritmo neperiano. ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree.

4.2.5 Comparação de abordagens: modelagem tradicional versus aprendizado de máquina

Na subseção 4.2.2 os modelos de regressão linear ajustados foram discutidos à luz das medidas de qualidade de ajustamento, indicativo de multicolinearidade problemática e violação das pressuposições do Modelo Clássico de Regressão Linear (MCRL). Na subseção corrente o interesse foi avaliar a qualidade da estimação pontual dos modelos de aprendizado de máquina frente à modelos volumétricos tradicionais genéricos, através da comparação da distribuição residual. Uma síntese comparativa da estimativa de desempenho, no conjunto de teste, dos melhores modelos de regressão linear clássica (simples e dupla entrada) e aprendizado de máquina para diferentes configurações do espaço de recursos está disponível (TABELA 18).

TABELA 18 – SÍNTESE DO DESEMPENHO DOS MELHORES MODELOS DE REGRESSÃO LINEAR CLÁSSICA E APRENDIZADO DE MÁQUINA PARA DIFERENTES CONFIGURAÇÕES DO ESPAÇO DE RECURSOS.

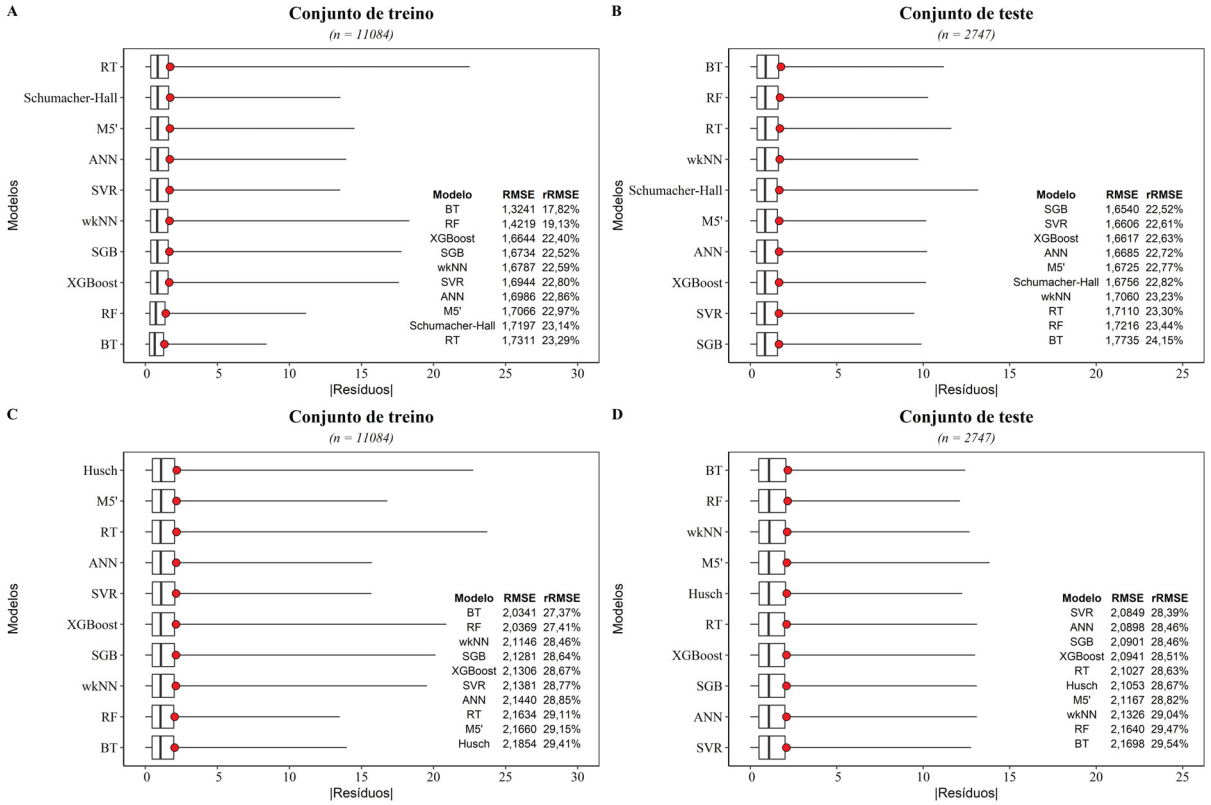
Espaço de recurso	**Modelo**	**Conjunto de teste (n = 2.747)**		
		r	R_a^2	$RMSE(\%)$
Dependente de D	SVR (Radial)	0,8910	0,7939	28,39
	Husch	0,8885	0,7881	28,68
Dependente D e H	SVR (Radial)	0,9319	0,8684	22,61
	Schumacher-Hall	0,9305	0,8657	22,82

FONTE: O autor (2020).

Os resíduos absolutos e as estimativas de desempenho dos MAM e modelos tradicionais foram comparados nos conjuntos de treinamento e teste (FIGURA 46). Os modelos de Husch (simples entrada) e Schumacher-Hall (dupla entrada), considerados de melhor desempenho na predição pontual, foram escolhidos para comparação com MAM. Assim, o modelo de Schumacher-Hall foi comparado com MAM aprendidos usando um espaço de recursos dependente do diâmetro e altura da árvore (abordagem 1), enquanto que Husch foi confrontado com MAM construídos usando um espaço de características dependente apenas do diâmetro da árvore (abordagem 2).

No conjunto de teste, em geral, os modelos de aprendizado de máquina (abordagem 1 e 2) e modelos tradicionais (Schumacher-Hall e Husch) apresentaram semelhança na distribuição residual e na estimativa de desempenho médio para predizer a variável resposta em amostras futuras. Avaliar o desempenho dos modelos de aprendizado de máquina sobre o conjunto de treinamento não é adequado, pois as estimativas podem ser excessivamente otimistas, como para BT e RF. Assim, a avaliação sobre o conjunto independente do ajuste é mais realista e compatível com os indicativos da validação cruzada. Do mesmo modo, o comportamento preditivo semelhante dos modelos foi ratificado por meio da análise do gráfico da função de distribuição cumulativa empírica inversa (do inglês, *Reversed Empirical Cumulative Distribution Function* - RECDF) para os resíduos absolutos (FIGURA 47).

FIGURA 46 – COMPARAÇÃO DE RESÍDUOS ABSOLUTOS NOS CONJUNTOS DE TREINO E TESTE PARA OS MODELOS DE APRENDIZADO DE MÁQUINA E MODELO CLÁSSICO DE REGRESSÃO LINEAR.

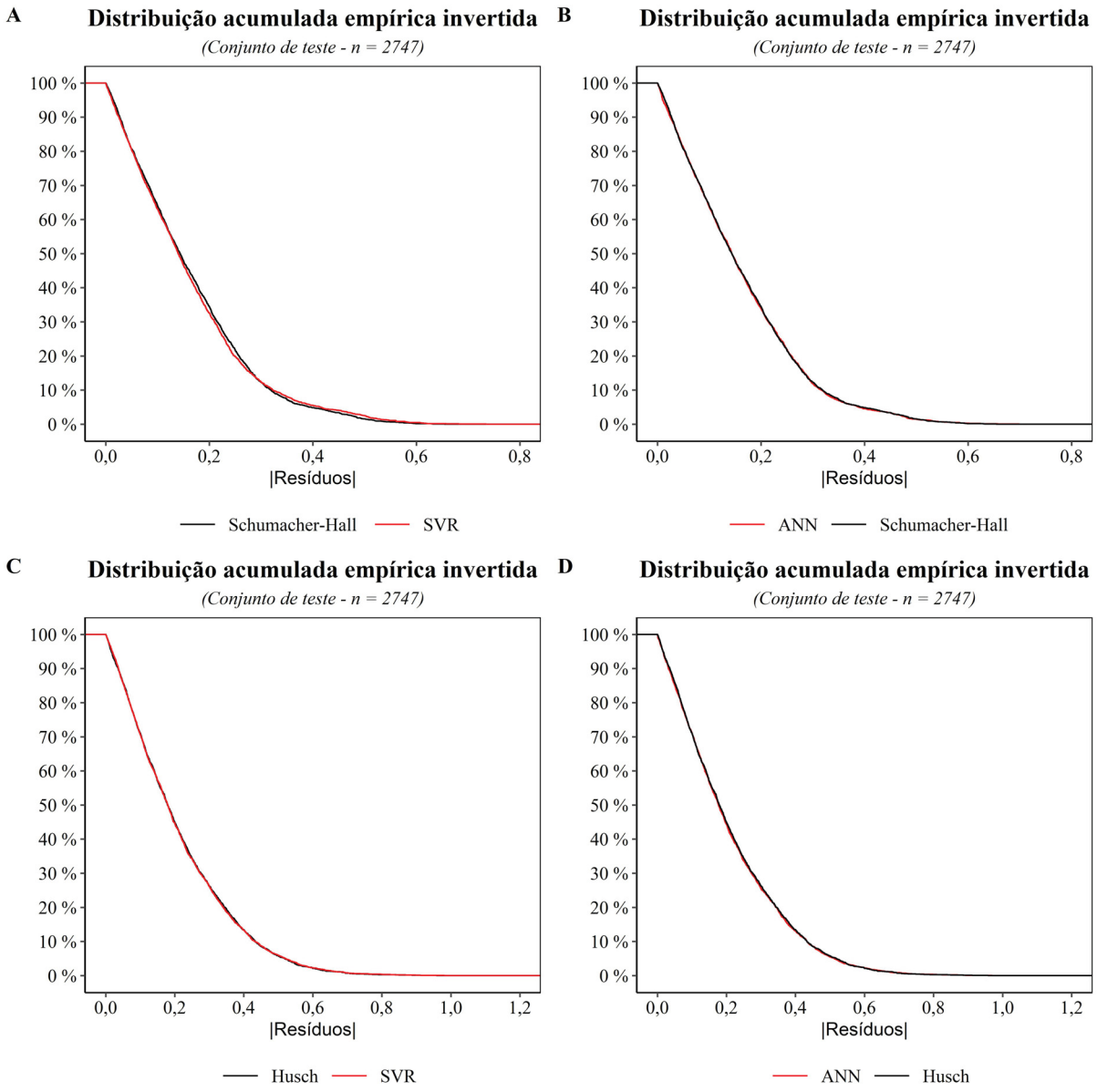


FONTE: O autor (2020).

NOTA: Elaborado com auxílio do pacote DALEX (BIECEK, 2018). Para os modelos estimados através de MQO foi calculado RSE (absoluto e %) corrigido pelo fator de Baskerville.

LEGENDA: ANN = Artificial Neural Networks; SGB = Stochastic Gradient Boosting; SVR = Support Vector Regression; XGBoost = Extreme Gradient Boosting; M5' = Model Tree; *wk*NN = *Weighted k*-Nearest-Neighbor ; RF = Random Forest; BT = Bagged Trees; RT = Regression Trees (CART); RMSE = Root Mean Square Error; rRMSE = Relative Root Mean Square Error. O ponto em vermelho representa o RMSE.

FIGURA 47 – COMPARAÇÃO DA DISTRIBUIÇÃO CUMULATIVA EMPÍRICA INVERTIDA PARA OS RESÍDUOS ABSOLUTOS ENTRE QUATRO MODELOS DE APRENDIZADO DE MÁQUINA E O MODELO ALTERNATIVO DE REGRESSÃO LINEAR



FONTE: O autor (2020).
NOTA: Elaborado com auxílio do pacote DALEX (BIECEK, 2018).
LEGENDA: ANN = Artificial Neural Networks; SVR = Support Vector Regression.

4.2.6 **MLMVol** - Aplicação na web com modelos de aprendizado de máquina para predição do volume comercial com casca de espécies manejadas na Amazônia brasileira

MLMVol é uma aplicação web com modelos de aprendizado de máquina para predição do volume comercial com casca de espécies manejadas na Amazônia brasileira desenvolvida usando o *framework* Shiny. A aplicação web foi desenvolvida para disponibilizar de modo fácil e prático os três modelos mais acurados encontrados para estimar o volume comercial obtido em romaneio: Modelo SVR com função kernel de base radial (sigma = 0,003; C = 128); Modelo ANN (size = 7; decay = 0,003) e Modelo M5' (pruned = Yes; smoothed = No). Os modelos foram ajustados ao conjunto de dados completo (n = 13.831) e disponibilizados na aplicação web no seguinte endereço: https://deivisonsouza.shinyapps.io/MLVolume/.

Na modelagem do volume comercial com casca, os MAM foram treinados usando as seguintes nomenclaturas para as variáveis: **D** = diâmetro 1,30m do solo (centímetros); **H** = altura total (metros); e **V** = volume comercial com casca (m^3). Portanto, a mesma nomenclatura de variáveis deve ser usada para uma nova base de dados. Assim, os modelos construídos encontrarão as variáveis no conjunto de dados e, portanto, as predições serão realizadas. Também é importante ater-se às unidades de medidas usadas para cada variável.

A aplicação web dispõe de um "botão" para upload de arquivos do tipo .csv com até 100 megabytes. Além disso, a interface de usuário possui oito menus de fácil interatividade que auxiliam na navegação do site e extração de informações do conjunto de dados: 1) About the app; 2) About file; 3) Dataset; 4) Explore; 5) Prediction; 6) Compare; 7) Metrics; e 8) Reference. A segui as principais funcionalidades dos menus são descritas. Um video interativo apresentando o funcionamento da aplicação foi desenvolvido (FIGURA 48).

1. **About the app**: apresenta um breve relato sobre a aplicação desenvolvida. Informa sobre o conjunto de dados usado na modelagem preditiva. Relata a linguagem de programação, os algoritmos e pacotes usados para aprendizado dos modelos preditivos. Destaca os MAM encapsulados e como as variáveis devem ser nomeadas para que os modelos possam realizar corretamente as predições.

2. **About file**: apresenta uma descrição básica do conjunto de dados carregados. Revela o tipo de variáveis e a dimensão do conjunto de dados (quantidade de observações e número de variáveis).

3. **Dataset**: revela o conjunto de dados carregado e possibilita salvar em diferentes extensões (.csv, .xlsx, .pdf).

4. **Explore**: Neste menu, quatro submenus estão disponíveis: **Summary**, **Histogram**, **Scaterplot** e **BoxPlot**. No submenu **Summary**, o usuário tem acesso a uma rápida estatística descritiva para as variáveis numéricas. Em **Histogram**, histogramas de frequência iterativos, para cada variável, podem ser criados. Ao passar o curso sobre o(s) gráfico(s) construídos, o usuário pode consultar para cada classe: i) ponto médio, limites inferior e superior e frequência absoluta. No submenu **Scaterplot**, gráficos de dispersão podem ser criados para mostrar a relação marginal entre as variáveis. Um análise de correlação de Pearson é realizada para a relação específica. Do mesmo modo, ao passar o curso sobre o(s) gráfico(s) construído(s) informações de cada ponto são reveladas. No submenu **BoxPlot**, gráficos boxplots podem ser construídos, e as informações de mínimo, máximo, primeiro e terceiro quartil, mediana e potenciais *outliers* são interativamente visualizadas. Todos os gráficos gerados podem ser salvos no formato .png, e o sumário estatístico pode ser salvo em .csv e .xlsx.

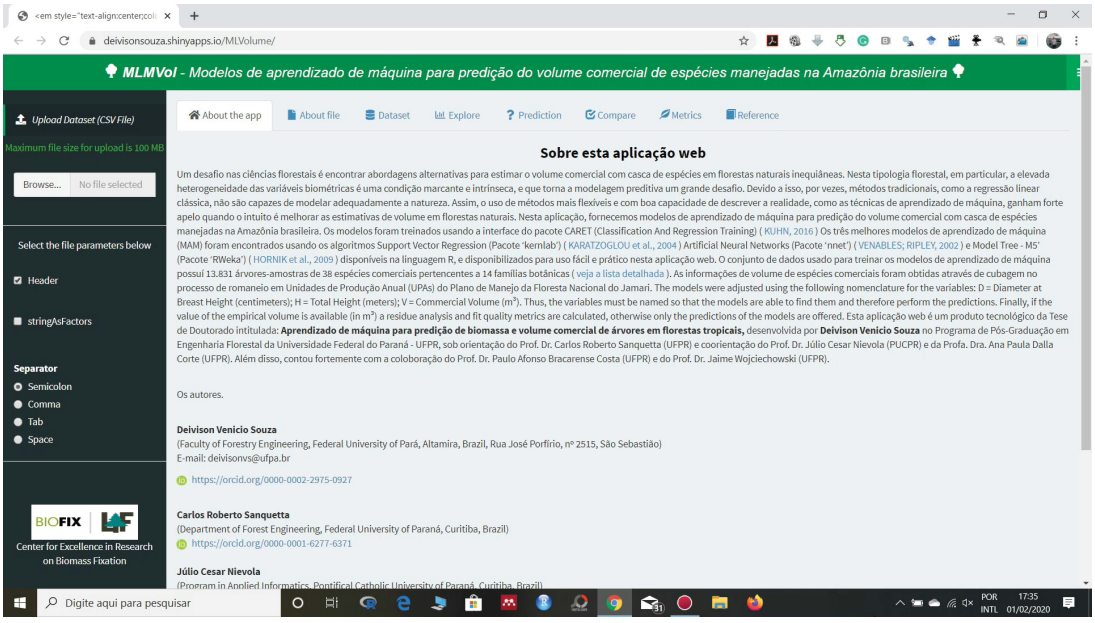
5. **Prediction**: Neste menu, os modelos de aprendizado de máquina são usados para realizar as predições do volume comercial. Na prática, duas situações são possíveis: 1) quando o volume comercial observado está disponível; e 2) quando o volume comercial observado não está disponível. Na primeira situação, um quadro de dados é gerado com as variáveis originais (**D**, **H** e **V**), três colunas com predições (**Pred_SVR**, **Pred_ANN** e **Pred_M5**) e três colunas com os resíduos ordinários (**res_SVR**, **res_ANN** e **res_M5**). Em que, **Pred_SVR**: predição do volume comercial usando o modelo SVR; **pred_ANN**: predição do volume comercial usando o modelo ANN; **Pred_M5**: predição do volume comercial usando o modelo M5'; **res_SVR**: resíduos ordinários para o modelo SVR; **res_ANN**: resíduos ordinários para o modelo ANN; **res_M5**: resíduos ordinários para o modelo M5'. Na segunda situação, uma vez que o volume observado não está disponível, não é possível realizar o cálculo de resíduos.

6. **Compare**: Neste menu, gráfico(s) de valores observados como função dos preditos podem ser visualizados. Diversas métricas de desempenho são calculadas (MSE, RMSE, RMSE%, r, R^2, Bias, Bias%, MAE, MAPE). Além disso, um quadro interativo revela os maiores resíduos, para cada modelo. Os gráficos e tabela com resíduos ranqueados, em ordem decrescente, estão disponíveis para download.

7. **Metrics**: apresenta as expressões matemáticas das métricas calculadas no menu **Compare**.

8. **Reference**: lista os pacotes da linguagem R usados para a construção da aplicação web.

FIGURA 48 – ANIMAÇÃO DEMONSTRANDO O FUNCIONAMENTO DA APLICAÇÃO WEB MLMVOL.



FONTE: O autor (2020).
NOTA: O arquivo com a animação pode ser acessado no endereço: https://github.com/DeivisonSouza/PhD-thesis.

4.2.7 Conclusão

- O sucesso do emprego de técnicas de aprendizado de máquina para predizer o volume comercial com casca em florestas naturais inequiâneas foi admitido e mostrou-se bastante promissor. Em particular, a construção de um espaço de recursos dependente de diâmetro e altura da árvore favoreceu a construção de modelos de aprendizado de máquina mais acurados, e com precisão similar aos modelos de dupla entrada com resposta logarítmica (Spurr e Schumacher-Hall).
- Quando usado um espaço de recursos dependente da altura e diâmetro, alguns MAM parecem mostrar resíduos mais comportados (melhor distribuídos em torno da média zero) do que os modelos de dupla entrada com resposta logarítmica (Spurr e Schumacher-Hall), apesar da análise visual subjetiva ainda indicar presença de heterocedasticidade residual para os MAM.
- Na modelagem tradicional, os modelos de resposta logarítmica com a variável altura inclusa (Spurr e Schumacher-Hall) apresentaram melhores estatísticas de qualidade de ajuste, porém a rejeição da hipótese subjacente de homocedasticidade não conservou confiança às interpretações inferenciais, pois assumiu-se que as variâncias dos estimadores de MQO são tendenciosas.
- O modelo de Husch, com apenas o diâmetro incluso, apesar da rejeição da hipótese subjacente de homocedasticidade, parece apresentar resíduos de MQO mais comportados comparativamente aos ajustes Spurr e Schumacher-Hall, que apresentaram forte padrão sistemático na distribuição residual.
- As variáveis diâmetro e altura da árvore em conjunto contribuem para explicar grande parte das varições do volume comercial com casca em florestas naturais inequiâneas, seja usando a modelagem tradicional, seja empregando técnicas de aprendizado de máquina. No entanto, em florestas nativas determinar com precisão a altura de árvores em pé é um grande desafio. Portanto, se as medidas de altura não puderem ser tomadas com qualidade, parece ser razoável considerar o uso de modelos preditivos que incorporem apenas a variável diâmetro.

5 CONSIDERAÇÕES FINAIS E PESQUISAS FUTURAS

A experiência adquirida durante o desenvolvimento desta pesquisa permitiu sumarizar alguns argumentos favoráveis e contra-argumentos no emprego das técnicas de regressão linear clássica e aprendizado de máquina (FIGURA 49).

FIGURA 49 – ARGUMENTOS FAVORÁVEIS E CONTRA-ARGUMENTOS NO EMPREGO DAS TÉCNICAS DE REGRESSÃO LINEAR CLÁSSICA E APRENDIZADO DE MÁQUINA.

	Argumentos favoráveis	Contra-argumentos
Regressão Linear Clássica	- **Modelagem**: a modelagem preditiva é simples e rápida; - **Interpretabilidade**: o(s) parâmetro(s) estimado(s) associado(s) à(s) covariável(is) podem ser interpretados; e - **Estimação intervalar**: possibilita estimar a incerteza relacionada à(s) estimativa(s) do(s) parâmetro(s) do modelo estatístico (intervalos de confiança e predição).	- **Formas funcionais**: exige a especificação *a priori* de formas funcionais; - **Pressuposições**: exige o atendimento à pressuposições sobre o termo de erro estocástico, quando o interesse é estimação intervalar; e - **Relações complexas**: possuem maior dificuldade em modelar relações não-lineares mais complexas.
Técnicas de Aprendizado de Máquina	- **Formas funcionais**: não exige a especificação *a priori* de formas funcionais; - **Flexíveis**: são mais eficazes para modelar relações não-lineares mais complexas; - **Versáteis**: podem ser aplicados à diversas situações; e - **Variáveis qualitativas**: são flexíveis à inclusão de variáveis qualitativas no espaço de preditores, estratégia que pode contribuir para construção de modelos biométricos mais acurados.	- **Modelagem**: o processo de modelagem preditiva é mais complexo; - **Custo**: o custo computacional no processo de aprendizagem dos modelos é maior; - **Interpretabilidade:** a compreensão/interpretação da importância de variáveis preditoras e a relação preditor-resposta em modelos de aprendizado de máquina é mais difícil, mas exigem técnicas desenvolvidas para esta finalidade; e - **Modelador:** da parte do modelador exige tempo e dedicação na busca de ótimos ajustes.

FONTE: O autor (2020).

1. **Considerações finais**

   a) *Aprendizado de Máquina versus Modelagem Tradicional*

      i. **Capacidade preditiva**: O emprego de técnicas de aprendizado de máquina para estimativa de biomassa aérea total e volume comercial de espécies manejadas em florestas naturais inequiâneas é bastante promissor, apesar do desempenho preditivo similar à abordagem tradicional.

      ii. **Interpretabilidade**: Na Mensuração Florestal, é tradicional o uso de regressão linear simples para predição de variáveis biométricas, como biomassa e volume de árvores individuais. Um modelo tradicional simples e bastante admitido nos estudos de modelagem preditiva é, por exemplo, $v = \beta_0 + \beta_1 d + \epsilon_i$ (Berkhout), em que: v = volume e d = diâmetro. É fato que para o modelo de Berkhout existe uma "relação biológica" entre "v" e "d" o que torna a função ajustada facilmente explicável pela interpretação direta do parâmetro 'β_1 associado à covariável d (diâmetro). Porém, quando técnicas de aprendizado de máquina são empregadas para modelar algum fenômeno da natureza, a metáfora "caixa preta" é bastante usual e alvo de bastante discussão. A metáfora é usada para simbolizar a dificuldade e/ou impossibilidade de explicar a função modelada pelos algoritmos de aprendizado supervisionado. Felizmente, algumas iniciativas têm demonstrado que é possível obter algum nível de explicação para os modelos de aprendizado de máquina. No ambiente R, por exemplo, inúmeras bibliotecas especializadas, como DALEX (*Descriptive mAchine Learning Explanations*), disponibilizam métodos desenvolvidos para prover algum nível de interpretabilidade para os modelos de aprendizado de máquina. Assim, já existem métodos disponíveis para responder questões como: Qual a importância relativa de preditoras para o modelo de aprendizado de máquina? Qual a relação (ou efeito) marginal entre resposta-preditor no modelo? As abordagens podem auxiliar na melhor compreensão dos modelos aprendidos e, posteriormente, em tomadas de decisões. Para além disso, alguns algoritmos podem prover MAM extremamente intuitivos e altamente interpretáveis, como é caso das árvores de regressão e árvores modelos (M5').

      iii. **Complexidade da modelagem**: A complexidade computacional dos algoritmos de aprendizado de máquina é bastante discutida. Obviamente, é verdadeiro que o custo computacional do processo de treinamento de modelos de rede neural artificial ou *k*-NN, por exemplo, é maior do que o ajuste de uma regressão linear simples. Apesar disso, devido a evolução computacional e o surgimento de computadores com grande capacidade de processamento, treinar modelos de aprendizado de máquina não parece ser um grande empecilho. Em especial, na Mensuração Florestal, as pesquisas estão longe de possuir *big datas*, que indiscutivelmente encarecem o processo de treinamento de MAM. Por outro lado, a experiência desta pesquisa, levar a admitir que o processo de busca por hiperparâmetros de ajuste ótimo, através da estratégia *grid search*, requer uma boa parte do tempo e dedicação do modelador, devido a necessidade contínua de avaliação e refinamento dos valores dos hiperparâmetros candidatos. Para além disso, a questão de seleção e engenharia de recursos também é um fator crucial para o sucesso da modelagem, sendo o conhecimento de domínio fundamental neste processo.

   b) *Vantagens das técnicas de AM frente à abordagem de regressão clássica*

      i. **Técnicas de AM não fazem suposição sobre a forma funcional**: Em geral, os métodos paramétricos, como a regressão linear clássica, envolvem a suposição sobre a forma funcional da relação entre a resposta e o(s) preditor(es).

Por exemplo, para estimar o volume de árvores em um determinado povoamento pode-se partir da suposição de que o modelo estatístico de Berkhout ($v = \beta_0 + \beta_1 d + \epsilon_i$) é razoável para descrever a relação existente entre volume e diâmetro e, portanto, é um problema linear. Em seguida, as estimativas dos parâmetros do modelo podem ser obtidas usando, por exemplo, método dos Mínimos Quadrados Ordinários (MQO). Em suma, na regressão tradicional primeiro o pesquisador faz a especificação de um modelo razoável e, somente depois realiza a modelagem preditiva. O grande problema é que muitas vezes a verdadeira relação entre a variável resposta e preditoras é desconhecida. Assim, ao usar abordagens de aprendizado de máquina a fase de especificação de formas funcionais é abandonada, e isso pode ser considerado uma vantagem frente à regressão clássica.

ii. **Técnicas de AM são eficazes para modelar relações não lineares complexas**: Por natureza, os modelos de regressão linear assumem que existe uma relação linear entre a variável resposta e o(s) preditor(es). No entanto, em muitos casos, a relação verdadeira entre a resposta e o(s) preditor(es) pode não ser linear. Na Mensuração Florestal, a regressão polinomial é recorrentemente usada para acomodar relações não-lineares, porém a depender da complexidade da relação entre variáveis, o uso de funções polinomiais pode não ser suficiente para construção de modelos acurados. Portanto, em situações de existência de relações não-lineares complexas, espera-se que as abordagem de aprendizagem de máquina sejam superiores aos métodos lineares tradicionais.

c) *Desenvolvimento de aplicação web*

i. Modelos de aprendizado de máquina (MAM), em geral, possuem estrutura bastante complexa e, portanto, o uso prático e intuitivo dos modelos aprendidos é bastante dificultoso. Assim, para superar as dificuldades de uso dos MAM duas aplicações web foram construídas usando o *framework* 'Shiny': **MLMBio** e **MLMVol**. **MLMBio** é uma aplicação web com MAM para predição da biomassa aérea total de árvores em florestas tropicais. Os modelos mais acurados encontrados para estimar a BAT estão disponíveis: Modelo ANN (size = 9; decay = 0,2) e Modelo SGB (interaction.depth = 5; shrinkage = 0,01; n.trees = 1500; n.minobsinnode = 5). O modelo pantropical desenvolvido por Chave *et al.* (2014) também está disponível para comparação com os MAM. A aplicação está acessível em: https://deivisonsouza.shinyapps.io/MLBiomass/. **MLMVol** é uma aplicação web com MAM para predição do volume comercial com casca de espécies manejadas na Amazônia brasileira. Os três modelos mais acurados encontrados para estimar o volume comercial estão disponíveis: Modelo SVR com função kernel de base radial (sigma = 0,003; C = 128); Modelo ANN (size = 7; decay = 0,003) e Modelo M5' (pruned = Yes; smoothed = No). A aplicação está acessível em: https://deivisonsouza.shinyapps.io/MLVolume/.

2. **Pesquisas futuras**

a) *Reflexão de novas preditoras para melhoria da precisão*

i. Algumas pesquisas têm sido desenvolvidas na busca de modelos mais acurados para estimativa de volume e biomassa de árvores. Apesar disso, em florestas naturais inequiâneas, a maioria desses estudos têm focado na avaliação de estratégias e/ou métodos estatísticos, sem refletir a possibilidade de inclusão de novas preditoras para além daquelas tradicionalmente usadas em modelos de regressão linear. Assim, é possível que a melhoria da precisão dos modelos biométricos em florestas naturais inequiâneas, não esteja simplesmente condicionada ao método estatístico de modelagem preditiva, mas também à reflexão de outras variáveis omitidas (ou negligenciadas), que podem ter influência na variável resposta. Portanto, ponderar diferentes abordagens de modelagem e refletir novas variáveis preditoras, parece ser o caminho a seguir na buscar por modelos biométricos mais acurados, especialmente em situações de alta heterogeneidade, condição intrínseca de variáveis biométricas em florestas naturais inequiâneas.

b) *Proposições de modificações de algoritmos baseado no conhecimento de domínio*

i. Em termos gerais, os algoritmos de aprendizagem de máquina aqui avaliados apresentaram desempenho bastante similar aos modelos clássicos de regressão linear. Alguns algoritmos podem ser extremamente intuitivos e interpretáveis, como as árvores de regressão e árvores modelos (M5'). As árvores modelos, por exemplo, usam modelos lineares como preditores nos nós terminais da árvore crescida. Portanto, uma questão de pesquisa que emergiu foi: Seria possível modificar o algoritmo M5' para considerar modelos tradicionalmente usados na Ciência Florestal nos nós terminais? Particularmente, é apenas uma questão vaga que precisa ser amadurecida, e isso só será possível com o completo entendimento da matemática subjacente do algoritmo M5'. Uma linha similar de raciocínio poderia ser usada para refletir proposições de modificações para as árvores de regressão. Em verdade, isso não implica em criar um novo algoritmo, apenas modificar passos específicos apoiados no conhecimento de domínio.

c) *Avaliação de novos algoritmos de aprendizado de máquina*

i. No mundo, é constante o desenvolvimento e proposição de novos algoritmos de aprendizado de máquina com capacidade de modelar a natureza. Algumas vezes as proposições são grandes inovações, porém, na maioria das vezes, as proposições são pequenas mudanças ou acréscimos de hiperparâmetros que melhoram significativamente o desempenho preditivo de técnicas de aprendizado de máquina tradicionais. Neste cenário, é desejável que as pesquisas florestais acompanhem os avanços tecnológicos e avaliem a adequação das novas técnicas em surgimento para modelar as variáveis florestais de interesse.

REFERÊNCIAS

AGUIRRE-SALADO, C. A. *et al.* Mapping aboveground tree carbon in managed Patula pine forests in Hidalgo, México. **Agrociencia (Montecillo)**, Colegio de Postgraduados, v. 43, n. 2, p. 209–220, 2009. ISSN 1405-3195. Disponível em: http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-31952009000200011&nrm=iso. Citado 3 vezes nas páginas 25, 31, 33.

ALLEN, D. M. The relationship between variable selection and data agumentation and a method for prediction. **Technometrics**, Taylor & Francis, v. 16, n. 1, p. 125–127, 1974. DOI: 10.1080/00401706.1974.10489157. Citado 1 vez na página 68.

ALTMAN, D. G.; BLAND, J. M. Measurement in medicine: the analysis of method comparison studies. **Journal of the Royal Statistical Society: Series D (The Statistician)**, Wiley Online Library, v. 32, n. 3, p. 307–317, 1983. DOI: 10.2307/2987937. Citado 1 vez na página 77.

ANSCOMBE, F. J. Graphs in statistical analysis. **The American Statistician**, Taylor & Francis Group, v. 27, n. 1, p. 17–21, 1973. DOI: 10.2307/2682899. Citado 1 vez na página 93.

ARAÚJO, E. J. G. d. *et al.* Allometric models to biomass in restoration areas in the Atlantic rain forest. **Floresta e Ambiente**, SciELO Brasil, v. 25, n. 1, 2018. DOI: 10.1590/2179-8087.019316. Citado 3 vezes nas páginas 24, 25.

ASHRAF, M. I. *et al.* A novel modelling approach for predicting forest growth and yield under climate change. **PloS one**, Public Library of Science, v. 10, n. 7, e0132066, 2015. DOI: 10.1371/journal.pone.0132066. Citado 2 vezes nas páginas 25, 31.

AWAD, M.; KHANNA, R. **Efficient learning machines: theories, concepts, and applications for engineers and system designers**. [S.l.]: Apress, 2015. Citado 1 vez na página 58.

AYER, T. *et al.* Comparison of logistic regression and artificial neural network models in breast cancer risk estimation. **Radiographics**, Radiological Society of North America, v. 30, n. 1, p. 13–22, 2010. DOI: 10.1148/rg.301095057. Citado 2 vez na página 71.

BARRETO, W. F. *et al.* Equação de volume para apoio ao manejo comunitário de empreendimento florestal em Anapu, Pará. **Pesquisa Florestal Brasileira**, v. 34, n. 80, p. 321–329, 2014. DOI: 10.4336/2014.pfb.34.80.721. Citado 1 vez na página 70.

BASAK, D.; PAL, S.; PATRANABIS, D. C. Support vector regression. **Neural Information Processing-Letters and Reviews**, v. 11, n. 10, p. 203–224, 2007. Disponível em: https://pdfs.semanticscholar.org/c5a9/67eaded74a9fc414de4ad5120b0b66acd2c3.pdf. Citado 1 vez na página 58.

BASKERVILLE, G. Use of logarithmic regression in the estimation of plant biomass. **Canadian Journal of Forest Research**, NRC Research Press, v. 2, n. 1, p. 49–53, 1972. DOI: 10.1139/x72-009. Citado 4 vezes nas páginas 63, 68, 93, 107, 111.

BATISTA, J.; COUTO, H.; SILVA FILHO, D. **Quantificação de recursos florestais: árvores, arvoredos e florestas**. 1. ed. São Paulo: Oficina de Textos, 2014. 384 p. ISBN 9788579751530. Citado 4 vezes nas páginas 29, 30.

BERGMEIR, C.; BENÍTEZ, J. M. Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. **Journal of Statistical Software**, v. 46, n. 7, p. 1–26, 2012. DOI: 10.18637/jss.v046.i07. Citado 1 vez na página 61.

BIECEK, P. DALEX: explainers for complex predictive models. **ArXiv e-prints**, 2018. arXiv: 1806.08915 [stat.ML]. Disponível em: https://arxiv.org/abs/1806.08915. Citado 4 vezes nas páginas 26, 89, 92, 97, 98, 123, 125, 126.

BINOTI, D. H. B.; SILVA BINOTI, M. L. M. da; LEITE, H. G. Configuração de redes neurais artificiais para estimação do volume de árvores. **Revista Ciência da Madeira (Brazilian Journal of Wood Science)**, v. 5, n. 1, p. 10–12953, 2014. Citado 2 vezes nas páginas 25, 31.

BINOTI, M. L. M. S.; BINOTI, D. H. B.; LEITE, H. G. APLICAÇÃO DE REDES NEURAIS ARTIFICIAIS PARA ESTIMAÇÃO DA ALTURA DE POVOAMENTOS EQUIÂNEOS DE EUCALIPTO. **Revista Árvore**, Universidade Federal de Viçosa, v. 37, n. 4, p. 639–645, 2013. DOI: 10.1590/S0100-67622013000400007. Citado 3 vezes nas páginas 25, 31, 32.

BLAND, J. M.; ALTMAN, D. G. Statistical methods for assessing agreement between two methods of clinical measurement. **International journal of nursing studies**, Elsevier, v. 47, n. 8, p. 931–936, 2010. DOI: 10.1016/S0140-6736(86)90837-8. Citado 1 vez na página 77.

BRAGA, A. P.; CARVALHO, A. P. L. F.; LUDERMIR, T. B. **Redes neurais artificiais: teoria e aplicações**. 2.ed. [S.l.]: LTC Editora Rio de Janeiro, Brazil: 2016. Citado 5 vezes nas páginas 59, 60.

BREIMAN, L. Bagging predictors. **Machine learning**, Springer, v. 24, n. 2, p. 123–140, 1996. DOI: 10.1007/BF00058655. Citado 2 vez na página 48.

BREIMAN, L. Random forests. **Machine learning**, Springer, v. 45, n. 1, p. 5–32, 2001. Disponível em: https://link.springer.com/content/pdf/10.1023/A:1010933404324.pdf. Citado 3 vezes nas páginas 50, 52.

BREIMAN, L. Statistical modeling: The two cultures (with comments and a rejoinder by the author). **Statistical science**, Institute of Mathematical Statistics, v. 16, n. 3, p. 199–231, 2001. DOI: 10.1214/ss/1009213726. Citado 1 vez na página 25.

BREIMAN, L. **Using adaptive bagging to debias regressions**. [S.l.], 1999. Disponível em: https://www.stat.berkeley.edu/users/breiman/adaptbag99.pdf. Citado 1 vez na página 54.

BREIMAN, L. *et al.* Classification and Regression Trees. Wadsworth, 1984. Citado 7 vezes nas páginas 41, 43, 44, 46.

BROWN, S.; GILLESPIE, A.; LUGO, A. Biomass estimation methods for tropical forests with applications to forest inventory data. **Forest science**, Oxford University Press, v. 35, n. 4, p. 881–902, 1989. Citado 6 vezes nas páginas 24, 30, 70.

CASSON, R. J.; FARMER, L. D. Understanding and checking the assumptions of linear regression: a primer for medical researchers. **Clinical & experimental ophthalmology**, Wiley Online Library, v. 42, n. 6, p. 590–596, 2014. DOI: 10.1111/ceo.12358. Citado 1 vez na página 108.

CHAI, T.; DRAXLER, R. R. Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. **Geoscientific model development**, Copernicus GmbH, v. 7, n. 3, p. 1247–1250, 2014. DOI: 10.5194/gmd-7-1247-2014. Citado 1 vez na página 75.

CHAMBERS, J. *et al.* Tree damage, allometric relationships, and above-ground net primary production in central Amazon forest. **Forest Ecology and Management**, Elsevier, v. 152, n. 1-3, p. 73–84, 2001. DOI: 10.1016/S0378-1127(00)00591-0. Citado 2 vez na página 30.

CHANG, W. *et al.* **shiny: Web Application Framework for R**. [S.l.], 2019. R package version 1.3.2. Disponível em: https://CRAN.R-project.org/package=shiny. Citado 2 vezes nas páginas 26, 78.

CHAVE, J. *et al.* Improved allometric models to estimate the aboveground biomass of tropical trees. **Global change biology**, Wiley Online Library, v. 20, n. 10, p. 3177–3190, 2014. DOI: 10.1111/gcb.12629. Citado 28 vezes nas páginas 24, 27, 30, 63, 64, 68, 70, 78, 81, 93, 96, 100, 101, 132.

CHAVE, J. *et al.* Tree allometry and improved estimation of carbon stocks and balance in tropical forests. **Oecologia**, Springer, v. 145, n. 1, p. 87–99, 2005. DOI: 10.1007/s00442-005-0100-x. Citado 8 vezes nas páginas 24, 30, 63, 70.

CHAVE, J. *et al.* Error propagation and scaling for tropical forest biomass estimates. **Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences**, The Royal Society, v. 359, n. 1443, p. 409–420, 2004. DOI: 10.1098/rstb.2003.1425. Citado 2 vez na página 30.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: ACM. PROCEEDINGS of the 22nd acm sigkdd international conference on knowledge discovery and data mining. [S.l.: s.n.], 2016. P. 785–794. DOI: 10.1145/2939672.2939785. Citado 3 vez na página 56.

CHEN, T. *et al.* **xgboost: Extreme Gradient Boosting**. [S.l.], 2019. R package version 0.82.1. Disponível em: https://CRAN.R-project.org/package=xgboost. Citado 4 vezes nas páginas 56, 74, 75.

CORTE, A. P. D.; SILVA, F.; SANQUETTA, C. R. Fator de expansão de biomassa e razão de raizes-parte aérea para Pinus spp. plantadas no sul do Brasil. **Floresta**, v. 42, n. 4, p. 755–768, 2012. DOI: 10.5380/rf.v42i4.17771. Citado 2 vez na página 30.

CSÁRDI, G. **cranlogs: Download Logs from the 'RStudio' 'CRAN' Mirror**. [S.l.], 2019. R package version 2.1.1. Disponível em: https://CRAN.R-project.org/package=cranlogs. Citado 0 vez na página 35.

CYSNEIROS, V. *et al.* Modelos genéricos e específicos para estimativa do volume comercial em uma floresta sob concessão na Amazônia. **Scientia Forestalis**, v. 45, n. 114, p. 295–304, 2017. DOI: 10.18671/scifor.v45n114.06. Citado 5 vezes nas páginas 24, 25, 70, 104.

DASE, R.; PAWAR, D. Application of Artificial Neural Network for stock market predictions: A review of literature. **International Journal of Machine Intelligence**, Bioinfo Publications, 49, Vighnahar Shopping Complex Kharghar, Navi Mumbai, v. 2, n. 2, p. 14–17, 2010. DOI: 10.9735/0975-2927.2.2.14-17. Citado 1 vez na página 59.

DATTA, D. **blandr: a Bland-Altman Method Comparison package for R**. [S.l.], 2017. DOI: 10.5281/zenodo.824514. Disponível em: https://github.com/deepankardatta/blandr. Citado 1 vezes nas páginas 77, 87, 117.

DIAMANTOPOULOU, M. J. Artificial neural networks as an alternative tool in pine bark volume estimation. **Computers and electronics in agriculture**, Elsevier, v. 48, n. 3, p. 235–244, 2005. DOI: 10.1016/j.compag.2005.04.002. Citado 1 vez na página 31.

DIAMANTOPOULOU, M. J.; MILIOS, E. Modelling total volume of dominant pine trees in reforestations via multivariate analysis and artificial neural network models. **Biosystems engineering**, Elsevier, v. 105, n. 3, p. 306–315, 2010. DOI: 10.1016/j.biosystemseng.2009.11.010. Citado 3 vezes nas páginas 25, 31, 32.

ELITH, J.; LEATHWICK, J. R.; HASTIE, T. A working guide to boosted regression trees. **Journal of Animal Ecology**, Wiley Online Library, v. 77, n. 4, p. 802–813, 2008. DOI: 10.1111/j.1365-2656.2008.01390.x. Citado 2 vezes nas páginas 54, 56.

FARAWAY, J. **faraway: Functions and Datasets for Books by Julian Faraway**. [S.l.], 2016. R package version 1.0.7. Disponível em: https://CRAN.R-project.org/package=faraway. Citado 1 vez na página 69.

FATH, A. H.; MADANIFAR, F.; ABBASI, M. Implementation of multilayer perceptron (MLP) and radial basis function (RBF) neural networks to predict solution gas-oil ratio of crude oil systems. **Petroleum**, Elsevier, 2018. DOI: 10.1016/j.petlm.2018.12.002. Citado 1 vez na página 60.

FEHRMANN, L. *et al.* Comparison of linear and mixed-effect regression models and ak-nearest neighbour approach for estimation of single-tree biomass. **Canadian Journal of Forest Research**, NRC Research Press, v. 38, n. 1, p. 1–9, 2008. DOI: 10.1139/X07-119. Citado 9 vezes nas páginas 24–26, 31, 32, 78.

FELDPAUSCH, T. R. *et al.* Tree height integrated into pantropical forest biomass estimates. **Biogeosciences**, York, p. 3381–3403, 2012. DOI: 10.5194/bg-9-3381-2012. Citado 3 vez na página 30.

FOX, J.; WEISBERG, S. **An R Companion to Applied Regression**. Third. Thousand Oaks CA: Sage, 2019. Disponível em: https://socialsciences.mcmaster.ca/jfox/Books/Companion/. Citado 2 vez na página 69.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. **Journal of computer and system sciences**, Elsevier, v. 55, n. 1, p. 119–139, 1997. DOI: 10.1006/jcss.1997.1504. Citado 1 vez na página 52.

FRIEDMAN, J. H. Stochastic gradient boosting. **Computational statistics & data analysis**, Elsevier, v. 38, n. 4, p. 367–378, 2002. DOI: 10.1016/S0167-9473(01)00065-2. Citado 2 vez na página 54.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. **Annals of statistics**, JSTOR, p. 1189–1232, 2001. Disponível em: https://www.jstor.org/stable/pdf/2699986.pdf?casa_token=al4ShJql9A8AAAAA:sTcFijb-qkllSYg1js53hFKkORcGsjrQlVzTPZUiZYkVf4V3ylIYXWgxTE-lzxwAXc2p2ByeXPOB-KBnZk84bbRvbZyf6EAEscAowTam76RH5FDFqe_fcQ. Citado 4 vezes nas páginas 26, 56, 89, 120.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). **The annals of statistics**, Institute of Mathematical Statistics, v. 28, n. 2, p. 337–407, 2000. DOI: 10.1214/aos/1016218223. Citado 2 vezes nas páginas 53, 56.

FRITSCH, S.; GUENTHER, F.; WRIGHT, M. N. **neuralnet: Training of Neural Networks**. [S.l.], 2019. R package version 1.44.2. Disponível em: https://CRAN.R-project.org/package=neuralnet. Citado 1 vez na página 61.

GAMA, J. *et al.* **Extração de conhecimento de dados: data mining**. [S.l.: s.n.], 2015. ISBN 978-972-618-811-7. Citado 5 vezes nas páginas 44, 48, 60.

GAVRILOV, I.; PUSEV, R. **normtest: Tests for Normality**. [S.l.], 2014. R package version 1.1. Disponível em: https://CRAN.R-project.org/package=normtest. Citado 2 vez na página 69.

GEVREY, M.; DIMOPOULOS, I.; LEK, S. Review and comparison of methods to study the contribution of variables in artificial neural network models. **Ecological modelling**, Elsevier, v. 160, n. 3, p. 249–264, 2003. DOI: 10.1016/S0304-3800(02)00257-0. Citado 1 vez na página 89.

GIAVARINA, D. Understanding bland altman analysis. **Biochemia medica: Biochemia medica**, Medicinska naklada, v. 25, n. 2, p. 141–151, 2015. DOI: 10.11613/BM.2015.015. Citado 2 vezes nas páginas 77, 78.

GOODMAN, R. C.; PHILLIPS, O. L.; BAKER, T. R. The importance of crown dimensions to improve tropical tree biomass estimates. **Ecological Applications**, Wiley Online Library, v. 24, n. 4, p. 680–698, 2014. DOI: 10.1890/13-0070.1. Citado 1 vez na página 30.

GOUSSANOU, C. A. *et al.* Specific and generic stem biomass and volume models of tree species in a West African tropical semi-deciduous forest. **Silva Fennica**, v. 50, n. 2, p. 1474, 2016. DOI: 10.14214/sf.1474. Citado 2 vez na página 24.

GREENWELL, B. *et al.* **gbm: Generalized Boosted Regression Models**. [S.l.], 2019. R package version 2.1.5. Disponível em: https://CRAN.R-project.org/package=gbm. Citado 4 vezes nas páginas 54, 55, 74, 75.

GUJARATI, D.; PORTER, D. **Econometria Básica**. 5. ed. Porto Alegre: AMGH Editora Ltda, 2011. 924 p. Citado 11 vezes nas páginas 69, 94, 96, 105, 108, 111.

HAARA, A.; MALTAMO, M.; TOKOLA, T. The K-nearest-neighbour method for estimating basal-area diameter distribution. **Scandinavian Journal of Forest Research**, Taylor & Francis, v. 12, n. 2, p. 200–208, 1997. DOI: 10.1080/02827589709355401. Citado 1 vez na página 25.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. New York: Springer, 2016. 745 p. Citado 4 vezes nas páginas 25, 40, 50, 51.

HAYKIN, S. **Redes neurais: princípios e prática**. 2.ed. [S.l.]: Bookman Editora, 2001. Citado 3 vezes nas páginas 59, 60.

HE, Q. Estimation of coniferous forest above-ground biomass using LiDAR and SPOT-5 data. In: IEEE. 2012 2nd International Conference on Remote Sensing, Environment and Transportation Engineering. [S.l.: s.n.], 2012. P. 1–4. DOI: 10.1109/RSETE.2012.6260562. Citado 1 vez na página 30.

HECHENBICHLER, K.; SCHLIEP, K. Weighted k-nearest-neighbor techniques and ordinal classification, 2004. Citado 4 vezes nas páginas 40–42.

HIGA, R. C. V. *et al.* Protocolo de medição e estimativa de biomassa e carbono florestal. **Colombo: Embrapa Florestas**, 2014. Citado 1 vez na página 29.

HIGUCHI, N. *et al.* Biomassa da parte aérea da vegetação da floresta tropical úmida de terra-firme da Amazônia brasileira. **Acta Amazonica**, SciELO Brasil, v. 28, n. 2, p. 153–153, 1998. DOI: 10.1590/1809-43921998282166. Citado 3 vezes nas páginas 24, 30.

HIRAKATA, V. N.; CAMEY, S. A. Análise de concordância entre métodos de Bland-Altman. **Clinical & Biomedical Research**, v. 29, n. 3, p. 261–268, 2009. ISSN 2357-9730. Disponível em: https://seer.ufrgs.br/hcpa/article/view/11727/7021. Citado 3 vezes nas páginas 77, 78.

HORNIK, K.; BUCHTA, C.; ZEILEIS, A. Open-Source Machine Learning: R Meets Weka. **Computational Statistics**, v. 24, n. 2, p. 225–232, 2009. DOI: 10.1007/s00180-008-0119-7. Citado 3 vezes nas páginas 47, 74, 75.

ISHIBUCHI, H.; NOJIMA, Y. Repeated double cross-validation for choosing a single solution in evolutionary multi-objective fuzzy classifier design. **Knowledge-Based Systems**, Elsevier, v. 54, p. 22–31, 2013. DOI: 10.1016/j.knosys.2013.09.023. Citado 1 vez na página 71.

JAMES, G. *et al.* **An introduction to statistical learning**. [S.l.]: Springer, 2013. v. 112. Citado 16 vezes nas páginas 25, 43, 44, 48–50, 53, 54, 71.

JARQUE, C. M.; BERA, A. K. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. **Economics letters**, Elsevier, v. 6, n. 3, p. 255–259, 1980. DOI: 10.1016/0165-1765(80)90024-5. Citado 1 vez na página 95.

JIANG, G.; WANG, W. Error estimation based on variance analysis of k-fold cross-validation. **Pattern Recognition**, Elsevier, v. 69, p. 94–106, 2017. DOI: 10.1016/j.patcog.2017.03.025. Citado 4 vezes nas páginas 71, 72.

JORDAN, M. I.; MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. **Science**, American Association for the Advancement of Science, v. 349, n. 6245, p. 255–260, 2015. DOI: 10.1126/science.aaa8415. Citado 2 vezes nas páginas 25, 33.

KARATZOGLOU, A. *et al.* kernlab – An S4 Package for Kernel Methods in R. **Journal of Statistical Software**, v. 11, n. 9, p. 1–20, 2004. Disponível em: http://www.jstatsoft.org/v11/i09/. Citado 3 vezes nas páginas 58, 74, 75.

KAVAKLIOGLU, K. Modeling and prediction of Turkey's electricity consumption using Support Vector Regression. **Applied Energy**, Elsevier, v. 88, n. 1, p. 368–375, 2011. DOI: 10.1016/j.apenergy.2010.07.021. Citado 1 vez na página 57.

KIM, J.-H. Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. **Computational statistics & data analysis**, Elsevier, v. 53, n. 11, p. 3735–3745, 2009. DOI: 10.1016/j.csda.2009.04.009. Citado 2 vez na página 72.

KORHONEN, K. T.; KANGAS, A. Application of nearest-neighbour regression for generalizing sample tree information. **Scandinavian Journal of Forest Research**, Taylor & Francis, v. 12, n. 1, p. 97–101, 1997. DOI: 10.1080/02827589709355389. Citado 1 vez na página 33.

KOZAK, A.; KOZAK, R. Does cross validation provide additional information in the evaluation of regression models? **Canadian Journal of Forest Research**, NRC Research Press, v. 33, n. 6, p. 976–987, 2003. DOI: 10.1139/x03-022. Citado 1 vez na página 93.

KUHN, M. A Short Introduction to the caret Package. **R Found Stat Comput**, Citeseer, p. 1–10, 2015. Citado 1 vez na página 77.

KUHN, M. Building Predictive Models in R Using the caret Package. **Journal of Statistical Software, Articles**, v. 28, n. 5, p. 1–26, 2008. ISSN 1548-7660. DOI: 10.18637/jss.v028.i05. Disponível em: https://www.jstatsoft.org/v028/i05. Citado 8 vezes nas páginas 35, 37, 70, 71.

KUHN, M. Caret: classification and regression training. **Astrophysics Source Code Library**, 2015. Disponível em: https://ascl.net/1505.003. Citado 2 vezes nas páginas 40, 70.

KUHN, M.; JOHNSON, K. **Applied predictive modeling**. [S.l.]: Springer, 2013. v. 810. Citado 31 vezes nas páginas 36, 44, 46, 48–51, 53, 54, 57, 70–72, 75, 77, 86, 89.

KUHN, M. *et al.* **caret: Classification and Regression Training**. [S.l.], 2016. R package version 6.0-73. Disponível em: https://CRAN.R-project.org/package=caret. Citado 8 vezes nas páginas 37, 38, 40, 70, 72, 73, 83, 89, 90, 121, 122.

KUPLICH, T. M. Classifying regenerating forest stages in Amazonia using remotely sensed images and a neural network. **Forest Ecology and Management**, Elsevier, v. 234, n. 1-3, p. 1–9, 2006. DOI: 10.1016/j.foreco.2006.05.066. Citado 2 vezes nas páginas 25, 31.

KVÅLSETH, T. O. Cautionary note about R2. **The American Statistician**, Taylor & Francis, v. 39, n. 4, p. 279–285, 1985. DOI: 10.2307/2683704. Citado 1 vez na página 75.

LEITE, H. *et al.* Redes Neurais Artificiais para a estimação da densidade básica da madeira. **Scientia Forestalis**, v. 44, n. 109, p. 149–154, 2016. DOI: 10.18671/scifor.v44n109.14. Citado 2 vezes nas páginas 25, 31.

LEWIS, R. J. An introduction to classification and regression tree (CART) analysis. In: ANNUAL meeting of the society for academic emergency medicine in San Francisco, California. [S.l.: s.n.], 2000. v. 14. Disponível em: https://www.researchgate.net/profile/Roger_Lewis6/publication/240719582_An_Introduction_to_Classification_and_Regression_Tree_CART_Analysis/links/0046352d3fb18f1740000000/An-Introduction-to-Classification-and-Regression-Tree-CART-Analysis.pdf. Citado 2 vezes nas páginas 43, 44.

LI, C.; QIU, Z.; LIU, C. An improved weighted k-nearest neighbor algorithm for indoor positioning. **Wireless Personal Communications**, Springer, v. 96, n. 2, p. 2239–2251, 2017. DOI: 10.1007/s11277-017-4295-z. Citado 1 vez na página 40.

LI, X. *et al.* Estimating bamboo forest aboveground biomass using EnKF-assimilated MODIS LAI spatiotemporal data and machine learning algorithms. **Agricultural and Forest Meteorology**, Elsevier, v. 256, p. 445–457, 2018. DOI: 10.1016/j.agrformet.2018.04.002. Citado 1 vez na página 30.

LIANG, T.; DAVIER, A. A. von. Cross-validation: An alternative bandwidth-selection method in kernel equating. **Applied Psychological Measurement**, Sage Publications Sage CA: Los Angeles, CA, v. 38, n. 4, p. 281–295, 2014. DOI: 10.1177/0146621613518094. Citado 1 vez na página 71.

LIAW, A.; WIENER, M. Classification and Regression by randomForest. **R News**, v. 2, n. 3, p. 18–22, 2002. Disponível em: https://www.researchgate.net/profile/Andy_Liaw/publication/228451484_Classification_and_Regression_by_RandomForest/links/53fb24cc0cf20a45497047ab/Classification-and-Regression-by-RandomForest.pdf. Citado 3 vezes nas páginas 52, 74, 75.

LIM, H. *et al.* Biomass expansion factors and allometric equations in an age sequence for Japanese cedar (Cryptomeria japonica) in southern Korea. **Journal of forest research**, Springer, v. 18, n. 4, p. 316–322, 2013. DOI: 10.1007/s10310-012-0353-2. Citado 1 vez na página 30.

LOETSCH, F.; ZOHRER, F.; HALLER, K. Forest inventory. V. 2. **Munique, B LV-Verlagsgesellschaft**, 1973. Citado 0 vez na página 67.

MALTAMO, M. *et al.* Most similar neighbour-based stand variable estimation for use in inventory by compartments in Finland. **Forestry: An International Journal of Forest Research**, v. 76, n. 4, p. 449–464, 2003. DOI: 10.1093/forestry/76.4.449. Citado 1 vez na página 25.

MALTAMO, M.; EERIKAINEN, K. The most similar neighbour reference in the yield prediction of Pinus kesiya stands in Zambia. **Silva Fennica**, THE FINNISH SOCIETY OF FOREST SCIENCE, v. 35, n. 4, p. 437–451, 2001. DOI: 10.14214/sf.579. Citado 2 vezes nas páginas 25, 33.

MAROCO, J. **Análise estatística: com utilização do SPSS**. 3. ed. Lisboa: [s.n.], 2010. 824 p. Citado 1 vez na página 105.

MARTINS, J. *et al.* A database for automatic classification of forest species. **Machine vision and applications**, Springer, v. 24, n. 3, p. 567–578, 2013. DOI: 10.1007/s00138-012-0417-5. Citado 5 vezes nas páginas 25, 31, 33, 78.

MARTINS, J. *et al.* Forest species recognition based on dynamic classifier selection and dissimilarity feature vector representation. **Machine Vision and Applications**, Springer, v. 26, n. 2-3, p. 279–293, 2015. DOI: 10.1007/s00138-015-0659-0. Citado 4 vezes nas páginas 25, 31, 33, 78.

MARUYAMA, T. *et al.* Automatic classification of native wood charcoal. **Ecological informatics**, Elsevier, v. 46, p. 1–7, 2018. DOI: 10.1016/j.ecoinf.2018.05.008. Citado 3 vezes nas páginas 25, 31, 33.

MATE, R.; JOHANSSON, T.; SITOE, A. Biomass equations for tropical forest tree species in Mozambique. **Forests**, Multidisciplinary Digital Publishing Institute, v. 5, n. 3, p. 535–556, 2014. DOI: 10.3390/f5030535. Citado 1 vez na página 30.

MAUYA, E. *et al.* Modelling aboveground forest biomass using airborne laser scanner data in the miombo woodlands of Tanzania. **Carbon balance and management**, BioMed Central, v. 10, n. 1, p. 28, 2015. DOI: 10.1186/s13021-015-0037-2. Citado 2 vezes nas páginas 64, 75.

MCCARTHY, J. **Programs with common sense**. [S.l.]: RLE e MIT computation center, 1959. P. 300–307. Citado 1 vez na página 31.

MCROBERTS, R. E. Estimating forest attribute parameters for small areas using nearest neighbors techniques. **Forest Ecology and Management**, Elsevier, v. 272, p. 3–12, 2012. DOI: 10.1016/j.foreco.2011.06.039. Citado 3 vezes nas páginas 25, 31, 33.

MCROBERTS, R. E.; NÆSSET, E.; GOBAKKEN, T. Optimizing the k-Nearest Neighbors technique for estimating forest aboveground biomass using airborne laser scanning data. **Remote Sensing of Environment**, Elsevier, v. 163, p. 13–22, 2015. DOI: 10.1016/j.rse.2015.02.026. Citado 1 vez na página 30.

MEYER, D. *et al.* **e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien**. [S.l.], 2019. R package version 1.7-2. Disponível em: https://CRAN.R-project.org/package=e1071. Citado 1 vez na página 58.

MITCHELL, R.; FRANK, E. Accelerating the XGBoost algorithm using GPU computing. **PeerJ Computer Science**, PeerJ Inc., v. 3, e127, 2017. DOI: 10.7717/peerj-cs.127. Citado 1 vez na página 56.

MOLINARO, A. M.; SIMON, R.; PFEIFFER, R. M. Prediction error estimation: a comparison of resampling methods. **Bioinformatics**, Oxford University Press, v. 21, n. 15, p. 3301–3307, 2005. DOI: 10.1093/bioinformatics/bti499. Citado 4 vezes nas páginas 71, 72.

MOLNAR, C.; BISCHL, B.; CASALICCHIO, G. iml: An R package for Interpretable Machine Learning. **JOSS**, Journal of Open Source Software, v. 3, n. 26, p. 786, 2018. DOI: 10.21105/joss.00786. Disponível em: http://joss.theoj.org/papers/10.21105/joss.00786. Citado 2 vezes nas páginas 26, 89.

MONTAÑO, R. A. N. R. *et al.* Artificial Intelligence Models to Estimate Biomass of Tropical Forest Trees. **Polibits**, v. 56, p. 29–37, 2017. DOI: 10.17562/PB-56-4. Citado 4 vezes nas páginas 25, 26, 32, 78.

MORAL, R. A.; HINDE, J.; DEMÉTRIO, C. G. B. Half-Normal Plots and Overdispersed Models in R: The hnp Package. **Journal of Statistical Software**, v. 81, n. 10, p. 1–23, 2017. DOI: 10.18637/jss.v081.i10. Citado 1 vez na página 64.

MORETTIN, P. A.; BUSSAB, W. O. **Estatística básica**. 9. ed. São Paulo: Editora Saraiva, 2017. 554 p. Citado 1 vez na página 108.

MOUSELIMIS, L. KernelKnn: Kernel k nearest neighbors. **R package version 1.0.8**, 2018. Disponível em: https://cran.r-project.org/web/packages/KernelKnn/index.html. Citado 1 vez na página 40.

NAESSET, E. *et al.* Mapping and estimating forest area and aboveground biomass in miombo woodlands in Tanzania using data from airborne laser scanning, TanDEM-X, RapidEye, and global forest maps: A comparison of estimated precision. **Remote sensing of Environment**, Elsevier, v. 175, p. 282–300, 2016. DOI: 10.1016/j.rse.2016.01.006. Citado 1 vez na página 30.

NASTOS, P. *et al.* Rain intensity forecast using Artificial Neural Networks in Athens, Greece. **Atmospheric Research**, v. 119, p. 153–160, 2013. ADVANCES IN PRECIPITATION SCIENCE. ISSN 0169-8095. DOI: 10.1016/j.atmosres.2011.07.020. Citado 1 vez na página 60.

NATH, A. J. *et al.* Allometric models for estimation of forest biomass in North East India. **Forests**, Multidisciplinary Digital Publishing Institute, v. 10, n. 2, p. 103, 2019. DOI: 10.3390/f10020103. Citado 1 vez na página 30.

NIETO, P. G. *et al.* Support vector machines and neural networks used to evaluate paper manufactured using Eucalyptus globulus. **Applied Mathematical Modelling**, Elsevier, v. 36, n. 12, p. 6137–6145, 2012. DOI: 10.1016/j.apm.2012.02.016. Citado 4 vezes nas páginas 25, 31, 32.

NJANA, M. A. Indirect methods of tree biomass estimation and their uncertainties. **Southern Forests: a Journal of Forest Science**, Taylor & Francis, v. 79, n. 1, p. 41–49, 2017. DOI: 10.2989/20702620.2016.1233753. Citado 1 vez na página 30.

NOGUEIRA, E. M. *et al.* Estimates of forest biomass in the Brazilian Amazon: new allometric equations and adjustments to biomass from wood-volume inventories. **Forest Ecology and Management**, Elsevier, v. 256, n. 11, p. 1853–1867, 2008. DOI: 10.1016/j.foreco.2008.07.022. Citado 2 vez na página 30.

NORD-LARSEN, T.; NIELSEN, A. T. Biomass, stem basic density and expansion factor functions for five exotic conifers grown in Denmark. **Scandinavian journal of forest research**, Taylor & Francis, v. 30, n. 2, p. 135–153, 2015. DOI: 10.1080/02827581.2014.986519. Citado 1 vez na página 30.

NUNES, M. H.; GÖRGENS, E. B. Artificial intelligence procedures for tree taper estimation within a complex vegetation mosaic in Brazil. **PloS one**, Public Library of Science, v. 11, n. 5, e0154738, 2016. DOI: 10.1371/journal.pone.0154738. Citado 1 vez na página 32.

ODOR, P. M.; BAMPOE, S.; CECCONI, M. Cardiac Output Monitoring: Validation Studies–how Results Should be Presented. **Current anesthesiology reports**, Springer, v. 7, n. 4, p. 410–415, 2017. DOI: 10.1007/s40140-017-0239-0. Citado 3 vez na página 77.

OLIVEIRA, L. Z. *et al.* Robust volumetric models for supporting the management of secondary forest stands in the Southern Brazilian Atlantic Forest. **Anais da Academia Brasileira de Ciências**, SciELO Brasil, v. 90, n. 4, p. 3729–3744, 2018. DOI: 10.1590/0001-3765201820180111. Citado 1 vez na página 68.

OUNBAN, W.; PUANGCHIT, L.; DILOKSUMPUN, S. Development of general biomass allometric equations for Tectona grandis Linn. f. and Eucalyptus camaldulensis Dehnh. plantations in Thailand. **Agriculture and Natural Resources**, Elsevier, v. 50, n. 1, p. 48–53, 2016. DOI: 10.1016/j.anres.2015.08.001. Citado 1 vez na página 24.

OVERMAN, J. P. M.; WITTE, H. J. L.; SALDARRIAGA, J. G. Evaluation of regression models for above-ground biomass determination in Amazon rainforest. **Journal of tropical Ecology**, Cambridge University Press, v. 10, n. 2, p. 207–218, 1994. DOI: 10.1017/S0266467400007859. Citado 1 vez na página 30.

ÖZÇELIK, R. *et al.* Estimating tree bole volume using artificial neural network models for four species in Turkey. **Journal of environmental management**, Elsevier, v. 91, n. 3, p. 742–753, 2010. DOI: 10.1016/j.jenvman.2009.10.002. Citado 1 vez na página 32.

PAULA FILHO, P. L. *et al.* Forest species recognition using macroscopic images. **Machine Vision and Applications**, Springer, v. 25, n. 4, p. 1019–1031, 2014. DOI: 10.1007/s00138-014-0592-7. Citado 4 vezes nas páginas 25, 31, 33, 78.

PEDERSEN, T. L.; BENESTY, M. **lime: Local Interpretable Model-Agnostic Explanations**. [S.l.], 2019. R package version 0.5.0. Disponível em: https://CRAN.R-project.org/package=lime. Citado 2 vezes nas páginas 26, 89.

PELISSARI, A. L.; LANSSANOVA, L. R.; DRESCHER, R. Modelos volumétricos para Pinus tropicais, em povoamento homogêneo, no Estado de Rondônia. **Pesquisa Florestal Brasileira**, v. 31, n. 67, p. 173, 2011. DOI: 10.4336/2011.pfb.31.67.173. Citado 1 vez na página 24.

PETERS, A.; HOTHORN, T. **ipred: Improved Predictors**. [S.l.], 2019. R package version 0.9-9. Disponível em: https://CRAN.R-project.org/package=ipred. Citado 3 vezes nas páginas 49, 50, 74, 75.

PRETZSCH, H. Forest dynamics, growth, and yield. In: FOREST dynamics, growth and yield. [S.l.]: Springer, 2009. P. 1–39. Citado 1 vez na página 75.

PROBST, P. *et al.* **Tunability: Importance of hyperparameters of machine learning algorithms**. [S.l.: s.n.], 2018. eprint: 1802.09596. Disponível em: https://arxiv.org/abs/1802.09596. Citado 2 vez na página 38.

PRODAN, M. **Mensura forestal**. [S.l.]: Agroamerica, 1997. Citado 0 vez na página 67.

PUT, R. *et al.* Classification and regression tree analysis for molecular descriptor selection and retention prediction in chromatographic quantitative structure–retention relationship studies. **Journal of Chromatography A**, Elsevier, v. 988, n. 2, p. 261–276, 2003. DOI: 10.1016/S0021-9673(03)00004-9. Citado 1 vez na página 41.

QIN, X.; HAN, J. Variable selection issues in tree-based regression models. **Transportation Research Record**, SAGE Publications Sage CA: Los Angeles, CA, v. 2061, n. 1, p. 30–38, 2008. DOI: 10.3141/2061-04. Citado 1 vez na página 44.

QUINLAN, J. R. Learning with continuous classes. In: 5TH Australian Joint Conference on Artificial Intelligence. Singapore: World Scientific, 1992. v. 92, p. 343–348. Citado 3 vezes nas páginas 41, 46.

R CORE TEAM. **R: A Language and Environment for Statistical Computing**. Vienna, Austria, 2019. Disponível em: https://www.R-project.org/. Citado 3 vezes nas páginas 67, 69, 71.

REFAEILZADEH, P.; TANG, L.; LIU, H. Cross Validation, Encyclopedia of Database Systems (EDBS). **Arizona State University, Springer**, p. 532–538, 2009. Citado 4 vezes nas páginas 71, 72.

REIS, L. P. *et al.* Estimation of mortality and survival of individual trees after harvesting wood using artificial neural networks in the amazon rain forest. **Ecological Engineering**, Elsevier BV, v. 112, p. 140–147, mar. 2018. DOI: 10.1016/j.ecoleng.2017.12.014. Citado 3 vezes nas páginas 25, 26, 31.

REIS, L. P. *et al.* Modeling of tree recruitment by artificial neural networks after wood harvesting in a forest in eastern Amazon rain forest. **Ciência Florestal**, Universidad Federal de Santa Maria, v. 29, n. 2, p. 583, jun. 2019. DOI: 10.5902/1980509825808. Citado 3 vezes nas páginas 25, 26, 31.

RÉJOU-MÉCHAIN, M. *et al.* biomass: an r package for estimating above-ground biomass and its uncertainty in tropical forests. **Methods in Ecology and Evolution**, Wiley Online Library, v. 8, n. 9, p. 1163–1167, 2017. DOI: 10.1111/2041-210X.12753. Citado 2 vezes nas páginas 63, 93.

RÉJOU-MÉCHAIN, M. *et al.* **BIOMASS: Estimating Aboveground Biomass and Its Uncertainty in Tropical Forests**. [S.l.], 2017. R package version 1.1. Disponível em: https://CRAN.R-project.org/package=BIOMASS. Citado 2 vezes nas páginas 63, 93.

RIDGEWAY, G. The state of boosting. **Computing Science and Statistics**, p. 172–181, 1999. Disponível em: https://pdfs.semanticscholar.org/1aac/6453fbb8333ee638b6d8b2bb2aff06c3654b.pdf. Citado 2 vez na página 53.

ROBNIK-SIKONJA, M.; SAVICKY, P. CORElearn: Classification, regression and feature evaluation. **R package version**, v. 1, n. 3, 2017. Disponível em: https://cran.r-project.org/web/packages/CORElearn/index.html. Citado 1 vez na página 40.

ROCHA, S. J. S. S. da *et al.* Artificial neural networks: Modeling tree survival and mortality in the Atlantic Forest biome in Brazil. **Science of The Total Environment**, Elsevier BV, v. 645, p. 655–661, dez. 2018. DOI: 10.1016/j.scitotenv.2018.07.123. Citado 3 vezes nas páginas 25, 26, 31.

SAKR, G. E. *et al.* Artificial intelligence for forest fire prediction. In: IEEE. 2010 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. [S.l.: s.n.], 2010. P. 1311–1316. DOI: 10.1109/AIM.2010.5695809. Citado 2 vezes nas páginas 25, 31.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of research and development**, n. 3, p. 211–229, 1959. DOI: 10.1147/rd.33.0210. Citado 2 vez na página 31.

SANQUETTA, C. R.; BALBINOT, R. **Metodologias para determinação de Biomassa Florestal**. Edição: Carlos Roberto Sanquetta, Rafaelo Balbinot e Marco A. Zilliotto. Curitiba, PR: UFPR, 2004. P. 77–93. 205 p. ISBN 8590090647. Citado 2 vezes nas páginas 29, 30.

SANQUETTA, C. R. *et al.* **As florestas e o carbono**. Curitiba: Imprensa Universitária da UFPR, 2002. v. 1. 264 p. Citado 3 vezes nas páginas 29, 30.

SANQUETTA, C. R. *et al.* Estimativa de carbono individual para Araucaria angustifolia. **Pesquisa Agropecuária Tropical**, Escola de Agronomia e Engenharia de Alimentos, v. 44, n. 1, p. 1–8, 2014. DOI: 10.1590/S1983-40632014000100006. Citado 2 vez na página 30.

SANQUETTA, C. R. *et al.* On the use of data mining for estimating carbon storage in the trees. **Carbon balance and management**, BioMed Central, v. 8, n. 1, p. 6, 2013. DOI: 10.1186/1750-0680-8-6. Citado 5 vezes nas páginas 25, 26, 31, 32.

SANQUETTA, C. R. *et al.* Comparison of data mining and allometric model in estimation of tree biomass. **BMC bioinformatics**, BioMed Central, v. 16, n. 1, p. 247, 2015. DOI: 10.1186/s12859-015-0662-5. Citado 6 vezes nas páginas 25, 26, 31, 32, 78.

SANQUETTA, C. R. *et al.* Volume estimation of Cryptomeria japonica logs in southern Brazil using artificial intelligence models. **Southern Forests: a Journal of Forest Science**, Taylor & Francis, v. 80, n. 1, p. 29–36, 2018. DOI: 10.2989/20702620.2016.1263013. Citado 5 vezes nas páginas 26, 31, 32, 78.

SCHIKOWSKI, A. B.; CORTE, A. P. D.; SANQUETTA, C. R. Estudo da forma do fuste utilizando redes neurais artificiais e funções de afilamento. **Pesquisa Florestal Brasileira**, v. 35, n. 82, p. 119–127, 2015. DOI: 10.4336/2015.pfb.35.82.867. Citado 3 vezes nas páginas 25, 31, 32.

SCHIKOWSKI, A. B. *et al.* Modeling of stem form and volume through machine learning. **Anais da Academia Brasileira de Ciências**, SciELO Brasil, v. 90, n. 4, p. 3389–3401, 2018. DOI: 10.1590/0001-3765201820170569. Citado 2 vezes nas páginas 25, 31.

SCHLIEP, K.; HECHENBICHLER, K. **kknn: Weighted k-Nearest Neighbors**. [S.l.], 2016. R package version 1.3.1. Disponível em: https://CRAN.R-project.org/package=kknn. Citado 6 vezes nas páginas 40, 41, 74, 75.

SCHLOERKE, B. *et al.* **GGally: Extension to 'ggplot2'**. [S.l.], 2018. R package version 1.4.0. Disponível em: https://CRAN.R-project.org/package=GGally. Citado 0 vez na página 106.

SHAHID, R. *et al.* Comparison of distance measures in spatial analytical modeling for health service planning. **BMC health services research**, BioMed Central, v. 9, n. 1, p. 200, 2009. DOI: 10.1186/1472-6963-9-200. Citado 1 vez na página 41.

SILESHI, G. W. A critical review of forest biomass estimation models, common mistakes and corrective measures. **Forest Ecology and Management**, Elsevier, v. 329, p. 237–254, 2014. DOI: 10.1016/j.foreco.2014.06.026. Citado 4 vezes nas páginas 69, 81, 105, 108.

SILVA, E. N.; SANTANA, A. C. de. Modelos de regressão para estimação do volume de árvores comerciais, em florestas de Paragominas. **Ceres**, v. 61, n. 5, p. 631–636, 2015. DOI: 10.1590/0034-737X201461050005. Citado 1 vez na página 108.

SINGH, V.; MISRA, A. K. Detection of plant leaf diseases using image segmentation and soft computing techniques. **Information processing in Agriculture**, Elsevier, v. 4, n. 1, p. 41–49, 2017. DOI: 10.1016/j.inpa.2016.10.005. Citado 2 vezes nas páginas 25, 31.

SIRONEN, S. *et al.* Estimating individual tree growth with nonparametric methods. **Canadian Journal of Forest Research**, NRC Research Press, v. 33, n. 3, p. 444–449, 2003. DOI: 10.1139/x02-162. Citado 2 vezes nas páginas 25, 33.

SKOVSGAARD, J. P.; NORD-LARSEN, T. Biomass, basic density and biomass expansion factor functions for European beech (Fagus sylvatica L.) in Denmark. **European Journal of Forest Research**, Springer, v. 131, n. 4, p. 1035–1053, 2012. DOI: 10.1007/s10342-011-0575-4. Citado 1 vez na página 30.

SMOLA, A. J.; SCHÖLKOPF, B. A tutorial on support vector regression. **Statistics and computing**, Springer, v. 14, n. 3, p. 199–222, 2004. DOI: 10.1023/B:STCO.0000035301.49549.88. Citado 4 vez na página 58.

SOARES, F. A. A. *et al.* Recursive diameter prediction and volume calculation of eucalyptus trees using Multilayer Perceptron Networks. **Computers and electronics in agriculture**, Elsevier, v. 78, n. 1, p. 19–27, 2011. DOI: 10.1016/j.compag.2011.05.008. Citado 2 vezes nas páginas 25, 31.

SOARES, P.; TOMÉ, M. Biomass expansion factors for Eucalyptus globulus stands in Portugal. **Forest Systems**, INIA, v. 21, p. 141–152, 2012. DOI: 10.5424/fs/2112211-12086. Citado 1 vez na página 30.

SOMOGYI, Z. *et al.* Indirect methods of large-scale forest biomass estimation. **European Journal of Forest Research**, v. 126, n. 2, p. 197–207, 2007. ISSN 1612-4677. DOI: 10.1007/s10342-006-0125-7. Citado 1 vez na página 29.

SONG, Y. *et al.* An efficient instance selection algorithm for k nearest neighbor regression. **Neurocomputing**, Elsevier, v. 251, p. 26–34, 2017. DOI: 10.1016/j.neucom.2017.04.018. Citado 2 vezes nas páginas 40, 76.

SOUZA, D. V. *et al.* k-Nearest Neighbor Regression in the Estimation of Tectona Grandis Trunk Volume in the State of Pará, Brazil. **Journal of Sustainable Forestry**, Taylor & Francis, v. 38, n. 8, p. 755–768, 2019. DOI: 10.1080/10549811.2019.1607391. Citado 1 vez na página 32.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introdução ao datamining: mineração de dados**. [S.l.]: Ciência Moderna, 2009. Citado 1 vez na página 48.

TANAGO, J. G. *et al.* Estimation of above-ground biomass of large tropical trees with terrestrial LiDAR. **Methods in Ecology and Evolution**, Wiley Online Library, v. 9, n. 2, p. 223–234, 2018. DOI: 10.1111/2041-210X.12904. Citado 1 vez na página 30.

TANAKA, S. *et al.* Stand volume estimation using the k-NN technique combined with forest inventory data, satellite image data and additional feature variables. **Remote Sensing**, Multidisciplinary Digital Publishing Institute, v. 7, n. 1, p. 378–394, 2014. DOI: 10.3390/rs70100378. Citado 2 vezes nas páginas 64, 75.

THERNEAU, T.; ATKINSON, B. **rpart: Recursive Partitioning and Regression Trees**. [S.l.], 2019. R package version 4.1-15. Disponível em: https://CRAN.R-project.org/package=rpart. Citado 3 vezes nas páginas 45, 74, 75.

TIAN, M.; CHEN, H.; WANG, Q. Flower identification based on Deep Learning. **Journal of Physics: Conference Series**, IOP Publishing, v. 1237, p. 022060, 2019. DOI: 10.1088/1742-6596/1237/2/022060. Citado 1 vez na página 33.

TONINI, H.; BORGES, R. A. Equação de volume para espécies comerciais em Floresta Ombrófila Densa no sul de Roraima. **Pesquisa Florestal Brasileira**, v. 35, n. 82, p. 31–37, 2015. DOI: 10.4336/2015.pfb.35.82.738. Citado 1 vez na página 70.

TORGO, L. **Data mining with R: learning with case studies**. [S.l.]: Chapman e Hall/CRC, 2017. Citado 4 vezes nas páginas 57, 58, 71.

TRAWIŃSKI, B. *et al.* Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms. **International Journal of Applied Mathematics and Computer Science**, Versita, v. 22, n. 4, p. 867–881, 2012. DOI: 10.2478/v10006-012-0064-z. Citado 1 vez na página 34.

VAHEDI, A. A. Artificial neural network application in comparison with modeling allometric equations for predicting above-ground biomass in the Hyrcanian mixed-beech forests of Iran. **Biomass and Bioenergy**, Elsevier, v. 88, p. 66–76, 2016. DOI: 10.1016/j.biombioe.2016.03.020. Citado 2 vezes nas páginas 25, 31.

VANWINCKELEN, G.; BLOCKEEL, H. On estimating model accuracy with repeated cross-validation. In: BENELEARN 2012: Proceedings of the 21st Belgian-Dutch Conference on Machine Learning. [S.l.: s.n.], 2012. P. 39–44. ISBN 978-94-6197-044-2. Disponível em: $$Uhttps://lirias.kuleuven.be/retrieve/186558$$DOnEstimatingModelAccuracy.pdf%20[freely%20available]. Citado 3 vez na página 71.

VASHUM, K. T.; JAYAKUMAR, S. Methods to estimate above-ground biomass and carbon stock in natural forests-a review. **Journal of Ecosystem & Ecography**, v. 2, n. 4, p. 1–7, 2012. DOI: 10.4172/2157-7625.1000116. Citado 2 vezes nas páginas 29, 30.

VENABLES, W. N.; RIPLEY, B. D. **Modern Applied Statistics with S**. Fourth. New York: Springer, 2002. ISBN 0-387-95457-0. Disponível em: http://www.stats.ox.ac.uk/pub/MASS4. Citado 4 vezes nas páginas 61, 62, 74, 75.

VENABLES, W.; RIPLEY, B. **Modern applied statistics with S**. New York: Springer, 2002. DOI: 10.1007/978-0-387-21706-2. Citado 1 vez na página 40.

VIBRANS, A. C. *et al.* Generic and specific stem volume models for three subtropical forest types in southern Brazil. **Annals of forest science**, Springer, v. 72, n. 6, p. 865–874, 2015. DOI: 10.1007/s13595-015-0481-x. Citado 1 vez na página 24.

VOYANT, C. *et al.* Prediction intervals for global solar irradiation forecasting using regression trees methods. **Renewable energy**, Elsevier, v. 126, p. 332–340, 2018. DOI: 10.1016/j.renene.2018.03.055. Citado 1 vez na página 49.

WANG, Y.; WITTEN, I. H. Induction of model trees for predicting continuous classes. In: PROCEEDINGS of the poster papers of the 9th European Conference on Machine Learning. Prague: University of Economics, Faculty of Informatics e Statistics: Springer, 1997. P. 128–137. Disponível em: https://researchcommons.waikato.ac.nz/bitstream/handle/10289/1183/uow-cs-wp-1996-23.pdf?sequence=1&isAllowed=y. Citado 3 vezes nas páginas 46, 47.

WEISBERG, S. **Applied linear regression**. [S.l.]: John Wiley & Sons, 2005. v. 528. Citado 1 vez na página 95.

WICKHAM, H. **rvest: Easily Harvest (Scrape) Web Pages**. [S.l.], 2019. R package version 0.3.4. Disponível em: https://CRAN.R-project.org/package=rvest. Citado 0 vez na página 34.

WITTEN, I. H.; FRANK, E. **Data Mining: Practical Machine Learning Tools and Techniques**. 2nd. San Francisco: Morgan Kaufmann, 2005. Citado 1 vez na página 47.

WITTEN, I. H. *et al.* **Data Mining: Practical machine learning tools and techniques**. [S.l.]: Morgan Kaufmann, 2017. Citado 12 vezes nas páginas 36, 38, 40, 46, 47, 49, 86.

WONG, T.-T. Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation. **Pattern Recognition**, Elsevier, v. 48, n. 9, p. 2839–2846, 2015. DOI: 10.1016/j.patcog.2015.03.009. Citado 1 vez na página 71.

WU, C. *et al.* Comparison of machine-learning methods for above-ground biomass estimation based on Landsat imagery. **Journal of Applied Remote Sensing**, International Society for Optics e Photonics, v. 10, n. 3, p. 035010, 2016. DOI: 10.1117/1.JRS.10.035010. Citado 1 vez na página 30.

WU, X. *et al.* Top 10 algorithms in data mining. **Knowledge and Information Systems**, v. 14, n. 1, p. 1–37, 2008. DOI: 10.1007/s10115-007-0114-2. Citado 1 vez na página 40.

YADAV, S.; SHUKLA, S. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In: IEEE. 2016 IEEE 6th International Conference on Advanced Computing (IACC). [S.l.: s.n.], 2016. P. 78–83. DOI: 10.1109/IACC.2016.25. Citado 1 vez na página 72.

YIGIT, E. *et al.* A study on visual features of leaves in plant identification using artificial intelligence techniques. **Computers and electronics in agriculture**, Elsevier, v. 156, p. 369–377, 2019. DOI: 10.1016/j.compag.2018.11.036. Citado 1 vez na página 33.

ZEILEIS, A.; HOTHORN, T. Diagnostic Checking in Regression Relationships. **R News**, v. 2, n. 3, p. 7–10, 2002. Disponível em: https://www.semanticscholar.org/paper/Diagnostic-Checking-in-Regression-Relationships-Zeileis-Hothorn/58b4d02ee546c30e31a3ead4ccb956fc451e6e95. Acesso em: 20 dez. 2019. Citado 2 vez na página 69.

ZHANG, J. *et al.* Prediction of protein solvent accessibility using PSO-SVR with multiple sequence-derived features and weighted sliding window scheme. **BioData mining**, BioMed Central, v. 8, n. 1, p. 3, 2015. DOI: 10.1186/s13040-014-0031-3. Citado 2 vez na página 75.

APÊNDICES

APÊNDICE A – ESTIMATIVA DE DESEMPENHO MÉDIO E RESÍDUOS APÓS A REMOÇÃO DE PONTOS DISCREPANTES

TABELA 19 – CONFIGURAÇÃO ÓTIMA DE HIPERPARÂMETROS PARA CADA ALGORITMO E ESTIMATIVA DE DESEMPENHO MÉDIO NO ESQUEMA 5x10-FOLDS CV APÓS A REMOÇÃO DE PONTOS DISCREPANTES.

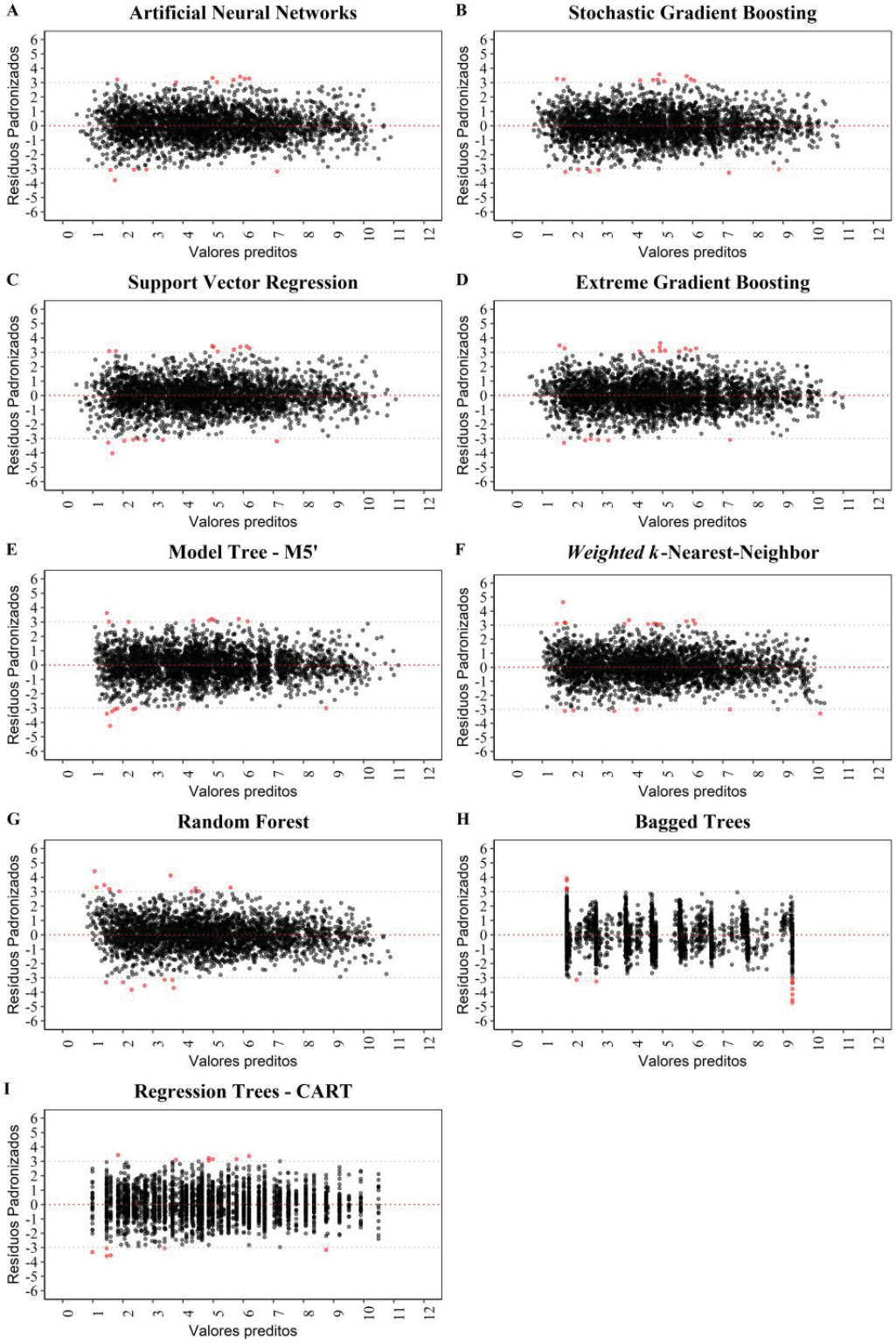
Modelo	Pré-Processamento	Hiperparâmetro tuning	Conjunto de validação (folds = 50)				
			RMSE	r	R^2	MAE	Bias%
RNA	Center and Scale; BoxCox	size = 8 decay = 0,2	0,3295 (0,0109)	0,988 (0,0008)	0,9762 (0,0015)	0,2561 (0,0092)	0,0012 (0,4141)
Boosting	Center and Scale; BoxCox	interaction.depth = 5 shrinkage = 0,01 n.trees = 1500 n.minobsinnode = 5	0,3292 (0,0126)	0,9881 (0,00083)	0,9763 (0,0016)	0,2561 (0,0111)	0,0047 (0,4374)
SVM Radial	Center and Scale; BoxCox	sigma = 0,005 C = 512	0,3308 (0,0119)	0,9879 (0,00081)	0,9761 (0,0016)	0,2568 (0,0103)	-0,0399 (0,4249)
XGBoost	Center and Scale; BoxCox	rounds = 500 eta = 0,05 max_depth = 2	0,3322 (0,0121)	0,9878 (0,0016)	0,9758 (0,0016)	0.2589 (0,0109)	0,0064 (0,4392)
M5’	-	pruned = Yes smoothed = No	0,3354 (0,0116)	0,9876 (0,0008)	0,9754 (0,0016)	0.2617 (0,0106)	-0,0009 (0,4322)
FA	-	mtry = 1 ntree = 500	0,3405 (0,0125)	0,9872 (0,0009)	0,9746 (0,0019)	0,2644 (0,0111)	0,0364 (0,4335)
*wk*NN	Center and Scale; BoxCox	k = 100 kernel = trwg d = 1	0,3376 (0,0135)	0,9875 (0,0009)	0,9752 (0,0018)	0,2622 (0,0116)	0,2059 (0,4252)
CART	-	cp = 0.00007	0,3576 (0,0128)	0,9859 (0,0011)	0,9720 (0,0021)	0,2797 (0,0112)	0,0040 (0,4573)
Bagging CART	Center and Scale; BoxCox	nbagg = 200	0,4189 (0,0181)	0,9807 (0,0013)	0,9617 (0,0026)	0,3304 (0,0146)	-0,0087 (0,5712)

FONTE: O autor (2020)

NOTA: Todas as métricas foram calculadas na escala da variável resposta.

LEGENDA: RNA = Rede Neural Artificial; MVS = Máquina de Vetores de Suporte; XGBoost = Extreme Gradiente Boosting; FA = Floresta Aleatória; RMSE = Root Mean Square Error; r = Coeficiente de correlação de Pearson; R^2 = Coeficiente de determinação; MAE = Mean Absolute Error; Entre parênteses está o desvio padrão na reamostragem.

FIGURA 50 – RESÍDUOS PADRONIZADOS NO CONJUNTO DE TREINAMENTO USANDO OS MODELOS COM CONFIGURAÇÃO ÓTIMA DE HIPERPARÂMETROS APÓS A REMOÇÃO DE PONTOS DISCREPANTES.



FONTE: O autor (2020)

NOTA: Pontos em vermelho indicam resíduos com valores fora do intervalo $-3 \leq z_i \leq 3$.

APÊNDICE B – CÓDIGO REPRODUZÍVEL PARA REPLICAÇÃO DO MODELO PANTROPICAL (n=4004) E AJUSTE DA MESMA FORMA FUNCIONAL USANDO CONJUNTO DE TREINO (n=3226).

```
#############################################################################
#             AJUSTE DE MODELO TRADICIONAL DE BIOMASSA                      #
#############################################################################
# Neste script, a modelagem realizada por Chaves et al. (2014) é
# replicada. Em seguida, a mesma forma funcional usada por Chaves et al.
# (2014) é ajustada aos dados de treinamento, mesmo conjunto usado para
# construir os modelos de aprendizado de máquina.

# Instalação condicional e carrega pacotes
CILP <- function(x){
  for(i in x){
    if(!require(i, character.only = TRUE)){
      install.packages(i, dependencies = TRUE)
      require(i, character.only = TRUE)
    }
  }
}

CILP(c("data.table","caret","Metrics"))

# Carrega conjunto de dados
#-----------------------------------------------------------------------
trainingSet <- fread("trainingSet.csv", stringsAsFactors=T)
testSet <- fread("testSet.csv", stringsAsFactors=T)
AGBiomass <- rbind(trainingSet, testSet)

#############################################################################
#   Modelo Pantropical - (Model 4 em Chave et al., 2014)  (n = 4004)    #
#############################################################################

fitPM <- lm(log(AGB)~I(log(WSG*D^2*H)),data=AGBiomass)
(summaryfitPM <- summary(fitPM))
AIC(fitPM)
confint(fitPM)

#############################################################################
#    Ajusta forma funcional do MP ao conjunto de treino (n = 3226)      #
#############################################################################

fitMA <- lm(log(AGB)~ I(log(WSG*D^2*H)), data=trainingSet)
summary(fitMA)
AIC(fitMA)
confint(fitMA)

#############################################################################
#                Estatísticas de qualidade de ajuste                        #
#############################################################################
# Função com métricas customizadas
Summary <- function(data,lev=NULL,model=NULL){
  out <- c(sqrt(mean((data$pred - data$obs)^2,na.rm=TRUE)),
           ((sqrt(mean((data$pred - data$obs)^2,na.rm = TRUE)))/mean(data$
    ↪ obs,na.rm = TRUE))*100,
           mean((data$pred - data$obs)^2,na.rm = TRUE),
           cor(data$pred, data$obs),
```

```
            (cor(data$pred, data$obs)^2),
            mad(data$pred - data$obs, na.rm = TRUE),
            mean(abs(data$pred - data$obs), na.rm = TRUE),
            mean(data$pred - data$obs, na.rm = TRUE),
            (mean(data$pred - data$obs, na.rm = TRUE)/mean(data$obs,na.rm =
    ↪ TRUE))*100)
  names(out) <- c("RMSE", "RMSE%", "MSE", "r", "Rsq", "MAD", "MAE", "Bias",
    ↪  "Bias%")
  out
}

# Estatisticas de qualidade de ajuste - Chaves et al (2014)
df_PM <- data.frame(obs=log(AGBiomass$AGB),
                    pred=predict(fitPM, AGBiomass))
round(Summary(data=df_PM),4)

# Estatísticas no conjunto de treino - Modelo Alternativo
#-------------------------------------------------------------------------
df_MA <- data.frame(obs=log(trainingSet$AGB),
                      pred=predict(fitMA,trainingSet))
round(Summary(data=df_MA),4)

# Estatísticas no conjunto de teste - Modelo Alternativo
#-------------------------------------------------------------------------
df_MATest <- data.frame(obs=log(testSet$AGB),
                          pred=predict(fitMA,testSet))
round(Summary(data=df_MATest),4)

# Desempenho por sítio (Chaves et al., 2014)
#-------------------------------------------------------------------------
bySite <- AGBiomass[,c(1:5, 16)][,`:=`(Countj=.N, dfj=.N-summaryfitPM$df
    ↪ [1],
                             MAGBj=mean(AGB)), by=Site
                       ][,AGBest_PM:=exp(((summaryfitPM$sigma^2)*0.5)+
                                           fitPM$fitted.values)
                         ][,`:=`(SE_PM=se(AGB,AGBest_PM),
                                 Biasij_PM=(AGBest_PM-MAGBj)/MAGBj)][]

ResultsBySite <- bySite[,list(
  Countj=.N,
  dfj=.N-summaryfitPM$df[1],
  SumAGBj=sum(AGB),
  MAGBj=mean(AGB),
  Biasj_PM=mean(Biasij_PM)*100,
  RSEj_PM=sqrt(sum(SE_PM)/(.N-summaryfitPM$df[1])),
  CVj_PM=(sqrt(sum(SE_PM)/(.N-summaryfitPM$df[1]))/mean(AGB))*100),
  by=list(Site=Site,Localization)]

# Fim
#.....................................................................#
```

APÊNDICE C – CÓDIGO REPRODUZÍVEL PARA TREINAMENTO DE MODELOS DE APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DA BIOMASSA AÉREA TOTAL EM FLORESTAS TROPICAIS.

```
#############################################################################
#             AJUSTE DE MODELOS DE APRENDIZADO DE MÁQUINA                   #
#############################################################################

# Instalação condicional e carrega pacotes
CILP <- function(x){
  for(i in x){
    if(!require(i, character.only = TRUE)){
      install.packages(i, dependencies = TRUE)
      require(i, character.only = TRUE)
    }
  }
}

CILP(c("data.table","caret","rpart","RWeka","ipred","nnet",
       "kknn","randomForest","xgboost","kernlab","gbm"))

# Carrega conjunto de dados
#-----------------------------------------------------------------------
trainingSet <- fread("trainingSet.csv", stringsAsFactors=T)
testSet <- fread("testSet.csv", stringsAsFactors=T)
AGBiomass <- rbind(trainingSet, testSet)

# Configuração do treinamento
#-----------------------------------------------------------------------
# Função com métricas customizadas
Summary <- function(data,lev=NULL,model=NULL){
  out <- c(sqrt(mean((data$pred - data$obs)^2,na.rm=TRUE)),
           ((sqrt(mean((data$pred - data$obs)^2,na.rm = TRUE)))/mean(data$
    ↪ obs,na.rm = TRUE))*100,
           mean((data$pred - data$obs)^2,na.rm = TRUE),
           cor(data$pred, data$obs),
           (cor(data$pred, data$obs)^2),
           mad(data$pred - data$obs, na.rm = TRUE),
           mean(abs(data$pred - data$obs), na.rm = TRUE),
           mean(data$pred - data$obs, na.rm = TRUE),
           (mean(data$pred - data$obs, na.rm = TRUE)/mean(data$obs,na.rm =
    ↪ TRUE))*100)
  names(out) <- c("RMSE", "RMSE%", "MSE", "r", "Rsq", "MAD", "MAE", "Bias",
    ↪  "Bias%")
  out
}

fitControlRT <- trainControl(method="repeatedcv",number=10,repeats=5,
                             returnResamp="final",savePredictions=TRUE,
                             allowParallel=T, summaryFunction=Summary,
                             verboseIter=T, selectionFunction="oneSE")

fitControl <- trainControl(method="repeatedcv",number=10,repeats=5,
                           returnResamp="final",savePredictions=TRUE,
                           allowParallel=T, summaryFunction=Summary,
                           verboseIter=T, selectionFunction="best")

#############################################################################
```

```
#                 ALGORITMO 1: Regression Tree - CART                    #
#########################################################################

# Grade de hiperparâmetros
tuneGridRT <- expand.grid(cp = seq(0.00001, 0.001, 0.00001))

# Treinamento de modelos
set.seed(1000)
rtTune <- train(log(AGB) ~.,
                data=trainingSet[,-c("Site","Localization")],
                method="rpart",
                tuneGrid=tuneGridRT,
                trControl=fitControlRT)

#########################################################################
#                   ALGORITMO 2: Model Tree - M5'                       #
#########################################################################

# Treinamento de modelos
set.seed(1000)
M5Tune <- train(log(AGB) ~.,
                data=trainingSet[,-c("Site","Localization")],
                method='M5',
                trControl=fitControl,
                tuneGrid=expand.grid(pruned=c("Yes","No"),
                                     smoothed=c("No"),
                                     rules=c("No")))

#########################################################################
#                   ALGORITMO 3: Bagged Tree - CART                     #
#########################################################################

# Treinamento de modelos
listModels <- list()

for(nbagg in c(25,50,100,150,200,500,1000, 1500)){

  set.seed(1000)
  BTTune <- train(log(AGB) ~.,
                  data=trainingSet[,-c("Site","Localization")],
                  method='treebag',
                  trControl=fitControl,
                  nbagg=nbagg,
                  control=rpart.control(minsplit=2,cp=0),
                  keepX=TRUE)

  key <- toString(nbagg)
  listModels[[key]] <- BTTune
}

#########################################################################
#             ALGORITMO 4: Artificial Neural Network - ANN              #
#########################################################################

# Grade de hiperparâmetros
tuneGridANN <- expand.grid(.size = seq(1,10,1),
                           .decay = seq(0.1,0.7,0.1))

# Treinamento de modelos
```

```
set.seed(1000)
ANNTune <- train(log(AGB) ~.,
                 data=trainingSet[,-c("Site","Localization")],
                 method="nnet",
                 trControl=fitControl,
                 tuneGrid=tuneGridANN,
                 maxit=1000,
                 preProcess=c("center","scale","BoxCox"),
                 trace=FALSE, linout=TRUE)

#############################################################################
#             ALGORITMO 5: Stochastic Gradient Boosting - SGB               #
#############################################################################

# Grade de hiperparâmetros
tuneGridSGB <- expand.grid(interaction.depth=1:5,
                           shrinkage=c(0.01,0.1,0.3),
                           n.trees=c(10,50,100,500,1000,1500),
                           n.minobsinnode=c(5,10,15,30))

# Treinamento de modelos
set.seed(1000)
sgbTune <- train(log(AGB) ~.,
                 data=trainingSet[,-c("Site","Localization")],
                 method="gbm",
                 trControl=fitControl,
                 tuneGrid=tuneGridSGB,
                 distribution="gaussian",
                 verbose=TRUE,
                 preProc=c("center","scale","BoxCox"))

#############################################################################
#           ALGORITMO 6: Suporte Vector Regression - SVR                    #
#############################################################################

# Grade de hiperparâmetros
tuneGridSVR <- expand.grid(sigma= c(0.0005,0.0001,0.005,0.001,0.1,0.5),
                           C= 2^c(5,6,7,8,9,10))

# Treinamento de modelos
set.seed(1000)
svrTune <- train(log(AGB) ~.,
                 data=trainingSet[,-c("Site","Localization")],
                 method='svmRadial',
                 trControl=fitControl,
                 tuneGrid=tuneGridSVR,
                 preProcess=c("center","scale","BoxCox"))

#############################################################################
#         ALGORITMO 7: Extreme Gradient Boosting - XGBoost                  #
#############################################################################

# Grade de hiperparâmetros
tuneGridXGB <- expand.grid(nrounds=seq(200,1000,50),
                           eta=c(0.025,0.05,0.1,0.3),
                           max_depth=2:6,
                           gamma=1,
                           colsample_bytree=seq(0.5,1,0.1),
                           min_child_weight=1,
```

```
                            subsample=1)

# Treinamento de modelos
set.seed(1000)
xgbTune <- train(log(AGB) ~.,
                 data=trainingSet[,-c("Site","Localization")],
                 method="xgbTree",
                 trControl=fitControl,
                 tuneGrid=tuneGridXGB,
                 verbose=TRUE,
                 preProcess=c("center", "scale", "BoxCox"),
                 nthread=6)

#########################################################################
#                   ALGORITMO 8: Random Forest - RF                     #
#########################################################################

# Customiza "Random Forest" para admitir hiperparâmetro "ntree"
#----------------------------------------------------------------------
customRF <- list(type="Regression",library="randomForest",loop=NULL)
customRF$parameters <- data.frame(parameter=c("mtry","ntree"),class=rep("
    ↪ numeric",2),label=c("mtry", "ntree"))
customRF$grid <- function(x,y,len=NULL,search="grid"){}
customRF$fit <- function(x,y,wts,param,lev,last,weights,classProbs, ...){
  randomForest(x,y,mtry=param$mtry,ntree=param$ntree, ...)
}
customRF$predict <- function(modelFit,newdata,preProc=NULL,submodels=NULL)
  predict(modelFit,newdata)
customRF$prob <- function(modelFit,newdata,preProc=NULL,submodels=NULL)
  predict(modelFit,newdata,type="prob")
customRF$sort <- function(x) x[order(x[,1]),]
customRF$levels <- function(x) x$classes
#----------------------------------------------------------------------

# Grade de hiperparâmetros
tuneGridRF <- expand.grid(mtry=c(1:13),
                          ntree=c(50,100,150,200,500))

# Treinamento de modelos
set.seed(1000)
rfTune <- train(log(AGB) ~.,
                data=trainingSet[,-c("Site","Localization")],
                method=customRF,
                tuneGrid=tuneGridRF,
                trControl=fitControl,
                importance=TRUE)

#########################################################################
#         ALGORITMO 9: Weighted k-Nearest-Neighbor - wkNN               #
#########################################################################

# Grade de hiperparâmetros
tuneGridWKNN <- expand.grid(kmax=seq(2,150,2),
                            kernel=c("rectangular","biweight","cos",
                                     "epanechnikov","gaussian","inv",
                                     "optimal","rank","triangular",
                                     "triweight"),
                            distance=1:3)

```

```
228 # Treinamento de modelos
229 set.seed(1000)
230 wkNNTune <- train(log(AGB) ~.,
231                   data=trainingSet[,-c("Site","Localization")],
232                   method="kknn",
233                   trControl=fitControl,
234                   tuneGrid=tuneGridWKNN,
235                   preProcess=c("center","scale","BoxCox"))
236 
237 # Fim
238 #.....................................................................#
```

APÊNDICE D – CÓDIGO REPRODUZÍVEL PARA AJUSTE DE MODELOS VOLUMÉTRICOS TRADICIONAIS GENÉRICOS.

```
#########################################################################
#             AJUSTE DE MODELOS TRADICIONAIS GENÉRICOS                  #
#########################################################################

# Instalação condicional e carrega pacotes
CILP <- function(x){
  for(i in x){
    if(!require(i, character.only = TRUE)){
      install.packages(i, dependencies = TRUE)
      require(i, character.only = TRUE)
    }
  }
}

CILP(c("data.table","car","caret","faraway","nortest","lmtest"))

# Carrega conjunto de dados
#------------------------------------------------------------------------
data <- fread("volume.csv", stringsAsFactors=T)

# Divisão aleatória estratificada
#------------------------------------------------------------------------
set.seed(100)
trainIndex <- createDataPartition(y=data$Esp,p=.80,list=FALSE)
trainingSet <- data[trainIndex,]
testSet <- data[-trainIndex,]

# Ajuste de modelos tradicionais
#------------------------------------------------------------------------
form <- c(M1=VR ~ D,
          M2=VR ~ I(D^2),
          M3=VR ~ D + I(D^2),
          M4=log(VR) ~ log(D),
          M5=log(VR) ~ log(D) + I(1/D),
          M6=VR ~ I(D^2*H),
          M7=log(VR) ~ I(log(D^2*H)),
          M8=log(VR) ~ log(D) + log(H),
          M9=VR ~ I(D^2) + I(D^2*H) + I(D*H^2) + I(H^2),
          M10=VR ~ D + I(D^2) + I(D*H) + I(D^2*H) + H)

fit <- lapply(form, function(f){ models <- lm(f, data=trainingSet); models
    ↪ })

# Função para calcular estatísticas de ajustes
#------------------------------------------------------------------------
# Obs.: As predições foram corrigidas pelo fator de Baskerville.

MyGeneric <- function(fit, data, obs){

  lapply(fit, function(f){
    metric <- function(fit,data,obs){

      y <- as.character(fit$terms[[2]])

      if(y[1] == "log"){
```

```
55          FC <- (summary(fit)$sigma^2)/2
56          pred <- exp(FC + predict(fit,data))
57          resid <- pred-obs
58
59        }else{
60          pred <- predict(fit,data)
61        }
62
63        if(length(fit$coefficients) >= 3){
64          vif <- faraway::vif(fit)
65        }else{
66          vif <- NA
67        }
68
69        SQT <- sum((obs - mean(obs))^2)
70        SQRes <- sum((obs - pred)^2)
71        SQReg <- SQT - SQRes
72        n <- nrow(data)
73        p <- length(fit$coefficients)-1
74        QMReg <-SQReg/p
75        QMRes <-SQRes/fit$df.residual
76        R2 <- SQReg/SQT
77        R2a <- 1-(1-R2)*((n-1)/(n-(p+1)))
78        RSE <- sqrt(SQRes/(n-length(fit$coefficients)))
79        RSEP <- (RSE/mean(obs,na.rm = TRUE))*100
80        MSE <- mean(residuals(fit)^2)
81        AIC <- AIC(fit)
82        Fstat <- QMReg/QMRes
83        r <- cor(obs,pred)
84
85        ADtest <- nortest::ad.test(fit$residuals)
86        BPtest <- lmtest::bptest(fit)
87        DWtest <- car::durbinWatsonTest(fit)
88
89        return(list(r=r, R2=R2, R2a=R2a, RSE=RSE, RSEP=RSEP,
90                    MSE=MSE, AIC=AIC, Fstat=Fstat, fit=fit,
91                    vif=vif, ADtest=ADtest, BPtest=BPtest,
92                    DWtest=DWtest))
93      }
94      med <- metric(f, data=data, obs=obs)
95      med
96    })
97  }
98
99  est <- MyGeneric(fit=fit, data=trainingSet, obs=trainingSet$VR)
100
101 # Extrair estatísticas de ajustes (conj. treino)
102 #-------------------------------------------------------------------------
103
104 (df <- as.data.frame(sapply(est, "[", j =
105                               c("r", "R2", "R2a",  "RSE",
106                                 "RSEP","MSE","AIC", "Fstat"))))
107
108
109
110
111 # Desempenho no conjunto de teste
112 #-------------------------------------------------------------------------
113 metricTest <- function(fit,data,obs){
```

```
  y <- as.character(fit$terms[[2]])

  if(y[1] == "log"){
    FM <- exp(summary(fit)$sigma^2*0.5)
    pred <- FM*exp(predict(fit,data))

  }else{
    pred <- predict(fit,data)
  }

  SQT <- sum((obs - mean(obs))^2)
  SQRes <- sum((obs - pred)^2)
  SQReg <- SQT - SQRes
  n <- nrow(data)
  p <- length(fit$coefficients)-1
  QMReg <-SQReg/p
  QMRes <-SQRes/fit$df.residual
  r <- cor(obs,pred)
  R2 <- SQReg/SQT
  R2a <- 1-(1-R2)*((n-1)/(n-(p+1)))
  RSE <- sqrt(SQRes/(n-length(fit$coefficients)))
  RSEP <- (RSE/mean(obs))*100
  AIC <- AIC(fit)

  return(data.frame("r"=r,"R2"=R2, "R2a"=R2a,
                    "RSE"=RSE, "RSEP"=RSEP))
}


(estTest <- sapply(fit, function(f){
  med <- metricTest(f, data=testSet, obs=testSet$VR)
  med
}))
```

```r
#########################################################################
#             FUNÇÃO PARA O CÁLCULO DA ESTATÍSTICA PRESS                 #
#########################################################################
# Obs.: As predições corrigidas pelo fator de Baskerville.

# Parâmetros da função:
# data = data.frame com conjuto de dados
# outcome = variável resposta
# model = modelo de classe "lm"

MyPRESS <- function(data, outcome, model) {
  res <- vector(mode="numeric",nrow(data))
  for (i in 1:nrow(data)) {

    newDATA <- data[-i, ]
    fit <- update(model, data = newDATA)

    cl <- gsub("log\\(", "", fit$terms[[2]])

    if(cl[1] == "log"){
      FC <- exp(summary(fit)$sigma^2*0.5)
      yhat <- FC*exp(predict(fit, newdata=data[i, ]))

    }else{
      yhat <- predict(fit, newdata=data[i, ])
    }

    res[i] <- yhat - outcome[i]
  }
  PRESS <- sum(res^2)
  return(PRESS)
}

# Exemplo de uso:
M1 <- lm(VR ~ D, data=trainingSet)
MyPRESS(data=trainingSet,outcome=trainingSet$VR, model=M1)
```

APÊNDICE E – CÓDIGO REPRODUZÍVEL PARA TREINAMENTO DE MODELOS DE APRENDIZADO DE MÁQUINA PARA PREDIÇÃO DO VOLUME COMERCIAL COM CASCA DE ESPÉCIES MANEJADAS NA AMAZÔNIA BRASILEIRA.

```
#############################################################################
#             AJUSTE DE MODELOS DE APRENDIZADO DE MÁQUINA                   #
#############################################################################

# Instalação condicional e carrega pacotes
CILP <- function(x){
  for(i in x){
    if(!require(i, character.only = TRUE)){
      install.packages(i, dependencies = TRUE)
      require(i, character.only = TRUE)
    }
  }
}

CILP(c("data.table","caret","rpart","RWeka","ipred","nnet",
       "kknn","randomForest","xgboost","kernlab","gbm"))

# Carrega conjunto de dados
#----------------------------------------------------------------------------
data <- fread("volume.csv", stringsAsFactors=T)

# Engenharia de recursos
#----------------------------------------------------------------------------
data[,lnVR:=log(VR)
     ][,D2:=D^2
       ][,lnD:=log(D)
         ][,invD:=1/D
           ][,D2H:=D^2*H
             ][,lnH:=log(H)
               ][,DH2:=D*H^2
                 ][,lnD2H:=log(D^2*H)
                   ][,H2:=H^2
                     ][,DH:=D*H]

# Divisão aleatória estratificada
#----------------------------------------------------------------------------
set.seed(100)
trainIndex <- createDataPartition(y=data$Esp,p=.80,list=FALSE)
trainingSet <- data[trainIndex,][,-"VR"]
testSet <- data[-trainIndex,][,-"VR"]

# Configuração do treinamento
#----------------------------------------------------------------------------
# Função com métricas customizadas
Summary <- function(data,lev=NULL,model=NULL){
  out <- c(sqrt(mean((exp(data$pred) - exp(data$obs))^2,na.rm=TRUE)),
           ((sqrt(mean((exp(data$pred) - exp(data$obs))^2,na.rm = TRUE)))/
    ↪ mean(exp(data$obs),na.rm = TRUE))*100,
           mean((exp(data$pred) - exp(data$obs))^2,na.rm = TRUE),
           cor(exp(data$pred), exp(data$obs)),
           (cor(exp(data$pred), exp(data$obs))^2),
           mad(exp(data$pred) - exp(data$obs), na.rm = TRUE),
           mean(abs(exp(data$pred) - exp(data$obs)), na.rm = TRUE),
           mean(exp(data$pred) - exp(data$obs), na.rm = TRUE),
```

```
            (mean(exp(data$pred) - exp(data$obs), na.rm = TRUE)/mean(exp(
    ↪ data$obs),na.rm = TRUE))*100)
  names(out) <- c("RMSE", "RMSE%", "MSE", "r", "Rsq", "MAD", "MAE", "Bias",
    ↪  "Bias%")
  out
}

fitControlRT <- trainControl(method="cv",number=10,
                             returnResamp="final",savePredictions=TRUE,
                             allowParallel=T, summaryFunction=Summary,
                             verboseIter=T, selectionFunction="oneSE")

fitControl <- trainControl(method="cv",number=10,
                           returnResamp="final",savePredictions=TRUE,
                           allowParallel=T, summaryFunction=Summary,
                           verboseIter=T, selectionFunction="best")

#########################################################################
#                 ALGORITMO 1: Regression Tree - CART                   #
#########################################################################

#.....................................................................#
# ABORDAGEM 1: espaço de características dependente de D e H          #
#.....................................................................#

# Grade de hiperparâmetros
tuneGridRT <- expand.grid(cp = seq(0.00001, 0.001, 0.00001))

# Treinamento de modelos
set.seed(1000)
rtTune <- train(lnVR ~.,
                data=trainingSet[,-"Esp"],
                method="rpart",
                tuneGrid=tuneGridRT,
                trControl=fitControlRT)

#.....................................................................#
# ABORDAGEM 2: espaço de características dependente apenas do D       #
#.....................................................................#

# Treinamento de modelos
set.seed(1000)
rtTune2 <- train(lnVR ~.,
                 data=trainingSet[,c(2,4:7)],
                 method="rpart",
                 tuneGrid=tuneGridRT,
                 trControl=fitControlRT)

#########################################################################
#                   ALGORITMO 2: Model Tree - M5'                       #
#########################################################################

#.....................................................................#
# ABORDAGEM 1: espaço de características dependente de D e H          #
#.....................................................................#

# Treinamento de modelos
set.seed(1000)
M5Tune1 <- train(lnVR~.,
```

```
111                   data=trainingSet[,-"Esp"],
112                   method='M5',
113                   trControl=fitControl,
114                   tuneGrid=expand.grid(pruned=c("Yes","No"),
115                                        smoothed=c("Yes","No"),
116                                        rules=c("No")),
117                   preProcess=c("center","scale","BoxCox"))
118 
119 #.....................................................................#
120 # ABORDAGEM 2: espaço de características dependente apenas do D       #
121 #.....................................................................#
122 
123 # Treinamento de modelos
124 set.seed(1000)
125 M5Tune2 <- train(lnVR~.,
126                   data=trainingSet[,c(2,4:7)],
127                   method='M5',
128                   trControl=fitControl,
129                   tuneGrid=expand.grid(pruned=c("Yes","No"),
130                                        smoothed=c("Yes","No"),
131                                        rules=c("No")),
132                   preProcess=c("center","scale","BoxCox"))
133 
134 ########################################################################
135 #                   ALGORITMO 3: Bagged Tree - CART                    #
136 ########################################################################
137 
138 #.....................................................................#
139 # ABORDAGEM 1: espaço de características dependente de D e H          #
140 #.....................................................................#
141 
142 # Treinamento de modelos
143 listModels <- list()
144 
145 for(nbagg in c(25,50,100,150,200,500,1000, 1500)){
146 
147   set.seed(1000)
148   BTTune1 <- train(lnVR~.,
149                    data=trainingSet[,-"Esp"],
150                    method='treebag',
151                    trControl=fitControl,
152                    preProc=c("center","scale","BoxCox"),
153                    nbagg=nbagg,
154                    control=rpart.control(minsplit=2,cp=0),
155                    keepX=TRUE)
156 
157   key <- toString(nbagg)
158   listModels[[key]] <- BTTune1
159 }
160 
161 #.....................................................................#
162 # ABORDAGEM 2: espaço de características dependente apenas do D       #
163 #.....................................................................#
164 
165 # Treinamento de modelos
166 listModels2 <- list()
167 
168 for(nbagg in c(25,50,100,150,200,500,1000, 1500)){
169 
```

```
  set.seed(1000)
  BTTune2 <- train(lnVR~.,
                   data=trainingSet[,c(2,4:7)],
                   method='treebag',
                   trControl=fitControl,
                   preProc=c("center","scale","BoxCox"),
                   nbagg=nbagg,
                   control=rpart.control(minsplit=2,cp=0),
                   keepX=TRUE)

  key <- toString(nbagg)
  listModels2[[key]] <- BTTune2
}

#########################################################################
#             ALGORITMO 4: Artificial Neural Network - ANN              #
#########################################################################

#.....................................................................#
# ABORDAGEM 1: espaço de características dependente de D e H          #
#.....................................................................#

# Grade de hiperparâmetros
tuneGridANN <- expand.grid(.size = seq(1,11,1),
                           .decay = seq(0.001, 0.01, 0.001))

# Treinamento de modelos
set.seed(1000)
ANNTune1 <- train(lnVR~.,
                  data=trainingSet[,-"Esp"],
                  method="nnet",
                  trControl=fitControl,
                  tuneGrid=tuneGridANN,
                  maxit=1000,
                  preProcess=c("center","scale","BoxCox"),
                  trace=FALSE, linout=TRUE)

#.....................................................................#
# ABORDAGEM 2: espaço de características dependente apenas do D       #
#.....................................................................#

# Treinamento de modelos
set.seed(1000)
ANNTune2 <- train(lnVR~.,
                  data=trainingSet[,c(2,4:7)],
                  method="nnet",
                  trControl=fitControl,
                  tuneGrid=tuneGridANN,
                  maxit=1000,
                  preProcess=c("center","scale","BoxCox"),
                  trace=FALSE, linout=TRUE)

#########################################################################
#             ALGORITMO 5: Stochastic Gradient Boosting - SGB           #
#########################################################################

#.....................................................................#
# ABORDAGEM 1: espaço de características dependente de D e H          #
#.....................................................................#
```

```
229
230 # Grade de hiperparâmetros
231 tuneGridSGB <- expand.grid(interaction.depth=1:5,
232                            shrinkage=seq(0.001,0.1,0.01),
233                            n.trees=c(100,200,300,500,700),
234                            n.minobsinnode=c(5,10,20,30,50))
235
236 # Treinamento de modelos
237 set.seed(1000)
238 sgbTune1 <- train(lnVR~.,
239                   data=trainingSet[,-"Esp"],
240                   trControl=fitControl,
241                   tuneGrid=tuneGridSGB,
242                   distribution="gaussian",
243                   method="gbm",
244                   verbose=TRUE,
245                   preProc=c("center","scale","BoxCox"))
246
247 #.....................................................................#
248 # ABORDAGEM 2: espaço de características dependente apenas do D       #
249 #.....................................................................#
250
251 set.seed(1000)
252 sgbTune2 <- train(lnVR~.,
253                   data=trainingSet[,c(2,4:7)],
254                   trControl=fitControl,
255                   tuneGrid=tuneGridSGB,
256                   distribution="gaussian",
257                   method="gbm",
258                   verbose=TRUE,
259                   preProc=c("center","scale","BoxCox"))
260
261 ######################################################################
262 #           ALGORITMO 6: Suporte Vector Regression - SVR             #
263 ######################################################################
264
265 #.....................................................................#
266 # ABORDAGEM 1: espaço de características dependente de D e H          #
267 #.....................................................................#
268
269 # Grade de hiperparâmetros
270 tuneGridSVR <- expand.grid(sigma= seq(0,.01,0.001),
271                            C= 2^c(4,5,6,7))
272
273 # Treinamento de modelos
274 set.seed(1000)
275 svrTune1 <- train(lnVR~.,
276                   data=trainingSet[,-"Esp"],
277                   method='svmRadial',
278                   trControl=fitControl,
279                   tuneGrid=tuneGridSVR,
280                   preProcess=c("center","scale","BoxCox"))
281
282 #.....................................................................#
283 # ABORDAGEM 2: espaço de características dependente apenas do D       #
284 #.....................................................................#
285
286 # Treinamento de modelos
287 set.seed(1000)
```

```
svrTune2 <- train(lnVR~.,
                  data=trainingSet[,c(2,4:7)],
                  method='svmRadial',
                  trControl=fitControl,
                  tuneGrid=tuneGridSVR,
                  preProcess=c("center","scale","BoxCox"))

#########################################################################
#         ALGORITMO 7: Extreme Gradient Boosting - XGBoost              #
#########################################################################

#.....................................................................#
# ABORDAGEM 1: espaço de características dependente de D e H          #
#.....................................................................#

# Grade de hiperparâmetros
tuneGridXGB <- expand.grid(nrounds=seq(200,1000,50),
                           eta=seq(0.01,0.07,0.01),
                           max_depth=1:6,
                           gamma=0,
                           colsample_bytree=seq(0.7,1,0.1),
                           min_child_weight=seq(5,50,5),
                           subsample=seq(0.7,1,0.1))

# Treinamento de modelos
set.seed(1000)
xgbTune1 <- train(lnVR~.,
                  data=trainingSet[,-"Esp"],
                  trControl=fitControl,
                  tuneGrid=tuneGridXGB,
                  method="xgbTree",
                  verbose=TRUE,
                  nthread=6)

#.....................................................................#
# ABORDAGEM 2: espaço de características dependente apenas do D       #
#.....................................................................#

# Treinamento de modelos
set.seed(1000)
xgbTune2 <- train(lnVR~.,
                  data=trainingSet[,c(2,4:7)],
                  trControl=fitControl,
                  tuneGrid=tuneGridXGB,
                  method="xgbTree",
                  verbose=TRUE,
                  nthread=6)

#########################################################################
#                  ALGORITMO 8: Random Forest - RF                      #
#########################################################################

# Customiza "Random Forest" para admitir hiperparâmetro "ntree"
#-----------------------------------------------------------------------
customRF <- list(type="Regression",library="randomForest",loop=NULL)
customRF$parameters <- data.frame(parameter=c("mtry","ntree"),class=rep("
    ↪ numeric",2),label=c("mtry", "ntree"))
customRF$grid <- function(x,y,len=NULL,search="grid"){}
customRF$fit <- function(x,y,wts,param,lev,last,weights,classProbs, ...){
```

```
346   randomForest(x,y,mtry=param$mtry,ntree=param$ntree, ...)
347 }
348 customRF$predict <- function(modelFit,newdata,preProc=NULL,submodels=NULL)
349   predict(modelFit,newdata)
350 customRF$prob <- function(modelFit,newdata,preProc=NULL,submodels=NULL)
351   predict(modelFit,newdata,type="prob")
352 customRF$sort <- function(x) x[order(x[,1]),]
353 customRF$levels <- function(x) x$classes
354 #-----------------------------------------------------------------------
355
356 #.....................................................................#
357 # ABORDAGEM 1: espaço de características dependente de D e H          #
358 #.....................................................................#
359
360 # Treinamento de modelos
361 set.seed(1000)
362 rfTune1 <- train(lnVR~.,
363                  data=trainingSet[,-"Esp"],
364                  method=customRF,
365                  tuneGrid=expand.grid(mtry=c(1:11),
366                                       ntree=seq(50,500,50)),
367                  trControl=fitControl,
368                  importance=TRUE)
369
370 #.....................................................................#
371 # ABORDAGEM 2: espaço de características dependente apenas do D       #
372 #.....................................................................#
373
374 # Treinamento de modelos
375 set.seed(1000)
376 rfTune2 <- train(lnVR~.,
377                  data=trainingSet[,c(2,4:7)],
378                  method=customRF,
379                  tuneGrid=expand.grid(mtry=c(1:4),
380                                       ntree=seq(50,500,50)),
381                  trControl=fitControl,
382                  importance=TRUE)
383
384 ########################################################################
385 #         ALGORITMO 9: Weighted k-Nearest-Neighbor - wkNN              #
386 ########################################################################
387
388 #.....................................................................#
389 # ABORDAGEM 1: espaço de características dependente de D e H          #
390 #.....................................................................#
391
392 # Grade de hiperparâmetros
393 tuneGridWKNN <- expand.grid(kmax=seq(2,25,2),
394                             kernel=c("rectangular","biweight","cos",
395                                      "epanechnikov","gaussian","inv",
396                                      "optimal","rank","triangular",
397                                      "triweight"),
398                             distance=1:3)
399
400 # Treinamento de modelos
401 set.seed(1000)
402 wkNNTune1 <- train(lnVR~.,
403                    data=trainingSet[,-"Esp"],
404                    trControl=fitControl,
```

```
405                     tuneGrid=tuneGridWKNN,
406                     method="kknn",
407                     preProcess=c("center","scale","BoxCox"))
408 
409 #.....................................................................#
410 # ABORDAGEM 2: espaço de características dependente apenas do D       #
411 #.....................................................................#
412 
413 # Treinamento de modelos
414 set.seed(1000)
415 wkNNTune2 <- train(lnVR~.,
416                     data=trainingSet[,c(2,4:7)],
417                     trControl=fitControl,
418                     tuneGrid=tuneGridWKNN,
419                     method="kknn",
420                     preProcess=c("center","scale","BoxCox"))
421 
422 # Fim
423 #.....................................................................#
```