Consenso Monte Carlo em modelos BART: priori, agregação e predição.

Lucas Tavares Short Cabral

Dissertação apresentada
ao
Instituto de Matemática e Estatística
da
Universidade de São Paulo
para
obtenção do título
de
Mestre em Ciências

Programa: Estatística

Orientador: Prof. PhD Hedibert Freitas Lopes

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro do CNPq

São Paulo, fevereiro de 2018

Consenso Monte Carlo em modelos BART: priori, agregação e predição

Esta é a versão original da dissertação elaborada pelo candidato Lucas Tavares Short Cabral, tal como submetida à Comissão Julgadora.

Consenso Monte Carlo
em modelos BART:
priori, agregação e predição

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 28/02/2018. Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão Julgadora:

- Prof. PhD Hedibert Freitas Lopes (orientador) - INSPER
- Prof. Drł. Florencia Graciela Leonardi - IME-USP
- Prof. Dr. Rinaldo Artes - INSPER

Resumo

CABRAL, L. T. S. **Consenso Monte Carlo em modelos BART: priori, agregação e predição**. 2018. 81 f. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2018.

Esse trabalho visa estudar em um contexto de *Big Data* o Bayesian Additive Regression Tree (BART) quando amostrado por um *Consensus* Monte Carlo (CMC). O BART é um modelo de regressão não-paramétrica que utiliza da soma várias árvores binárias com profundidade regularizadas, via priori, para construir funções preditivas. Já o CMC é uma maneira de combinar amostras de Monte Carlo de diferentes computadores (ou de partições dos dados) e gerar uma aproximação da posteriori dos dados completos. Os objetivos do trabalho são estudar o comportamento das prioris nesse contexto em relação a capacidade de predição do modelo e verificar como se comporta seleção de variáveis do BART em um CMC. Para atingir o objetivo propõem-se uma nova maneira de agregar os resultados dos diferentes BART's no CMC usando a correlação de Pearson combinada a variância das predições realizadas por cada BART individual como peso. Os resultados mostram que o CMC do modelo BART é capaz de selecionar e predizer tão bem quanto o BART, com maior escalabilidade.

Palavras-chave: Consenso Monte Carlo, BART, CMC-BART.

Abstract

CABRAL, L. T. S.**Consensus Monte Carlo on BART models: prior, aggregation and prediction**. 2018. 81 f. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2018.

This work study the Bayesian Additive Regression Tree (BART) when sampled by a Consensus Monte Carlo. BART is a non-parametric regression model that sum the outcome of many binary trees with regularized depth, via prior, to make predictions. On the other hand, CMC is a method to aggregate MCMC samples from different computers (or data shards) to generate an approximation to the full data posterior. The objectives for this work are: study the impact of the priors choice in prediction of BART with CMC, and it's capability of variable selection. To accomplish the objective we suggest a new way to aggregate the consensus results from each BART in a CMC. This new aggregation method is based on Pearson's correlation and the variance of the predictions to generate the weights. The results shows that CMC-BART is able to select variable and make prediction as good as BART, but with more scalability.

Keywords: Consensus Monte Carlo, BART, CMC-BART.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Modelos de regressão via árvores

Leo Breiman *et al.* [BFOS84] lança a ideia dos modelos de regressão via uma única árvore. Uma árvore de regressão é representada por uma árvore binária, como ilustra a Figura 1.1. Em cada nó da árvore, uma variável x_i do espaço das variáveis é escolhida; em seguida, é feita a escolha de um ponto de corte associado à variável x_i. A partir de então, todas as observações que são menores que x_i no ponto de corte selecionado seguem à esquerda na árvore; caso contrário, elas vão para a direita, particionando, assim, o espaço das variáveis. Tal processo ocorre até que se encontre uma folha, que é o resultado da regressão, ou seja, o $f(x)$.

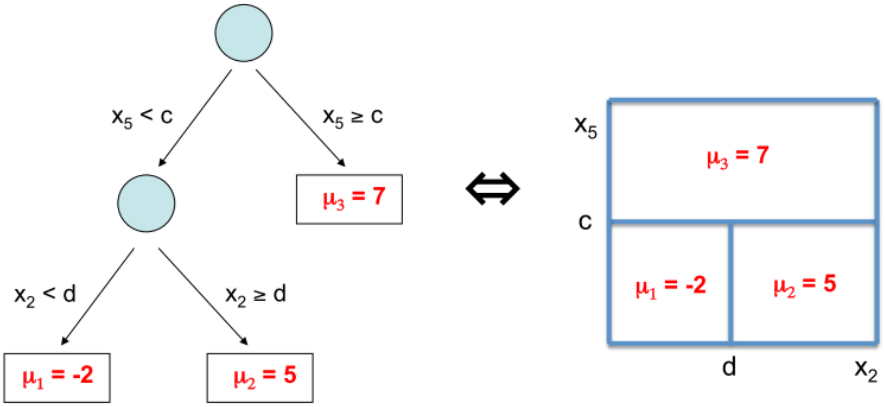


Figura 1.1: *Exemplo de uma árvore de regressão.*

As regressões via única árvore têm algumas vantagens: modelam bem não-linearidades, captam efeitos de interação entre as variáveis, têm capacidade de seleção de variáveis, são de fácil interpretação quando a árvore é pequena, possuem poucas suposições necessárias, não sofrem com o aumento do número de váriaveis, não necessitam de inversão de matrizes. Hastie *et al.* [HTF01] ainda cita como vantagens dos modelos de árvore o fato de serem invariantes sob escala e várias outras transformações das variáveis, robustez a inclusão de variáveis explicativas irrelevantes e modelos inspecionáveis.

No entanto, os autores dizem ainda que os modelos de árvore raramente são precisos. Outras desvantagens associadas a esses modelos são: instabilidade nas variáveis utilizadas quando são adicionadas novas ou retiradas variáveis relevantes ao modelo, e dificuldade de interpretação quando a árvore torna-se muito profunda ou complexa. Esta última desvantagem acaba por gerar um padrão irregular de predição fora-da-amostra, ela pode ser um efeito de sobreajuste do conjunto de treino.

Com o passar dos anos, surgiram modelos de regressão via múltiplas árvores, que possuem melhor qualidade de ajuste aos dados e maior capacidade de preditiva. Lakshminarayanan *et al.* [LRT15]

dividem tais modelos em duas grandes famílias: árvores de regressão independentes aleatórias, onde as árvores crescem independentemente umas das outras e as predições são realizadas pela média entre elas, com o objetivo de minimizar a variância; e as árvores de regressão aditivas, onde cada árvore ajusta o resíduo não explicado pelo restante das árvores [LRT15].

Exemplos de modelos na primeira categoria são *Bagged Decision Trees* [Bre96]; *Random Forests* [Bre01]; *Extremely Randomized Trees* [GEW06]; *Relevant Ensemble of Trees* [DB17]. Já para a segunda categoria, temos *Multiple Additive Regression Trees*, também conhecido como MART [Fri01, Fri02]; DART, que consiste num MART com *dropouts* [VGB15]; o *Bayesian Addtive Regression Trees* [CGM10]; e o *Additive Groves* [SCR07]. As regressões via múltiplas árvores melhoram as vantagens dos modelos com única árvore em sacrifício da interpretabilidade dos modelos.

Os modelos de regressão via árvores podem ser estendidos de forma a flexibilizar a relação entre y_i e a sua média condicional com função dos regressores $x_{i1}, \ldots, x_{ip}$, isto é,

$$y_i = f(x_{i1}, \ldots, x_{ip}) + \varepsilon_i \tag{1.1}$$

onde f pode ser: a Regressão Linear Múltipla, a Regressão Logística, ou modelos de regressão não paramétrica (Wavelets, Splines, Lowess), e ainda modelos de aprendizado de máquina: Máquina de Vetores de Suporte (SVM, na sigla em inglês) e Redes Neurais.

1.2 *Ensemble* de árvores

O *ensemble* tem como objetivo criar uma regressão altamente precisa através da combinação de outras menos precisas. Dietterich [Die00] cita pelo menos três razões fundamentais para a construção de bons *ensembles* a partir de regressões, sendo elas: uma razão estatística, uma razão computacional e uma razão representacional.

Para o autor, a razão estatística diz respeito ao fato de que a estimação dos parâmetros pode ser vista como uma busca no espaço $\mathcal{H}$ das hipóteses. O problema estatístico surge quando a quantidade de dados é pequena, principalmente ao ser comparada com o tamanho do espaço das hipóteses. Sem dados suficientes, o aprendizado pode levar a diversas hipóteses em $\mathcal{H}$ que dão a mesma acurácia no conjunto de treino. Ponderando-se as hipóteses com maior acurácia pode-se encontrar uma boa aproximação da verdadeira hipótese f.

Já a razão computacional é a de que muitos algoritmos possuem alguma forma de busca de máximos locais no espaço paramétrico. Essa busca pode levar a estimação a ficar presa em um mínimo local. Um *ensemble* de diversas regressões partindo de pontos iniciais distintos pode gerar uma melhor aproximação da função f.

A razão representacional, por sua vez, concerne a ideia de que em muitas aplicações a verdadeira função f não se pode ser representada por nenhuma das hipóteses no espaço $\mathcal{H}$. Formando um *ensemble* das hipóteses retiradas de $\mathcal{H}$ é possível expandir o espaço das funções representáveis.

Por fim, Dietterich [Die00] conclui que essas são três as razões fundamentais pelas quais os algoritmos de aprendizado falham. Assim, os métodos de *ensemble* podem solucionar a deficiência dos algoritmos de aprendizado tradicional. Os algoritmos mais populares de *ensemble* são: o *Boosting* [FS96], *Bagging* [Bre96] e o *Random Forests* [Bre01].

No contexto dos *ensembles* de árvores, cada uma delas representa uma hipótese retirada do espaço $\mathcal{H}$. A forma como a regressão estima e combina as árvores determina de qual das duas famílias propostas por Lakshminarayanan *et al.* [LRT15] a regressão pertence. Caso a estimação de cada árvore seja realizada de forma independente, então trata-se da família das árvores de regressão independentes aleatórias, que é um *ensemble* das árvores.

1.2.1 *Boosting*, *Bagging* e *Random Forests*

Nesta seção, será explicado como funcionam os três principais algoritmos de *ensemble* de árvores de classificação.

O *Boosting* [FS96] é um algoritmo sequencial de geração pesos que realiza a agregação de vários classificadores mais fracos. O algoritmo é inicializado com pesos iguais para todos os elementos da amostra. Um algoritmo de classificação simples gera uma hipótese de classificação. Essa hipótese prediz as classes baseadas na amostra e nos pesos atuais. Então, os erros produzidos pela hipótese são avaliados, e os pesos são recalculados baseados no desempenho dessa hipótese. Onde o classificador erra, os pesos são aumentados; onde ele acerta, os pesos são diminuídos. Assim, a nova distribuição de peso é normalizada. Esse processo de repete por T vezes. A hipótese final é uma votação ponderada de todas as hipóteses, onde o maior peso é dado às hipóteses com menor erro. O algoritmo de *Booting* é conhecido como AdaBoost e ganhou o Prêmio Gödel em 2003. O Prêmio Gödel é um prêmio anual para excelentes trabalhos na área de ciência da computação teórica, dado pela *European Association for Theoretical Computer Science* (EATCS) e pela *Association for Computing Machinery Special Interest Group on Algorithms and Computational Theory* (ACM SIGACT) [con17].

O *Bagging* [Bre96], um acrônimo de ***B**oostrap **Agg**regat**ing***, é um método de *ensemble* baseado em um *bootstrap* das amostras. Considere um conjunto de dados de treino $\mathcal{L}$, $(y_1, x_1), \ldots, (y_N, x_N)$, onde os $y_i \, \forall i$ são categorias ou respostas numéricas. Amostra-se aleatoriamente uma sequência de amostras *bootstrap* de treino $\{\mathcal{L}_k\}$, onde cada $\mathcal{L}_k$ é uma amostra aleatória de $\mathcal{L}$ com reposição e tamanho N. Usando-se essa sequência de conjuntos de treino, é gerada uma sequência de hipóteses de classificação, através de um classificador. A agregação da sequência de classificadores é feita pela média dos resultados, caso y seja númerico, ou pela mediana, caso y seja uma classe. O *Bagging* atinge melhores resultados quando pequenas alterações em $\mathcal{L}$ geram grandes alterações no resultado do classificador. Breiman [Bre96] chama esse fenômeno de instabilidade.

Segundo Hastie *et al.* [HTF01], o *Boosting* tem melhor desempenho que o *Bagging* para a maioria dos problemas. O *Random Forests* é uma evolução do *Bagging*; naquele, é construída uma grande coleção de árvores *decorrelacionadas*, e então, cada árvore gera uma resposta. A média das respostas é o resultado final da *Random Forests*. A decorrelação é o processo de reduzir a autocorrelação ou a correlação-cruzada entre as árvores. O *Random Forests*, por ser mais simples de treinar e afinar, tornou-se mais popular que o *Bagging*. No entanto, a ideia geral de *Bagging* pode ser utilizada para os mais diversos tipos de classificadores.

O *Random Forests* [Bre01] também utiliza metódos de *bootstrap* para gerar amostras. Gera-se uma sequência de amostras *bootstrap* do conjunto de treino. Para cada amostra é criada uma árvore aleatória (*random tree*) até atingir o número mínimo de nós $n_{\min}$. A criação de cada árvore é realizada repetindo-se três passos: (i) escolhe-se uma subconjunto de m variáveis das p variáveis; (ii) seleciona-se a melhor variável e o ponto de corte dentre as variáveis do subconjunto do passo (i); e (iii) divide-se o nó atual em dois nós filhos. Dessa maneira, é gerada uma sequência de classificadores baseados em árvores $\{T_b\}_1^B$. Para a realização de predições com um novo ponto x, existem duas situações: caso seja uma regressão, é feita a média do resultado de cada árvore; caso seja um problema de classificação, é escolhida a moda dos resultados de classificação. O número de variáveis escolhidas no passo (i) reduz a correlação entre as árvores do *ensemble*, assim reduzindo a variância da média do *Random Forests*.

1.3 BART e variações

O uso dos métodos de *ensemble* de árvores, em geral, aumentam a taxa de acerto na predição, em detrimento da utilização de uma única árvore. Essa estratégia também foi levada para os modelos bayesianos com o modelo *Bayesian Additive Regression Tree* (BART).

O BART é um modelo de regressão não-parâmetrica Bayesiana criado por Chipman, George e McCulloch [CGM10]. Ele é baseado em uma soma de várias árvores, sendo que cada uma tem seu tamanho restringido por meio de prioris regularizadoras. O ajuste e a inferência do modelo são realizados por meio de um MCMC com *backfitting*, que é o processo de amostragem baseado nos resíduos parciais. O BART ainda pode ser utilizado como método para seleção de variáveis sem uso de modelo, em inglês conhecida como *model free variable selection*. No trabalho mencionado, os autores ainda citam o modelo Probit BART para classificação.

Ao longo dos anos, foram criadas variações do modelo BART. A primeira delas foi o *Bayesian Additive Classification Tree* [ZH10], criado para dados de classificação utilizando do Modelo Aditivo Generalizado de [HT90] e o esquema de aumento dos dados de Tanner & Wang [TW87]. Alguns anos depois surgiram duas novas variações para extender a classificação para múltiplas classes: o *One-vs-All* BART [ARC14] e o MPBART [KWP16]. Surgiram também modelos BART que lidam com heterocedasticidade, a saber: [BK14] e [PCGM17], além de uma versão log-linear do BART para dados categóricos e de contagem [Mur17], outra versão para regressão quantílica [KWHP16] e outra que usa uma reparametrização para adaptarção do Probit BART para dados de sobrevivência [LMLS16].

Por fim, uma das variações é realizar um Consenso Monte Carlo (CMC), criado por Scott *et al.* [SBB$^+$16], com o BART, neste trabalho denominado CMC-BART. O CMC é um algoritmo de computação da posteriori de forma paralela entre vários computadores ou núcleos de CPUs. Tal computação paralela ocorre por meio da divisão dos dados entre alguns computadores (ou núcleos de um mesmo computador), onde cada um deles calcula uma subposteriori e então elas são agregadas em uma única distribuição preditiva.

1.4 Objetivos

Dada uma situação onde se tem um grande volume de dados, a ponto de não ser possível utilizar o BART em apenas um único computador para realizar predições, é preciso utilizar um CMC-BART. Este trabalho tem como objetivo melhorar a escolha das prioris do CMC-BART para diferentes volumes de dados e propor um novo método de agregação para o CMC. Os objetivos específicos são:

- verificar como se comporta seleção de variáveis do CMC-BART;
- estudar o impacto da escolha dos hiperparâmetros das prioris do CMC-BART para diferentes volumes de dados na capacidade preditiva do modelo;
- analisar quais métodos de agregação das amostras geram melhores predições.

1.5 Organização do Trabalho

No Capítulo 2, apresenta-se: o BART, definindo suas prioris, bem como é realizada a amostragem do modelo; o CMC, como é feita a divisão da amostra entre as diferentes máquinas e a agregação dos resultados, também são apresentados alguns exemplos; e por fim a combinação do CMC com o BART, isto é, o CMC-BART. No Capítulo 3 é exposto o resultado das simulações em diferentes conjuntos de dados e analisa-se os resultados, as simulações foram realizadas com dados sínteticos e dados reais, para os dados reais foram trabalhados dados de classificação e regressão. Finalmente, no Capítulo 4 discute-se os objetivos atingidos neste trabalho e são propostas novas frentes de pesquisa.

Os códigos utilizados estão disponíveis no Apêndice A.

Capítulo 2

Conceitos

Neste capítulo serão definidos o que são Consenso Monte Carlo e BART, suas prioris e o processo de amostragem desse modelo.

2.1 *Bayesian Additive Regression Tree*

O BART é um modelo de regressão aditivo com componentes multivariados, também conhecido como modelo de soma de árvores. Consideremos, tal como Chipman *et al.* em [CGM10], um problema de inferência onde a função f prediz a variável resposta Y usando um vetor p-dimensional $x = (x_1, \ldots, x_p)$ onde

$$Y = f(x) + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2). \tag{2.1}$$

Para tanto, aproxima-se $f(x) = E(Y|x)$ por uma soma de m regressões de árvore $f(x) \approx h(x) = \sum_{j=1}^{m} g_j(x)$, onde cada g_j denota uma regressão de árvore. Cada regressão de árvore é regularizada por uma priori a fim de manter os seus efeitos pequenos. Com isso, cada uma das árvores g_j explica uma pequena e diferente parte da f. Note que o BART não é equivalente a uma média a posteriori de ajustes de várias árvores individuais de toda a função f.

Agora serão definidas as notações para um modelo de árvore única e depois elas serão estendidas para um modelo de soma de árvores. Seja T uma árvore binária com profundidade d, onde cada um dos seus nós internos está associado a uma regra de divisão e cada uma de suas folhas está associada a uma entrada no vetor $M = \{\mu_1, \mu_2, \ldots, \mu_b\}$. As regras de divisão são separações binárias do espaço preditor da forma $\{x \in A\}$ e $\{x \notin A\}$ onde A é um subconjunto da imagem de x. Cada separação é baseada em uma única componente do $x = (x_1, x_2, \ldots, x_p)$ da forma $\{x_i \leq c\}$ ou $\{x_i > c\}$ para x_i contínuo. Cada valor de x é associado a uma única folha de T por conta da sequência das decisões de cada nó interno. Dados T e M, $g(x;\, T, M)$ será usado para denotar a função que associa $\mu_i \in M$ ao x. Assim,

$$Y = g(x;\, T, M) + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2) \tag{2.2}$$

Sob essa notação podemos estender o modelo de árvore única (2.2) para o modelo de soma de árvores como

$$Y = \sum_{j=1}^{m} g(x;\, T_j, M_j) + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2), \tag{2.3}$$

onde para cada árvore T_j está relacionada ao vetor M_j de suas folhas. Já $g(x;\, T_j, M_j)$ é a função que associa $\mu_{ij} \in M_j$ a x. Sob (2.3), a $E(Y|x)$ é igual à soma de todas as folhas μ_{ij}'s associadas a x pelas $g(x;\, T_j, M_j)$'s. Quando o número de árvores $m > 1$, cada μ_{ij} é uma mera parte da $E(Y|x)$, diferentemente do modelo de árvore única 2.2. Além disso, cada μ_{ij} representa um efeito principal quando $g(x;\, T_j, M_j)$ depende apenas de uma componente de x, isto é, de uma única variável preditora. Tem-se ainda um efeito de interação quando $g(x;\, T_j, M_j)$ depende de mais de uma componente de x. Sendo assim, o modelo de soma de árvores pode incorporar tanto efeitos principais quanto de interação, dependendo do número de componentes utilizadas em cada árvore.

2.1.1 Prioris

Chipman *et al.* apresentam as prioris para o modelo de uma única máquina [CGM10]. Nesta seção, tais prioris [CGM10] serão mostradas. A prioris são definidas para os parâmetros $(T_1, M_1), \ldots,$ (T_m, M_m) e σ e os autores ressaltam a importância de essas prioris serem regularizadoras com o objetivo de evitar que o efeito individual de cada árvore seja excessivamente influente. Sem esse tipo de regularização, os autores advogam que haveria grandes componentes de árvores que prevaleceriam sobre a rica estrutura de 2.3, com isso limitando ambas as vantagens da representação adivitiva: aproximação funcional e facilidade computacional.

Independência das prioris

A especificação das prioris é simplificada ao se restringir o conjunto das prioris nas quais

$$p((T_1, M_1), \ldots, (T_m, M_m), \sigma) = \Big[\prod_j p(T_j, M_j)\Big] p(\sigma) \tag{2.4}$$

$$= \Big[\prod_j p(M_j|T_j) p(T_j)\Big] p(\sigma) \tag{2.5}$$

e

$$p(M_j|T_j) = \prod_i p(\mu_{ij}|T_j), \tag{2.6}$$

onde $\mu_{ij} \in M_j$. Sob essas prioris, os componentes das árvores (T_j, M_j) são independentes entre si e de σ, e ainda, os parâmetros dos nós terminais de cada árvore são independentes.

A simplificação da escolha das prioris sob a restrição de independência ocorre por conta da necessidade de se ter que escolher prioris apenas para $p(T_j)$, $p(\mu_{ij}|T_j)$ e $p(\sigma)$. Os autores ainda tomam o cuidado de escolher prioris com poucos hiperparâmetros, e que sejam interpretáveis. No entanto, a escolha desses hiperparâmetros para as prioris $p(\mu_{ij}|T_j)$ e $p(\sigma)$ é feita por meio dos dados[1], sacrificando coerência em prol de facilidade computacional.

A priori para T_j

A priori $p(T_j)$ é a que controla a forma da árvore durante o processo de amostragem. Ela é especificada por três aspectos: (i) a probabilidade de que um nó de profundidade $d(= 0, 1, 2, \ldots)$ é não terminal é dada por

$$\alpha(1 + d)^{-\beta}, \qquad \alpha \in (0, 1) \text{ e } \beta \in [0, \infty). \tag{2.7}$$

α é um valor de referência para as probabilidades e β controla a profundidade das árvores. (ii) a distribuição da variável de separação binária de cada nó interno, e (iii) a distribuição da regra de separação designada a cada nó interno. Em (ii) a priori é uma distribuição uniforme entre as variáveis disponíveis; já em (iii) a priori também é uma uniforme discreta em um conjunto de possíveis valores de separação. Os autores ainda argumentam que essa abordagem da priori (iii) não é coerente com a teoria Bayesiana; porém, ela tem a vantagem de gerar invariância sob transformações monotônicas para as variáveis de divisão.

A escolha dos hiperparâmetros é feita de forma que individualmente cada árvore tenha poucos componentes, ou seja, não seja muito profunda. Partindo dessa ideia, os autores sugerem $\alpha = 0,95$ e $\beta = 2$ em 2.7. Note que com essa escolha de prioris as árvores com 1, 2, 3, 4, ou ≥ 5 nós terminais recebem probabilidade a priori iguais a 0,05; 0,55; 0,28; 0,09 e 0,03, respectivamente.

A priori para $\mu_{ij}|T_j$

A distribuição de $p(\mu_{ij}|T_j)$ controla os valores de cada folha. Para $p(\mu_{ij}|T_j)$, a priori escolhida é uma normal conjugada $N(\mu_\mu, \sigma_\mu^2)$ por conta do benefício computacional, pois μ_{ij} pode ser margi-

[1] Essa estratégia é conhecida como *Empirical Bayes*

nalizado.

A especificação dos hiperâmetros μ_μ e σ_μ^2 dessa priori é feita por meio dos dados. Observando que $E(Y|x)$ é a soma de m μ_{ij}'s no modelo de soma de árvores 2.3, logo a priori induzida de $E(Y|x)$ é $N(m\mu_\mu, m\sigma_\mu^2)$. A estratégia definida por Chipman *et al.* [CGM10] é escolher μ_μ e σ_μ^2, de tal forma que $N(m\mu_\mu, m\sigma_\mu^2)$ atribua densidade substancial ao intervalo $(y_{\min}, y_{\max})$. Para isso, é necessário resolver um sistema de equações para μ_μ e σ_μ^2 dado um k arbitrário,

$$\begin{cases} m\mu_\mu & - & k\sqrt{m}\sigma_\mu & = & y_{\min} \\ m\mu_\mu & + & k\sqrt{m}\sigma_\mu & = & y_{\max}. \end{cases}$$

Por exemplo, para $k = 2$ temos que $E(Y|x)$ fica no intervalo $(y_{\min}, y_{\max})$ com 95% de probabilidade a priori. Note que no sistema acima é usada a amostra para calcular $y_{\min}$ e $y_{\max}$. Isso é feito pelos autores a fim de distribuir a densidade da priori na região de valores plausíveis de $E(Y|x)$, evitando, assim, uma superconcentração e/ou sobredispersão da priori.

Chipman *et al.* [CGM10], por conveniência, fazem a implementação da priori de $\mu_{ij}|T_j$ utilizando uma transformação linear nos valores de Y, de forma que os valores alcance dos observados de y fiquem entre $y_{\min} = -0.5$ e $y_{\max} = 0.5$ [2]. A variável transformada é então usada como variável dependente. Centra-se a priori de $\mu_{ij} = 0$ e escolhe-se um σ_μ de tal forma que $k\sqrt{m}\sigma_\mu = 0.5$ pra um k arbitrário. A escolha de k entre 1 e 3 leva a bons resultados, e os autores recomendam $k = 2$. Alternativamente, o valor de k pode ser escolhido por validação cruzada, levando a

$$\mu_{ij} \sim N(0, \sigma_\mu^2), \qquad \text{onde } \sigma_\mu = 0.5/k\sqrt{m}. \tag{2.8}$$

Os autores afirmam que essa priori encolhe o efeito do parâmetro μ_{ij} de cada árvore em direção ao zero, limitando-se assim o efeito de cada árvore no modelo de várias árvores 2.3. Ao se aumentar o número de árvores m ou a constante k, a priori torna-se mais regularizadora, encolhendo ainda mais o efeito dos μ_{ij}'s.

Essas prioris sacrificam a coerência Bayesiana em troca de o modelo BART se tornar mais robusto. A priori informada pelos dados é útil para esse modelo, porque é difícil obter informação subjetiva confiável sobre a $E(Y|x)$.

A priori para σ

Para $p(\sigma)$, os autores também usam uma priori conjugada: a distribuição qui-quadrado invertida $\sigma^2 \sim \nu\lambda/\chi_\nu^2$. A escolha dos hiperparâmetros ν e λ é feita usando a abordagem de priori informada pelos dados. Nesse caso, atribui-se probabilidade substancial à região de valores plausíveis de σ enquanto evita-se superconcentração ou superdispersão. Os autores propõem calibrar os graus de liberdade ν e a escala λ usando uma superestimativa grosseira baseada nos dados de $\hat{\sigma}$ para σ. Dois possíveis métodos para escolher $\hat{\sigma}$ são: (1) toma-se o desvio padrão amostral de Y, ou (2) tomam-se os resíduos de uma regressão linear por mínimos quadrados de X em Y. Escolhe-se arbitrariamente o valor de ν entre 3 e 10, a fim de obter-se uma densidade com forma apropriada, e o valor de λ de forma que o q-ésimo quantil da priori de σ é localizada em $\hat{\sigma}$, isto é, $P(\sigma < \hat{\sigma}) = q$. Chipman *et al.* [CGM10] escolhem valores para q tais como 0,75; 0,90 ou 0,99 para que centro da distribuição fique abaixo de $\hat{\sigma}$.

Finalmente, os autores recomendam: (1) evitar valores de $\nu < 3$, pois isso gera uma concentração de densidade muito grande em valores pequenos de σ, levando a um sobreajuste; (2) para o uso automático das prioris, os hiperparâmetros $(\nu;\ q) = (3;\ 0{,}9)$; (3) alternativamente, os valores de $(\nu;\ q)$ podem ser escolhidos por meio de validação cruzada em um conjunto de valores razoáveis.

[2] Para realizar essa transformação basta resolver um sistema linear e aplicar a transformação aos valores observados de y.

A escolha do número de árvores m

Seria razoável tratar m como um parâmetro desconhecido, colocar uma priori em m e proceder com uma abordagem genuinamente Bayesiana para o BART. Outra estratégia seria selecionar o "melhor" valor para m por meio de validação cruzada. Todavia, ambas estratégias aumentam substancialmente o tempo computacional requerido.

Para evitar um alto custo computacional, os autores sugerem usar $m = 200$, e talvez testar uma ou duas outras opções para m e ver se surgem quaisquer diferenças. Neste trabalho, verificaremos se a escolha do número de árvores igual a 200 pode ser mantida no CMC-BART ou se outras opções seriam mais eficientes.

2.1.2 Amostragem do modelo BART

Esta seção apresenta o processo de amostragem para a inferência da posteriori do modelo BART. Dados os y, amostra-se a posteriori

$$p((T_1, M_1), \ldots, (T_m, M_m), \sigma | y) \tag{2.9}$$

com todos os parâmetros desconhecidos que determinam um modelo de soma de árvores 2.3. Para isso, é usado o algoritmo de MCMC com *backfitting*.

Em um nível geral, a amostragem do modelo BART ocorre por meio de um amostrador de Gibbs com passo de Metropolis-Hastings. Para notação conveniente, denota-se $T_{(j)}$ como a soma de todas as árvores, exceto a T_j, e analogamente define-se $M_{(j)}$. Dessa forma, $T_{(j)}$ tem $m-1$ árvores e $M_{(j)}$ são os nós terminais associados. O amostrador de Gibbs encarrega-se de gerar uma amostra de tamanho m de (T_j, M_j) condicionado a $(T_{(j)}, M_{(j)}, \sigma)$:

$$(T_j, M_j) | T_{(j)}, M_{(j)}, \sigma \tag{2.10}$$

$j = 1, \ldots, m$ seguido de amostrar σ a partir de sua condicional completa:

$$\sigma | T_1, \ldots, T_m, M_1, \ldots, M_m, y. \tag{2.11}$$

Hastie e Tibshirani [HT00] consideram um amostrador de Gibbs para modelos aditivos e modelos aditivos generalizados com σ fixo, que é uma generalização estocástica para o algorimto de *backfitting* para esses tipos de modelo. Por essa razão, Chipman *et al.* [CGM10] nomeiam o algoritmo como MCMC com *backfitting*.

A amostragem de σ em 2.11 consiste em amostrar da distribuição gama inversa. Para gerar as m amostras de (T_j, M_j) em 2.10 é importante tomar uma forma reduzida. Os autores notam que a distribuição condicional $p(T_j, M_j | T_{(j)}, M_{(j)}, \sigma, y)$ depende de $(T_{(j)}, M_{(j)}, y)$ apenas por meio de

$$R_j = y - \sum_{k \neq j} g(x; T_k, M_k) \tag{2.12}$$

Um vetor de n entradas dos resíduos baseados no ajuste que exclui a j-ésima árvore. Assim sendo, as m amostras de (T_j, M_j) dado $(T_{(j)}, M_{(j)}, \sigma, y)$ em 2.10 são equivalentes às m amostras de

$$(T_j, M_j) | R_j, \sigma, \qquad \text{onde } j = 1, \ldots, m. \tag{2.13}$$

Note que 2.13 é formalmente equivalente à posteriori do modelo de árvore única $R_j = g(x; T_j, M_j) + \varepsilon$ onde R_j faz o papel dos dados y. Por conta de os autores terem optado por uma priori conjugada para M_j,

$$p(T_j | R_j, \sigma) \propto p(T_j) \int p(R_j | M_j, T_j, \sigma)\, p(M_j | T_j, \sigma)\, dM_j \tag{2.14}$$

pode ser obtida em forma fechada a menos de constante normalizadora. Isso permite calcular cada

amostra de 2.13 em dois passos:

$$\begin{aligned} T_j|R_j,\sigma \text{ e} \\ M_j|T_j,R_j,\sigma. \end{aligned} \tag{2.15}$$

O amostrador para T_j é o passo de Metropolis-Hastings (MH) como em [CGM98]. Esse algoritmo propõe uma nova árvore baseada na árvore atual e em quatro possíveis movimentos: nascimento de um novo nó terminal (crescer), podar um par de nós terminais (podar), mudar a regra de divisão de um nó não terminal (mudar) e trocar a regra de divisão entre um nó pai e seu filho (permutar). Ainda que o nascimento de um novo nó e a poda de um par de nós terminais altere o número de nós terminais, devido à integração de M_j em 2.14, evita-se a complexidade associada aos saltos reversíveis (RJMCMC) entre os espaços contínuos de dimensão variável [Gre95].

O algoritmo MH para simular a cadeia $T_j^{(0)}, T_j^{(1)}, T_j^{(2)}, \ldots$ é definido a seguir. Inicializamos o processo com a árvore $T_j^{(0)}$, e iteradamente simulamos a transição de $T_j^{(i)}$ para $T_j^{(i+1)}$ seguindo os passos:

1. Gerar uma candidato T^* com probabilidade de transição $q(T^i, T^*)$;
2. Escolha $T^{i+1} = T^*$ com probabilidade

$$\alpha(T^i, T^*) = \min\left\{\frac{q(T^*, T^i)}{q(T^i, T^*)}\frac{p(Y|X,T^*)\,p(T^*)}{p(Y|X,T^i)\,p(T^i)},\ 1\right\}$$

   Caso contrário, $T^{i+1} = T^i$.

O núcleo de transição q é definido pelos possíveis movimentos da árvore. Considere o núcleo $q(T, T^*)$, o qual gera T^* a partir de T escolhendo aleatoriamente um dos quatro passos:

- *Crescer:* Escolher aleatoriamente um nó terminal. Dividi-lo em dois novos nós atribuindo aleatoriamente uma variável e uma regra de divisão.
- *Podar:* Escolher aleatoriamente o pai de um par de nós terminais e torná-lo um nó terminal, colapsando os nós abaixo dele.
- *Mudar:* Escolher aleatoriamente um nó interno e sortear uma nova regra de divisão com uma nova variável.
- *Permutar:* Escolher aleatoriamente um par pai-filho de nós internos. Permutar suas regras de divisão, a não ser que outro filho possua uma regra idêntica; nesse caso, a permuta da regra de divisão é entre o pai e todos filhos que seguem a mesma regra.

Para cada um desses movimentos associam-se as seguintes probabilidades: Crescer (0,25); Podar (0,25), Mudar (0,40) e Permutar (0,10). Assim, realiza-se a amostragem de $T_j|R_j,\sigma$.

Já a amostragem de $M_j|T_j,R_j,\sigma$ em 2.15 é realizada de forma direta; amostra-se de maneira independente da distribuição normal para cada μ_{ij}. A amostra de M_j é importante para o cálculo dos resíduos R_k, onde $k \neq j$ é crucial para o amostra seguinte de T_j. Os autores ainda afirmam que felizmente não é necessária uma implementação mais complexa usando saltos reversíveis [CGM10].

A cadeia é inicializada com m árvores com um único nó, e então as iterações são realizadas até que uma convergência satisfatória seja atingida. A cada iteração, cada árvore pode ganhar ou perder um nó terminal ou mudar uma, ou duas, regras de decisão. Cada μ irá mudar (nascer ou morrer) e σ mudará também. Não é incomum para uma árvore crescer muito no início do processo de amostragem e no fim colapsar a uma árvore de apenas um nó.

Comparado ao processo de amostragem do CART [BFOS84], a amostragem do BART por MCMC com *backfitting* tem melhor *mixing*. O MCMC com *backfitting* é resumido no algoritmo 1.

Algoritmo 1: MCMC com *backfitting* para a inferência da posteriori do modelo BART

Input : Dados (y, x), número de iterações N e os hiperpârametros $(\nu, q, k, m, \alpha, \beta)$
Initialization: $\forall j$, inicializar $T_j^{(0)}$ com um nó
Output : Amostra dos parâmetros do modelo BART

for $i = 1 : N$ **do**
  Amostrar $\sigma^{(i)}$
  for $j = 1 : m$ **do**
    Calcular $R_j = y - \sum_{k \neq j} g(x; T_k, M_k)$
    Amostrar $T_j^{(i)} | R_j, \sigma^{(i)}$
    Amostrar $M_j^{(i)} | T_j^{(i)}, R_j, \sigma^{(i)}$,
  end
end

2.1.3 Estatísticas da posteriori

Após a geração de uma sequência de amostras de $(T_1, M_1), \ldots, (T_m, M_m)), \sigma$ que converge em distribuição para a posteriori $p((T_1, M_1), \ldots, (T_m, M_m)), \sigma | y)$, usando o algoritmo de MCMC com *backfitting*, deseja-se realizar inferências sobre a posteriori. As amostras de $(T_1, M_1), \ldots, (T_m, M_m))$ induzem uma sequência das funções de soma de árvores

$$f^*(\cdot) = \sum_{j=1}^{m} g(\cdot, T_j^*, M_j^*), \tag{2.16}$$

que converge desta maneira para $p(f|y)$, a distribuição a posteriori da "verdadeira" $f(\cdot)$. Assim, após execução do algoritmo por um número de passos suficiente, ou seja, após um período razoável de aquecimento da cadeia, a sequência de amostras de f^* (digamos $f_1^*, \ldots, f_K^*$) é uma amostra com dependência de tamanho K da $p(f|y)$.

Para estimar $f(x)$ ou predizer Y em um x em particular, dentro ou fora da amostra, uma escolha natural é a média, após o *burn-in*, de $f_1^*, \ldots, f_K^*$, ou seja,

$$\widehat{f(x)} = \frac{1}{K} \sum_{k=1}^{K} f_k^*(x), \tag{2.17}$$

a qual aproxima-se da média da posteriori $E(f(x)|y)$. Outra boa opção seria a mediana de $f_1^*(x)$, $\ldots, f_K^*(x)$ na qual aproxima-se da mediana da posteriori de $f(x)$. Para mensurar a incerteza a cerca de $f(x)$, duas medidas podem ser tomadas: a variância das amostras $f_1^*(x), \ldots, f_K^*(x)$ ou um intervalo a posteriori $(1 - \alpha)\%$ de $f(x)$. Ou ainda, percentis 2,5% e 97,5% de $f(x)$, caso queira-se 95% de credbilidade.

O modelo BART também pode ser usado para seleção de variáveis; para isso, conta-se quais variáveis aparecem mais vezes em todas as árvores. Vale notar que essa estratégia é menos efetiva quando temos um m muito grande, já que a competição para entrar na árvore diminui e assim o modelo acaba combinando preditores relevantes com outros irrelevantes. Por outro lado, quando m, é pequeno a competição para entrar nas árvores aumenta, levando o modelo BART a selecionar somente variáveis relevantes para predição.

Esse método de seleção de variáveis livre de modelo é realizado pela observação do que ocorre com as componentes de x nas amostras MCMC de $f_1^*, \ldots, f_K^*$ com número de árvores m cada vez menores. Para cada simulação da soma de árvores f_k^*, seja z_{ik} a proporção de todas as regras de

divisão que usam a i-ésima componente de x. Então,

$$v_i = \frac{1}{K}\sum_{k=1}^{K} z_{ik} \tag{2.18}$$

é a média de uso da componente i-ésima de x em todas as regras de divisão. Os efeitos da escolha de m em v_i serão mais bem explorados no capítulo seguinte.

2.1.4 BART Probit

Nesta seção será abordada a extensão do modelo para variáveis binárias em Y (representadas por = 0 ou 1) feita por Chipman *et al.* [CGM10]. A extensão se dá de forma direta para um modelo de classficação probit

$$p(x) = P(Y = 1|x) = \Phi[G(x)] \tag{2.19}$$

onde

$$G(x) = \sum_{j=1}^{m} g(x; T_j, M_j) \tag{2.20}$$

e Φ representa a função densidade acumulada de uma normal padrão. Para extender o BART para o modelo 2.19, impõe-se uma priori regularizadora para em $G(x)$ (2.20) e usa-se o algoritmo de *backfitting* para cálculo da posteriori. Diferentemente do modelo 2.3, o modelo 2.19 implicitamente assume $\sigma = 1$, assim apenas a priori em $(T_1, M_1), \ldots, (T_m, M_m)$ é necessária. Supondo independência, assume-se a priori da seguinte forma

$$p((T_1, M_1), \ldots, (T_m, M_m)) = \prod_j \left[p(T_j) \prod_i p(\mu_{ij}|T_j) \right], \tag{2.21}$$

onde cada árvore tem priori $p(T_j)$ como no modelo BART. Para a priori de $p(\mu_{ij}|T_j)$, os autores consideram que o intervalo $(\Phi(-3), \Phi(3))$ contém a maior parte dos valores de interesse de $p(x)$, e procedendo de forma similar ao caso contínuo, recomendam a escolha

$$\mu_{ij} \sim N(0, \sigma_\mu^2), \qquad \text{onde } \sigma_\mu = 3.0/k\sqrt{m} \tag{2.22}$$

onde k é tal que $G(x)$ (2.20) dá alta probabilidade para o intervalo $(-3, 3)$. Da mesma maneira que no caso contínuo, essa priori tem efeito de encolhimento (regularização) dos parâmetros da árvore μ_{ij} em torno do 0, limitando o efeito individual de cada árvore em $G(x)$ (2.20). Valores maiores para k e/ou m aumentam o encolhimento dos μ_{ij}'s. O valor recomendado pelos autores é $k = 2$, e valores entre 1 e 3 levam a bons resultados. Eles ainda ressaltam que k pode ser escolhido por validação cruzada.

A regularização de $G(x)$ (2.20) em direção ao 0 tem como efeito regularizar $p(x) = \Phi(G(x))$ em direção a 0,5. Caso deseje-se alterar o efeito da regularização para um p_0 outro que não 0,5, pode-se trocar $G_c(x) = G(x) + c$ em 2.19 com o deslocamento $c = \Phi^{-1}(p_0)$.

O algoritmo de *backfitting* pode ser utilizado de maneira direta para simulação da posteriori: basta usar a ideia de *augmentation* proposta por Albert e Chib [AC93]. A ideia é reescrever o modelo 2.19 introduzindo as variáveis latentes $Z_1, \ldots, Z_n \overset{i.i.d.}{\sim} N(G(x), 1)$, tais que $Y_i = 1$ se $Z_i > 0$ e $Y_i = 0$ se $Z_i \leq 0$. Dessa forma, $Z_i|[y_i = 1] \sim \max\{N(g(x), 1),\ 0\}$ e $Z_i|[y_i = 0] \sim \min\{N(g(x), 1),\ 0\}$. Incorporando a simulação da variável latente Z_i no algoritmo de *backfitting*, o amostrador de Gibbs realiza n amostras sucessivas de $Z_i|y_i$, $i = 1, \ldots, n$, seguida de m amostras de $(T_j, M_j)|T_{(j)}, M_{(j)}, z_1, \ldots, z_n$ para $j = 1, \ldots, m$ como anteriormente explicado para o caso contínuo.

A sequência induzida pela funções de soma de árvores

$$p^*(\cdot) = \Phi\left[\sum_{j=1}^{m} g(\cdot, T_j^*, M_j^*) \right], \tag{2.23}$$

para a sequência de amostras $(T_1^*, M_1^*), \ldots, (T_m^*, M_m^*)$ é convergente para a "verdadeira" distribuição a posteriori de $p(\cdot)$. A sequência mostrada g^*, após um período razoável de aquecimento da cadeira, representada por $g_1^*, \ldots, g_K^*$, pode ser considerada uma aproximação da posteriori, assim podendo ser usada para tomar inferência de $p(\cdot)$. As inferências podem ser feitas de forma idêntica às descritas para o caso contínuo.

2.2 Consenso Monte Carlo

O Consenso Monte Carlo (CMC), proposto por Scott *et al.* (2013), tem como ideia dividir os dados em uma série de grupos, chamados *shards*, e entregar para cada máquina uma partição dos dados. Cada máquina, por sua vez, realiza a simulação de Monte Carlo da posteriori baseada nesse *shard*, e então, todas as posterioris de cada computador são combinadas a fim de produzir o Consenso Monte Carlo de todas as posterioris. Cada posteriori de um *shard* dos dados recebe o nome de subposteriori em Wang & Dunson (2013). Para os modelos com uma estrutura de independência apropriada, temos que:

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{\int_\Omega p(y|\theta)p(\theta)d\theta} = \frac{p(\theta)\prod_{s=1}^S p(y_s|\theta)}{\int_\Omega p(\theta)\prod_{s=1}^S p(y_s|\theta)d\theta} = \frac{\prod_{s=1}^S p(y_s|\theta)p(\theta)^{1/S}}{\int_\Omega \prod_{s=1}^S p(y_s|\theta)p(\theta)^{1/S}d\theta}, \tag{2.24}$$

onde y são os todos os dados, y_s os dados do *shard* s e θ representa todos os parâmetros do modelo. Note que a priori é divida em S partes para preservar a quantidade total de informação a priori no sistema.

As principais vantagens do CMC são: ele requer apenas uma comunicação durante o processo de amostragem, realiza boas aproximações, pode utilizar o mesmo código de uma abordagem não paralela. As principais desvantagens são: não existem garantias teóricas para modelos não-Gaussianos, funciona bem somente quando a média faz sentido para agregação dos resultados, não funciona bem com parâmetros discretos. A escolha do número de *shards*, S, pode gerar um problema de viés devido ao tamanho amostral se tornar muito pequeno para cada *shard*. No entanto, neste trabalho existe uma sugestão de mudança para que os efeitos da de viés por amostra pequena sejam minimizados para modelos de regressão. Apesar de suas desvantagens, o CMC possibilita trabalhar com uma grande quantidade de dados, sem aumento extraordinário de tempo computacional. Os exemplos também mostram que mesmo para dados não-Gaussianos, o CMC produz bons resultados.

2.2.1 Combinando as amostras em modelos Gaussianos

Suponha que cada uma das máquinas tenha gerado uma amostra de Monte Carlo dos parâmetros usando um *shard*. Assim, temos uma amostra dos parâmetros para cada $p(\theta|y_s) \propto p(y_s|\theta)p(\theta)^{(1/S)}$. A ideia é combinar as amostrar das diferentes subposterioris usando uma média ponderada. Para isso, é necessária uma matriz para cada subposteriori. Dessa forma, a amostra de CMC é dada por:

$$\theta_g = \left(\sum_{s=1}^S W_s\right)^{-1} \sum_{s=1}^S W_s\theta_{sg}, \tag{2.25}$$

onde θ_{sg} é a g-ésima realização da posteriori do *shard* s, ou seja, $p(\theta|y_s)$, θ_g é a g-ésima realização da posteriori de consenso e W_s é o peso escolhido para a *shard* s.

Supondo um modelo Gaussiano, para cada uma das subposterioris calcula-se a matriz de covariância amostral, ou seja, $\Sigma_s = \text{Var}(\theta|y_s)$, e escolhe-se o peso como $W_s = \Sigma_s^{-1}$. O exemplo a seguir ilustra o motivo dessa escolha para W_s.

Suponha que $y|\theta = (y_1, \ldots, y_n)$, onde i.i.d. $y|\theta \sim N(\theta, 1)$. Deseja-se estimar θ, e foi escolhida a priori $\theta \sim N(0, 1)$. Dividindo o $y|\theta$ em S grupos, tem-se:

$$\left.\begin{array}{rcc} y_1|\theta & = & (y_1,\ldots,y_{n_1}) \\ \vdots & & \vdots \\ y_S|\theta & = & (y_{n_{S-1}+1},\ldots,y_{n_S}) \end{array}\right\} S \text{ grupos com } \sum_{s=1}^{S} n_s = n.$$

A subposteriori do i-ésimo *shard* é dada por

$$\begin{aligned} p(\theta|y_i) &\propto p(y_i|\theta)p(\theta)^{1/S} \propto \exp\left\{-\frac{1}{2}[n_i\theta^2 - 2\theta n_i\overline{y}_i]\right\} \times \exp\left\{-\frac{1}{2}\frac{\theta^2}{S}\right\} \\ &\propto \exp\left\{-\frac{1}{2}\left[\left(n_i + \frac{1}{S}\right)\theta^2 - 2\theta n_i\overline{y}_i\right]\right\}. \end{aligned} \tag{2.26}$$

Dessa maneira,

$$\theta|y_i \sim N\left(\frac{n_i\overline{y}_i}{n_i + {}^1\!/_S},\ \frac{1}{n_i + {}^1\!/_S}\right). \tag{2.27}$$

Assim, o peso $W_i = n_i + {}^1\!/_S$, então $\sum_{s=1}^{S} W_s = n + 1$; logo, substituindo em 2.25, tem-se

$$\theta_g = \left(\frac{1}{n+1}\right)\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)\theta_{sg}\right) \tag{2.28}$$

Por fim, calculando $E(\theta_g)$ e $\mathrm{Var}(\theta_g)$ obtém-se

$$\begin{aligned} E(\theta_g) &= E\left[\left(\frac{1}{n+1}\right)\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)\theta_{sg}\right)\right] \\ &= \left(\frac{1}{n+1}\right)\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)E[\theta_{sg}]\right) \\ &= \left(\frac{1}{n+1}\right)\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)\frac{n_s\overline{y}_s}{(n_s + {}^1\!/_S)}\right) \\ &= \left(\frac{1}{n+1}\right)\left(\sum_{s=1}^{S} n_s\overline{y}_s\right) \\ &= \frac{\sum_{s=1}^{S}\sum_{i=1}^{n_s} y_i}{n+1} \\ &= \frac{\sum_{i=1}^{n} y_i}{n+1} \end{aligned} \tag{2.29}$$

$$\begin{aligned} \mathrm{Var}(\theta_g) &= \mathrm{Var}\left[\left(\frac{1}{n+1}\right)\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)\theta_{sg}\right)\right] \\ &= \left(\frac{1}{n+1}\right)^2\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)^2\,\mathrm{Var}[\theta_{sg}]\right) \\ &= \left(\frac{1}{n+1}\right)^2\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)^2\,\frac{1}{(n_i + {}^1\!/_S)}\right) \\ &= \left(\frac{1}{n+1}\right)^2\left(\sum_{s=1}^{S}(n_s + {}^1\!/_S)\right) \\ &= \frac{n+1}{(n+1)^2} = \frac{1}{n+1} \end{aligned} \tag{2.30}$$

Calculando a posteriori com todos os dados:

$$\begin{aligned} p(\theta|y) \propto p(y|\theta)p(\theta) &\propto \exp\left\{ -\frac{1}{2}(n\theta^2 - 2\theta n\overline{y})\right\} \times \exp\left\{ -\frac{\theta^2}{2}\right\} \\ &\propto \exp\left\{ -\frac{1}{2}\left[(n+1)\theta^2 - 2\theta(n+1)\frac{n\overline{y}}{(n+1)}\right]\right\}. \end{aligned} \tag{2.31}$$

Portanto, a posteriori tem distribuição

$$\theta|y \sim N\left(\frac{n\overline{y}}{n+1},\ \frac{1}{n+1}\right), \tag{2.32}$$

então a posteriori de consenso recupera a distribuição a posteriori com todos os dados.

Os autores ([SBB+16]) sugerem um algoritmo para combinar as amostras em um CMC para qualquer distribuição.

Algoritmo 2: O algoritmo de Consenso Monte Carlo.

Input : Dados y
Output: Amostra da posteriori via CMC

1 Dividir y em S partições $y_1, \ldots, y_S$.
2 Gerar S amostras de Monte Carlo separadamente de $\theta_{sg} \sim p(\theta|y_s)$ para $g = 1, \ldots, G$ com cada partição usando a priori fracionada $p(\theta)^{1/S}$.
3 Combinar as amostras de cada partição utilizando $\theta_g = (\sum_s W_s)^{-1}(\sum_s W_s\theta_{sg})$.

O algoritmo do CMC desta forma é exato apenas para o caso em que as subposterioris seguem a distribuição normal. No contexto deste estudo, ou seja, para o modelo BART como previamente definido, as folhas, dada a estrutura da árvore, seguem distribuição normal. No entanto, isso não garante que o modelo siga a distribuição normal.

Para ilustrar o algoritmo de CMC, serão exibido dois exemplos sintéticos. O primeiro, com a função geradora dos dados sendo uma distribuição normal e o segundo, com a função geradora dos dados sendo uma distribuição exponencial.

Exemplo 1. Imagine que se deseja estimar as posterioris $\mu|y$ e $\tau|y$ de uma amostra de tamanho $n = 30$ de $y|\mu, \tau \sim N(18;\ 0,25)$ com distribuição priori $\mu \sim N(0, 1)$ e $\tau \sim Gama(0, 0)$, isto é, uma densidade imprópria. Divide-se aleatoriamente, então, a amostra em duas partes, y_1 e y_2. Cada uma das amostras é enviada para um computador distinto. Cada um desses computadores amostra a subposteriori, $\mu_1, \tau_1|y_1$ e $\mu_2, \tau_2|y_2$, utilizando um amostrador de Gibbs, e envia a amostra para o computador primário. O computador primário então calcula a posteriori de consenso. Para isso, primeiramente ele calcula as matrizes de covariância das amostras, Σ_1 e Σ_2, onde

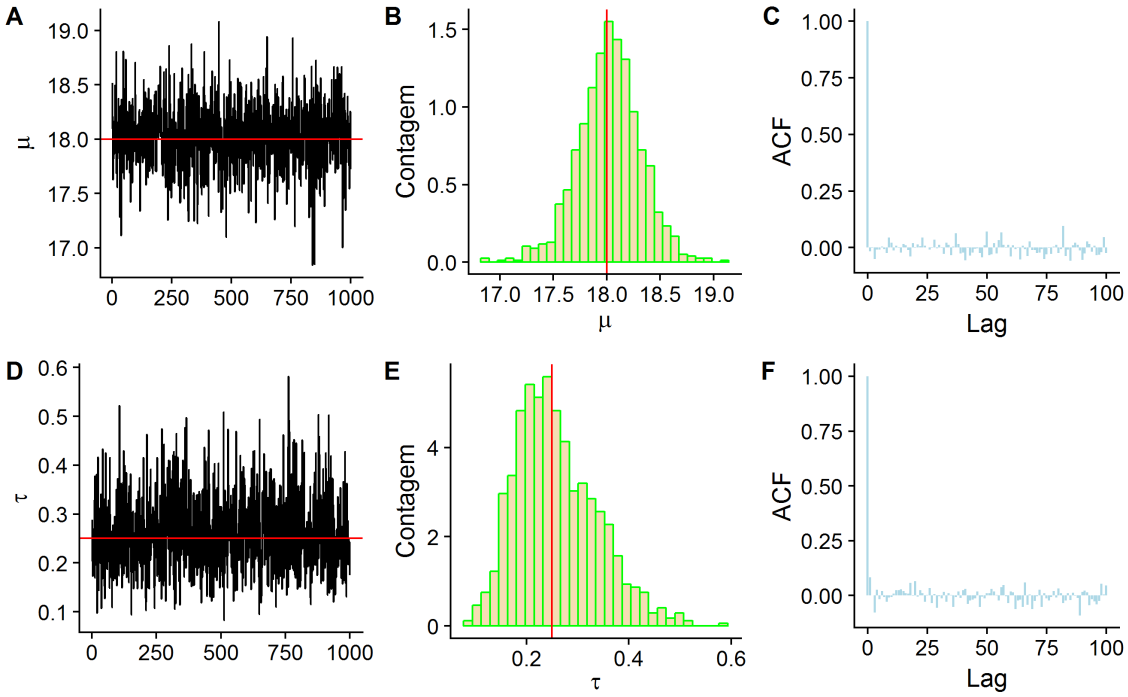
$$\Sigma_i = \begin{pmatrix} \text{Var}(\mu_i|y_i) & \text{Cov}(\mu_i|y_i,\ \tau_i|y_i) \\ \text{Cov}(\sigma_i^2|y_i,\ \mu_i|y_i) & \text{Var}(\tau_i|y_i) \end{pmatrix}. \tag{2.33}$$

Assim, são calculados os pesos de ponderação do consenso $W_1 = \Sigma_1^{-1}$ e $W_2 = \Sigma_2^{-1}$, e por fim a amostra de consenso é calculada:

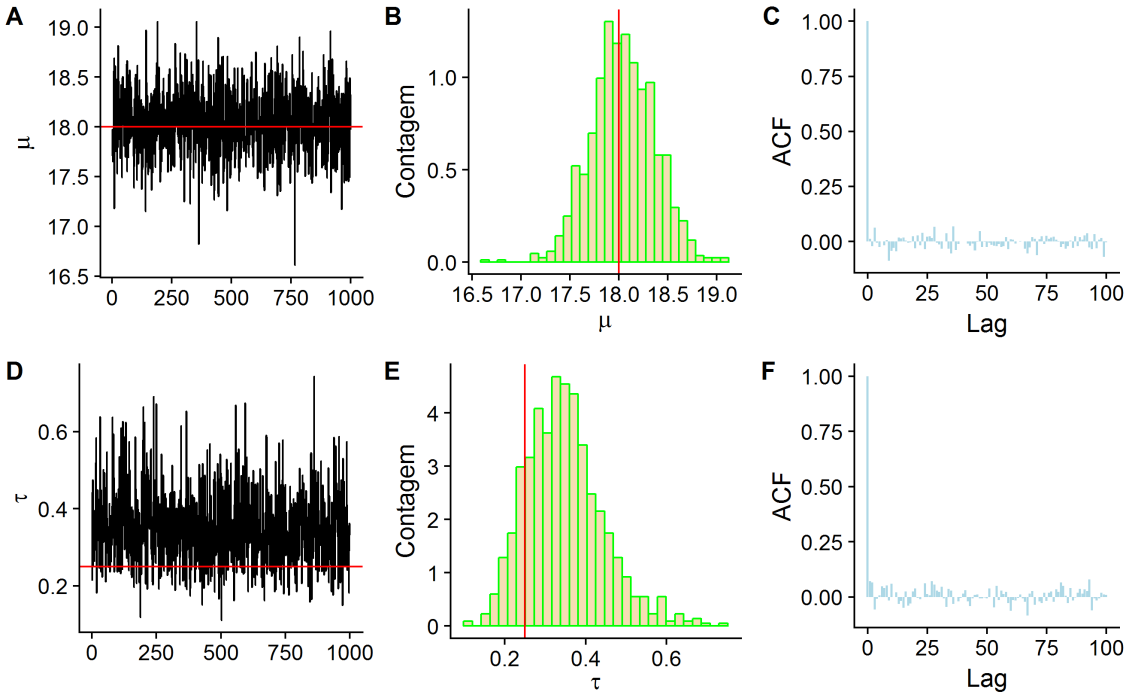
$$\mu, \tau|y = (W_1 + W_2)^{-1}\ (W_1 \times \mu_1, \tau_1|y_1\ +\ W_2 \times \mu_2, \tau_2|y_2). \tag{2.34}$$

A figura 2.1 apresenta o MCMC (dados completos) e o CMC (dados particionados em dois). Para o processo de CMC, Figura 2.1 (a) pode-se perceber que as posterioris de consenso de μ e τ ficaram centralizadas em torno do verdadeiros valores, isto é, 18 e 0,25, respectivamente. Pode-se notar que o amostrador de Gibbs teve um bom *mixing* e que o processo de amostragem teve uma baixa auto-correlação. Na Figura 2.1 (b) temos o MCMC, ou seja, foram utilizados todos os dados em uma única máquina para estimar as posterioris para μ e τ. A posteriori para μ ficou centrada em torno do verdadeiro valor, 18, porém a distribuição a posteriori de τ não ficou tão centralizada em torno de 0,25. Essa não centralização em torno do verdadeiro valor não se mostra problemática,

uma vez que a distribuição ainda captura o verdadeiro valor no intervalo de credibilidade. Contudo, nota-se que o CMC apresentou uma melhor performance. O MCMC também possui um bom *mixing* e o processo de amostragem não apresenta auto-correlação.



(a) *Posteriori de consenso com dois shards.*



(b) *Posteriori usando todos os dados.*

Figura 2.1: *Exemplo 1: 100 realizações da posterioris, via CMC (a) ou MCMC (b), de 30 observações da distribuição* $X|\mu,\ \tau \sim N(18; 0,25)$.

Exemplo 2. Agora deseja-se realizar inferência sobre 30 observações de $y|\lambda \sim \exp(5)$. Para isso, escolheu-se uma priori $\lambda \sim Gama(0,001; 0,001)$. Note que para os dados completos a posteriori

para n observações de $y|\lambda \sim \exp(\lambda)$ com priori $\lambda \sim Gama(\alpha, \beta)$ é dada por

$$\begin{aligned} p(\lambda|y) &\propto p(y|\lambda)p(\lambda) \\ &\propto \lambda^n \exp\big\{-\lambda n\overline{y}\big\} \times \lambda^{\alpha-1} \exp\big\{-\beta\lambda\big\} \\ &\propto \lambda^{\alpha+n-1} \exp\big\{-\lambda(\beta + n\overline{y})\big\}. \end{aligned} \tag{2.35}$$

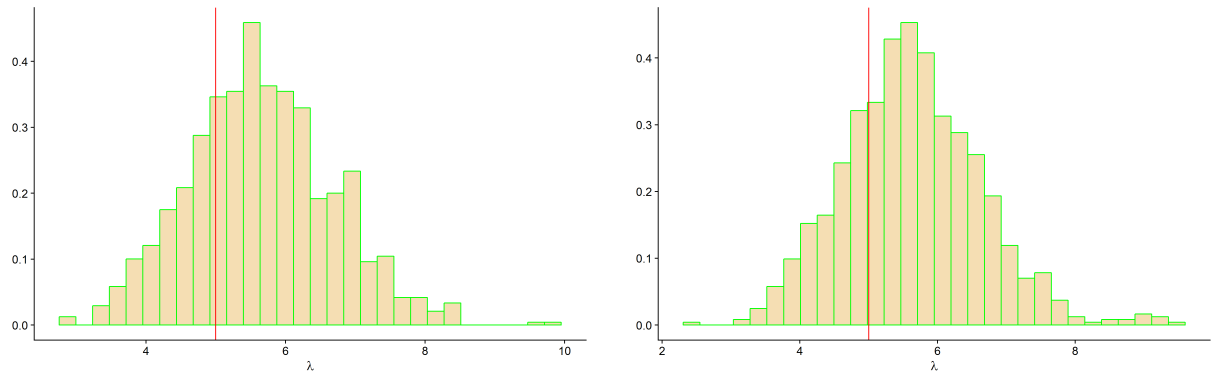
Portanto, a posteriori é $\lambda|x \sim Gama(\alpha + n, \beta + n\overline{y})$. Já para o CMC temos que a posteriori do $i-$ésimo *shard* é dada por

$$\begin{aligned} p(\lambda|y_i) &\propto p(y_i|\lambda)p(\lambda)^{1/S} \\ &\propto \lambda^{n_i} \exp\big\{-\lambda n_i\overline{y}_i\big\} \times \lambda^{\frac{\alpha-1}{S}} \exp\left\{\frac{-\beta\lambda}{S}\right\} \\ &\propto \lambda^{n_i+\frac{\alpha+S-1}{S}-1} \exp\left\{-\lambda\left(\frac{\beta}{S} + n_i\overline{y_i}\right)\right\}. \end{aligned} \tag{2.36}$$

Porém, o CMC, utlizando a média ponderada das subposterioris com peso $W_i = \text{Var}(\lambda|y_i)^{-1}$, não recupera a posteriori que utiliza todos os dados. Considerando $n_i = \frac{n}{S}$, tem-se

$$E(\lambda|y) = E\left(\frac{1}{\sum_{s=1}^{S} W_i}\left(\sum_{s=1}^{S} W_i \times \lambda|y_i\right)\right) = \frac{n + \alpha + S - 1}{\frac{[\sum_{s=1}^{S}(\sum_{j\in s} y_j + \beta/S)^2]}{(\sum_{i=1}^{n} y_i + \beta)/S}}. \tag{2.37}$$

Entretanto, o algoritmo de CMC com $S = 2$ consegue, ainda assim, estimar numericamente a posteriori $\lambda|y$ como nota-se na Figura 2.2. Desta maneira, vemos que mesmo quando o CMC não reproduz exatamente a posteriori, o algoritmo ainda consegue realizar uma boa aproximação da posteriori utilizando todos os dados.



(a) *Posteriori de consenso com dois shards.* **(b)** *Posteriori usando todos os dados*

Figura 2.2: *Exemplo 2. 100 realizações das posterioris de 30 observações da distribuição* $X|\lambda \sim exp(\lambda)$.

2.2.2 Consenso Monte Carlo BART (CMC-BART)

O Consenso Monte Carlo BART (CMC-BART) é o uso do BART com o CMC. É importante ressaltar que nesse caso não se realiza a agregação dos parâmetros de cada modelo BART, mas sim a agregação entre as distribuições preditivas geradas por cada modelo BART para cada *shard* dos dados. Dessa forma, existem outras maneiras de gerar a distribuição preditiva de consenso. Aqui serão apresentadas outras duas maneiras de agregar as predições: a primeira criada por Scott *et al.* [SBB+16] e a segunda criada pelo autor deste trabalho. Além disso, o CMC-BART também é usado para selecionar variáveis relevantes.

Combinando as amostras para o CMC-BART

Considere o conjunto de vetores onde se deseja gerar predição, dado por $\mathbf{x}_1, \ldots, \mathbf{x}_K$. Esse conjunto de vetores é o conjunto de teste do modelo, isto é, temos o mesmo conjunto para todos os *shards* do CMC. Assim, são geradas amostras das distribuições preditivas para cada $\mathbf{x}_k$ em cada *shard*. Pode-se assim calcular a variância amostral de cada distribuição preditiva para cada *shard* s denotada por

$$s_{sk}^2 = \sum_{g=1}^{G} \frac{(f_{sg}(\mathbf{x}_k) - \bar{f}_s(\mathbf{x}_k))^2}{G-1}, \tag{2.38}$$

onde $f_{sg}(\mathbf{x}_k)$ é a g-ésima amostra da distribuição preditiva de $\mathbf{x}_k$ no *shard* s, G é o número amostrado em cada *shard* e $\bar{f}_s(\mathbf{x}_k)$ é a média das preditivas no *shard* s. Usamos $(s_{sk}^2)^{-1}$ como peso para gerar a amostra de consenso. Desta forma, temos o consenso de cada $f(\mathbf{x}_k)$ dado por

$$f(\mathbf{x}_k) = \sum_{s=1}^{S} \frac{(s_{sk}^2)^{-1} f_{sg}(\mathbf{x}_k)}{s_k}, \tag{2.39}$$

onde $s_k = \sum_{j=1}^{S} (s_{jk}^2)^{-1}$.

A segunda forma de se realizar a combinação das subposterioris é baseada na proposta anterior, porém ao invés de levar-se em considereção apenas a $(s_{sk}^2)^{-1}$ como peso para formar o consenso, leva-se em consideração também a similaridade de cosenos entre os dados x_s e o ponto a ser predito $\mathbf{x}_k$. O consenso passa, então, a ser formado por

$$f(\mathbf{x}_k) = \sum_{s=1}^{S} \frac{(s_{sk}^2)^{-1} \rho_{\mathbf{x}_k, x_s} f_{sg}(\mathbf{x}_k)}{w_k}, \tag{2.40}$$

onde $\rho_{\mathbf{x}_k, x_s}$ é a maior similaridade por coseno de um ponto a ser predito $\mathbf{x}_k$ e os pontos do *shard* s, ou seja, x_s e $w_k = \sum_{s=1}^{S} (s_{jk}^2)^{-1} \rho_{\mathbf{x}_k, x_s}$.

A similaridade por coseno é dada por

$$\text{sim}(w, z) = cos(\theta) = \frac{z \cdot w}{||z||\,||w||}, \tag{2.41}$$

onde w e z são dos vetores $n-$dimensionais e $||\cdot||$ é a norma do vetor.

Quando realizamos o *shard* dos dados entre os diferentes computadores, existe um aumento do potencial de viés devido à diminuição da amostra. Em um contexto Bayesiano, teríamos um deslocamento da posteriori. Isso ocorre principalmente quando os dados são divididos em partes tão pequenas e o número de observações em cada *shard* torna-se muito pequeno. Outro possível problema seria existir nenhum ou poucos pontos similares ao ponto em que se deseja realizar a predição, e assim, acaba-se por errá-la. Para lidar com esse problema, é proposta essa forma de agregação por similaridade.

Seleção de variáveis no CMC-BART

O modelo CMC-BART também pode é usado para seleção de variáveis. Para isso, basta calcular o v_{is}, isto é, o uso da variável i no shard s, tal como é feito no BART. Em seguida, calcula-se

$$v_i = \frac{1}{S} \sum_{s=1}^{S} v_{is}, \tag{2.42}$$

que é a média do uso das variáveis nos *shards*. A seleção das variáveis no CMC-BART independe da maneira como os dados são agregados.

No próximo capítulo, são apresentados os resultados do CMC-BART para dados sintéticos e dados reais de regressão e classficação. Os resultados são comparados com o BART para todos os

conjuntos de dados apresentados.

Capítulo 3

Dados sintéticos e reais

Neste capítulo, abordam-se os resultados das simulações e experimentos com dados sintéticos e reais. Três tópicos serão aqui discutidos, a saber: a aplicação da função de teste de Friedman [Fri91] aos modelos BART e CMC-BART, um problema de classificação usando dados reais; e, por último, uma regressão, também utilizando dados reais.

Nas seções a seguir, avaliam-se os objetivos citados no capítulo 1, que são: (a) tratar sobre o impacto da escolha dos hiperparâmetros das prioris do CMC-BART para diferentes volumes de dados na capacidade preditiva do modelo, (b) verificar como se comporta seleção de variáveis do CMC-BART, e (c) estudar quais métodos de agregação das amostras geram melhores predições.

Para cada conjunto de dados e combinação de hiperparâmetros, é comparado o resultado do BART em um único *shard* com o CMC-BART com os três diferentes métodos de formação de consenso, sendo eles: (i) a matriz de covariância, (ii) a inversa da variância da distribuição de predição, e (iii) e a inversa da variância da distribuição de predição com a similaridade por conseno. A comparação é feita para avaliar os objetivos mencionados anteriormente.

3.1 Dados sínteticos - a função de teste de Friedman

A função de teste de Friedman (FTF) é uma função utilizada para avaliar um modelo quanto a sua capacidade de seleção de variáveis e predição. Para construir a FTF, simulam-se valores de $x = (x_1, x_2, \ldots, x_p)$ onde $x_1, x_2, \ldots, x_p$, onde $\forall i$, $x_i \overset{i.i.d.}{\sim}$ Uniforme$(0, 1)$ e y dado x é

$$y = f(x) + \varepsilon = 10 \operatorname{sen}(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \varepsilon, \tag{3.1}$$

onde $\varepsilon \sim N(0, \sigma^2)$. Note que a função y depende apenas de $x_1, \ldots, x_5$, logo as variáveis $x_6, \ldots, x_p$ são pseudo-variáveis. O domínio da FTF é $D_f = \{x \in \mathbb{R} \mid 0 \leq x \leq 30\}$.

A função criada por Friedman [Fri91] é utilizada por ele para avaliar regressão splines e por Chipman *et al.* [CGM10] e por Scott *et al.* [SBB+16] para avaliar os modelos BART e o CMC-BART, respectivamente. Para efeito de comparação com os trabalhos anteriores, foi escolhida essa mesma função de teste neste trabalho.

Foi simulada a FTF com erro $\sigma = 3$, $p = \{10,\ 20,\ 100,\ 1000\}$ dimensões e $n = \{5500,\ 11000\}$ observações, tendo as observações sido divididas em 11 partes, onde a primeira foi utilizada como conjunto de teste para todos os *shards*, e as partes restantes, como conjunto de treino. Para cada amostra da FTF foram ajustados os modelos BART e o CMC-BART, com 10 *shards*, para diferentes hiperparâmetros, sendo eles: $m = \{5,\ 10,\ 20,\ 100,\ 200\}$, $\alpha = \{0,90;\ 0,95;\ 0,99\}$ e $\beta = \{1,\ 2\}$ e $k = \{2,\ 3,\ 5\}$. Para todos os modelos os hiperparâmetros da priori σ foram $(\nu,\ q) = (3;\ 0,90)$, que são considerados os valores padrões por Chipman *et al.* [CGM10]. Foram escolhidos 100 pontos de cortes para cada variável explicativa. Por fim, foram amostradas 6000 observações a posteriori via MCMC, sendo que as primeiras 1000 observações são de *burn-in*. Nas subsessões seguintes avaliam-se: o desempenho na seleção de variáveis do CMC-BART e a capacidade de predição fora da amostra dos modelos CMC-BART para diferentes configurações de hiperparâmetros; em ambas

as análises foram comparadas a capacidade de seleção de variáveis e a qualidade das predições com os resultados do BART.

Seleção de variáveis do CMC-BART

Como mencionado no capítulo 2, o modelo BART tem uma boa capacidade de seleção de variáveis. A seguir, na presente seção, é analisada a capacidade de seleção de variáveis do CMC-BART e os resultados foram comparados à seleção de variáveis do BART. Tal análise tem como objetivo verificar se o CMC-BART possui um desempenho satisfatório na seleção de variáveis para diversas dimensões ($p = 10$, 20, 100 e 1000) da FTF. Cabe ressaltar que quanto maior o número de dimensões maior será o número de pseudo-variáveis.

Para selecionar o modelo com melhor desempenho na seleção das variáveis, realizou-se uma diferença entre $\gamma_{\text{var}} = \frac{1}{5}\sum_{i=1}^{5} v_i$ e $\gamma_{\text{pseudo-var}} = \frac{1}{p-5}\sum_{i=6}^{p} v_i$. Os modelos com a maior diferença entre $\gamma_{\text{var}} - \gamma_{\text{pseudo-var}}$ foram escolhidos para cada combinação de m, n e p. Vale relembrar que valores maiores, tanto de m quanto de k, produzem uma diminuição na importância de cada árvore no ajuste do modelo (*shrinkage effect*). Já valores maiores de α e β produzem, em geral, árvores com maior profundidade.

Seleção de variáveis do CMC-BART com FTF com $p = 10$

A Tabela 3.1, a seguir, mostra os hiperparâmetros que tiveram o melhor desempenho na seleção de variáveis com $p = 10$ para cada m. Na parte (a) $n = 5000$; já na parte (b) $n = 10000$, tanto para o modelo BART quanto para o CMC-BART. Na Figura 3.1 nota-se o desempenho na seleção de variáveis para cada um desses hiperparâmetros listados na Tabela 3.1. Ao se analisar ambos, pode-se notar que o modelo CMC-BART tem desempenho melhor quando o número de m é menor; vê-se que esse mesmo comportamento é encontrado no modelo BART. Para $m = 100$ ou 200, verifica-se que o modelo CMC-BART tem dificuldade em separar as variáveis das pseudo-variáveis; já para árvores menores, isto é, $m = 5$ ou 10, a seleção das variáveis torna-se mais adequada e semelhante ao desempenho encontrado no BART.

É importante notar que para $m = 20$, o desempenho na seleção de variáveis se encontra entre o desempenho de uma árvore pequena e uma grande. Árvores desse tamanho podem ser uma boa escolha quando tem-se como objetivo a predição e a seleção de variáveis simultaneamente. O aumento do tamanho da amostra da FTF não pareceu tão benéfico na escolha das variáveis tanto para o CMC-BART quanto para o próprio BART num contexto de baixa dimensão. Observa-se na Tabela 3.1 uma maior robustez para os hiperparâmetros do CMC-BART em relação aos do BART, o que pode ser considerado uma vantagem para o modelo CMC-BART.

Os modelos CMC-BART com melhor capacidade de seleção de variáveis, ou seja, com $m =$ 5 ou 10, têm hiperparâmetros $\alpha = 0,90$ e $\beta = 2$, quando $n_s = 500$. Essa escolha de α e β mantém, a priori, árvores pequenas. Dessa maneira, árvores com 1, 2, 3, 4 ou ≥ 5 nós terminais recebem probabilidade a priori de 0,10; 0,54; 0,25; 0,08 e 0,02, respectivamente. Note que na configuração padrão sugerida por Chipman *et al.* [CGM10], $\alpha = 0,95$; $\beta = 2$, essas probabilidades são 0,05; 0,55; 0,28; 0,09 e 0,03, respectivamente. Comparando ambos hiperparâmetros, percebemos que árvores mais regularizadas levam a melhores resultados para o CMC-BART. Cabe salientar que mesmo a priori tendo um efeito de encolhimento no tamanho das árvores, durante a amostragem da posteriori notaram-se árvores com até 19 nós terminais. Os autores também citam que quando o número de árvores m é pequeno as árvores tendem a ser mais profundas.

Por outro lado, quando $n_s = 1000$, os hiperparâmetros dos melhores modelos mudam ligeiramente. Para $m = 5$, tem-se $\alpha = 0,95$, e para $m = 10$, tem-se $\alpha = 0,99$. Em ambos casos, a priori que gera árvores mais profundas teve melhor desempenho na seleção das variáveis. Além disso, $\alpha = 0,95$, $\beta = 2$ e $k = 2$ são os valores sugeridos por Chipman *et al.* [CGM10], porém essa configuração só ocorre para $m = 5$. Já $m = 10$ e $\alpha = 0,99$ produzem árvores ainda mais profundas a priori, sendo elas com 1, 2, 3, 4 ou ≥ 5 nós terminais, e recebem probabilidade a priori de 0,01; 0,56; 0,29; 0,10 e 0,04, respectivamente.

Tanto para $n_s = 500$ quanto $n_s = 1000$, o percentual de uso das variáveis $x_1, \ldots, x_5$ foi muito maior do que o das pseudo-variáveis; sendo assim, o CMC-BART é capaz de selecionar variáveis tão bem quanto o BART, para $m = 5$ ou 10. Para esses valores de m, o percentual de uso das variáveis ficou acima de 10%; já para as pseudo-variáveis ficou abaixo de 5% de uso. Sendo assim, é possível distinguir quais são as variáveis realmente importantes para o modelo. Vale notar que o BART tem melhor capacidade de seleção de variáveis, porém um CMC-BART com boas prioris pode gerar resultados satisfatórios, levando um tempo até 5 vezes mais menor, e ser mais apropriado para aplicações de *Big Data*.

Modelo	m	α	β	k
CMC-BART	5	0,90	2	2
CMC-BART	10	0,90	2	2
CMC-BART	20	0,95	2	2
CMC-BART	100	0,90	2	5
CMC-BART	200	0,90	2	5
BART	5	0,99	1	5
BART	10	0,90	2	3
BART	20	0,90	1	2
BART	100	0,90	2	3
BART	200	0,90	2	2

(a) $n = 5000$

Modelo	m	α	β	k
CMC-BART	5	0,95	2	2
CMC-BART	10	0,99	2	2
CMC-BART	20	0,95	2	2
CMC-BART	100	0,90	2	5
CMC-BART	200	0,90	2	5
BART	5	0,90	2	2
BART	10	0,90	1	5
BART	20	0,90	2	3
BART	100	0,95	2	2
BART	200	0,95	2	2

(b) $n = 10000$

Tabela 3.1: *Modelos com melhor desempenho para FTF com* $p = 10$.

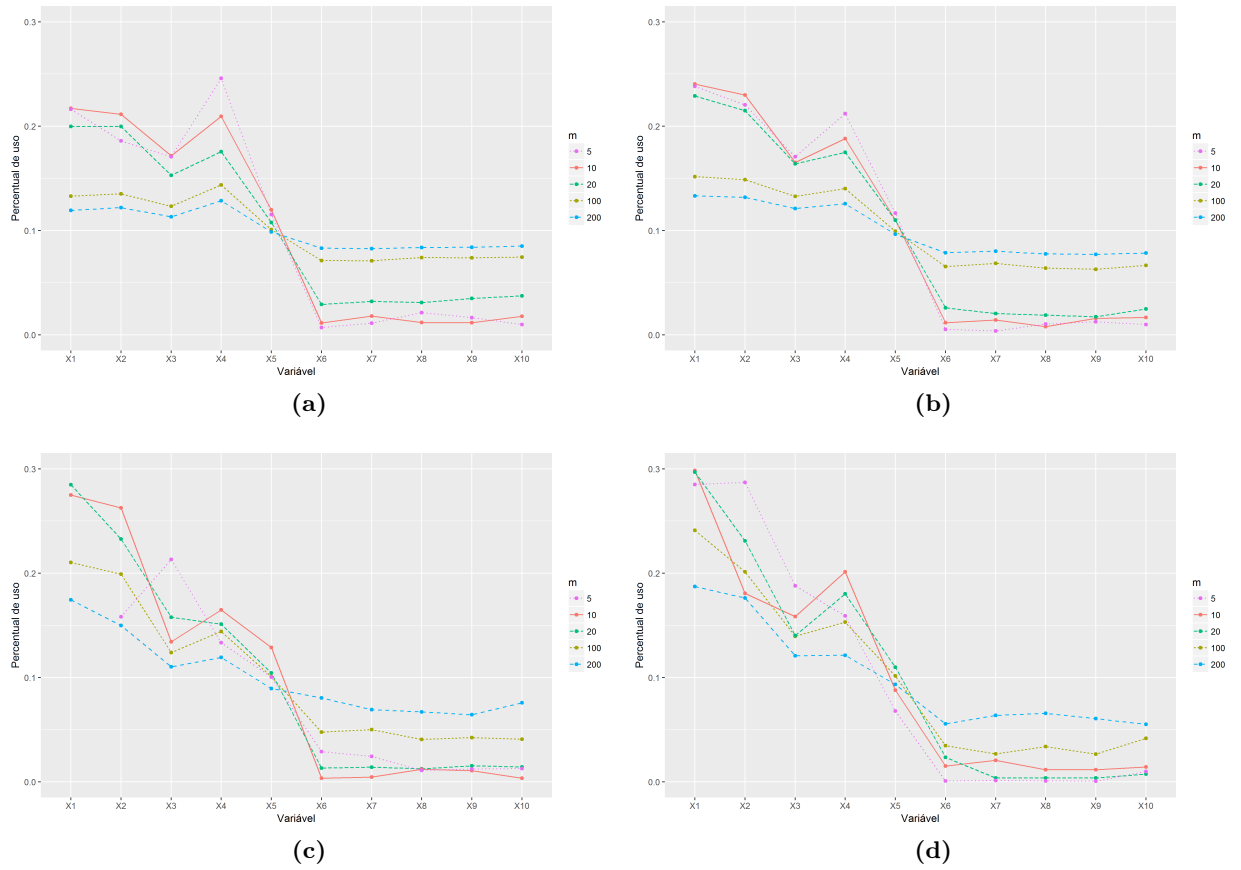


Figura 3.1: *Percentual de uso de cada variável para FTF com 10 dimensões, para diferentes tamanhos de amostra e modelo: (a) CMC-BART com $n_s = 500$ observações em cada shard; (b) CMC-BART com $n_s = 1000$ observações em cada shard; (c) BART com $n = 5000$ observações, (d) BART com $n = 10000$ observações.*

Seleção de variáveis do CMC-BART com FTF com $p = 20$

Na Tabela 3.2, temos os hiperparâmetros dos melhores modelos para seleção de variáveis com $p = 20$, e na Figura 3.2, o desempenho dos respectivos modelos. Nota-se novamente que ambos os modelos, tanto CMC-BART quanto BART, têm melhor desempenho para $m = 5$ ou 10, tal como com $p = 10$. Os modelos com hiperparâmetros que induzem árvores mais profundas a priori são os escolhidos como melhores para seleção de variáveis. Novamente, o CMC-BART mostra-se capaz de selecionar varáveis mesmo com o aumento dimensional.

Modelo	m	α	β	k
CMC-BART	5	0,95	2	3
CMC-BART	10	0,90	2	2
CMC-BART	20	0,90	2	2
CMC-BART	100	0,90	2	5
CMC-BART	200	0,90	2	5
BART	5	0,99	1	2
BART	10	0,99	1	2
BART	20	0,90	2	2
BART	100	0,90	2	2
BART	200	0,90	2	2

(a) $n = 5000$

Modelo	m	α	β	k
CMC-BART	5	0,95	2	2
CMC-BART	10	0,99	2	2
CMC-BART	20	0,90	2	2
CMC-BART	100	0,90	2	2
CMC-BART	200	0,90	2	5
BART	5	0,90	1	5
BART	10	0,90	2	2
BART	20	0,95	2	2
BART	100	0,90	2	2
BART	200	0,90	2	3

(b) $n = 10000$

Tabela 3.2: *Modelos com melhor desempenho para FTF com* $p = 20$.

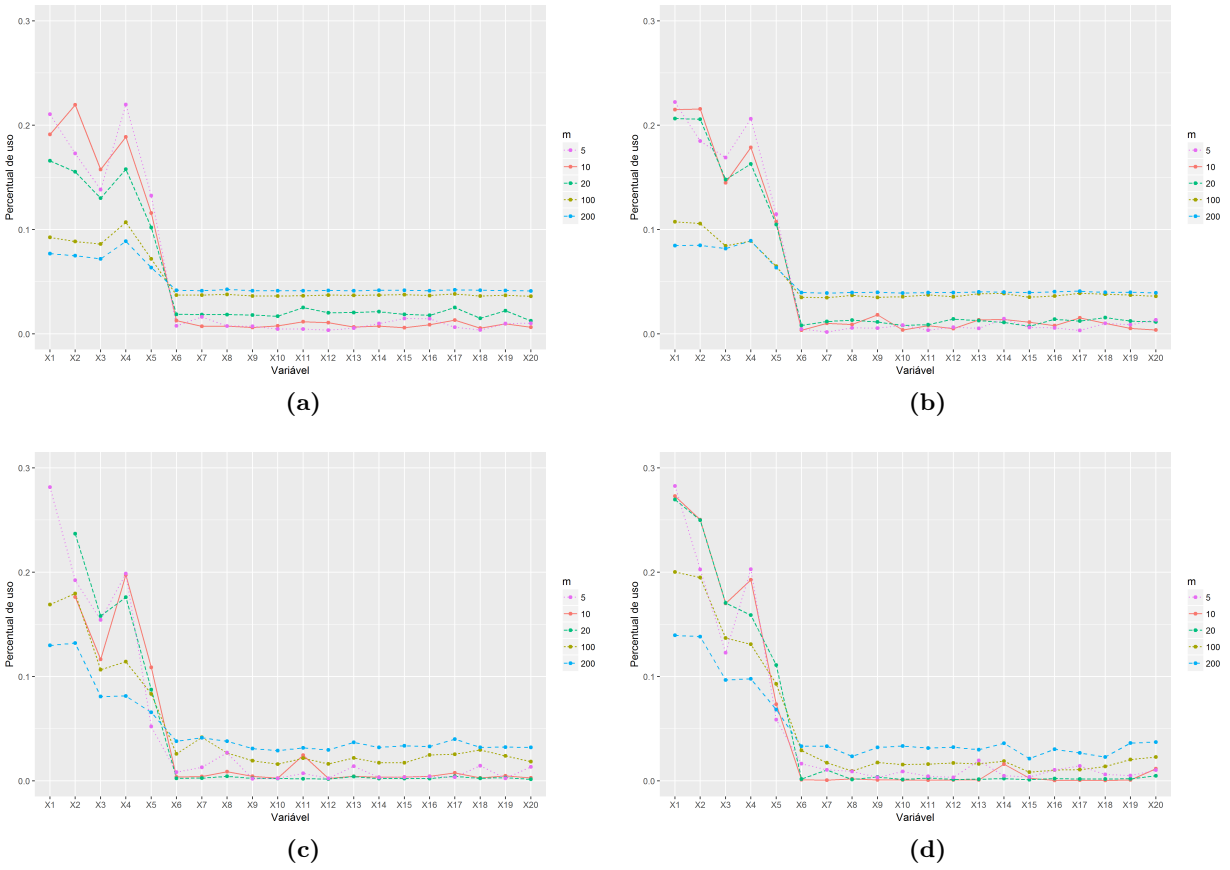


Figura 3.2: *Percentual de uso de cada variável para FTF com 20 dimensões, para diferentes tamanhos de amostra e modelo: (a) CMC-BART com* $n_s = 500$ *observações em cada shard; (b) CMC-BART com* $n_s = 1000$ *observações em cada shard; (c) BART com* $n = 5000$ *observações, (d) BART com* $n = 10000$ *observações.*

Seleção de variáveis do CMC-BART com FTF com $p = 100$

Na Tabela 3.3, temos os hiperparâmetros dos melhores modelos para seleção de variáveis com $p = 100$, enquanto na Figura 3.3 encontra-se o desempenho desses modelos. Novamente, os modelos com melhor desempenho são aqueles com $m = 5$ ou 10. Porém, pode-se observar que o CMC-BART possui maior capacidade de manter as pseudo-variáveis identificáveis do que o BART. Um possível motivo para isso pode residir no fato de que o algoritmo pode ter um melhor *mixing* devido

à mostra ser menor quando dividida em *shards*. Essa qualidade do CMC-BART pode ser ainda melhor percebida quando $p = 1000$.

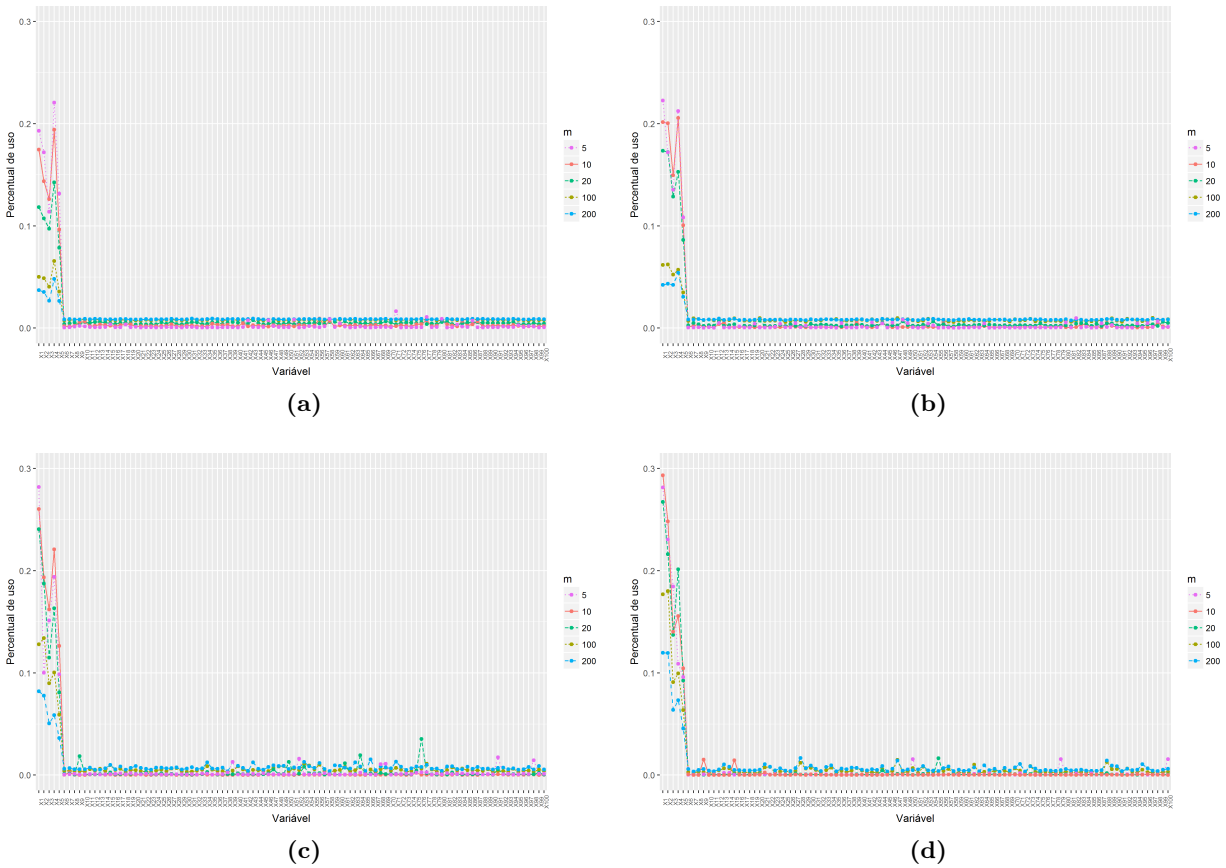
Modelo	m	α	β	k
CMC-BART	5	0,90	2	2
CMC-BART	10	0,90	2	5
CMC-BART	20	0,90	2	5
CMC-BART	100	0,90	2	5
CMC-BART	200	0,90	2	5
BART	5	0,90	1	5
BART	10	0,90	2	2
BART	20	0,95	2	5
BART	100	0,90	2	2
BART	200	0,90	2	2

(a) $n = 5000$

Modelo	m	α	β	k
CMC-BART	5	0,90	2	3
CMC-BART	10	0,90	2	2
CMC-BART	20	0,90	2	2
CMC-BART	100	0,90	2	2
CMC-BART	200	0,90	2	5
BART	5	0,99	1	2
BART	10	0,95	2	3
BART	20	0,90	2	5
BART	100	0,95	2	2
BART	200	0,90	2	2

(b) $n = 10000$

Tabela 3.3: *Modelos com melhor desempenho para FTF com* $p = 100$.



(a) **(b)**

(c) **(d)**

Figura 3.3: *Percentual de uso de cada variável para FTF com 100 dimensões, para diferentes tamanhos de amostra e modelo: (a) CMC-BART com* $n_s = 500$ *observações em cada shard; (b) CMC-BART com* $n_s = 1000$ *observações em cada shard; (c) BART com* $n = 5000$ *observações, (d) BART com* $n = 10000$ *observações.*

Seleção de variáveis do CMC-BART com FTF com $p = 1000$

Para $p = 1000$, a seleção de variáveis do CMC-BART parece ser melhor do que a do BART. Note na Figura 3.4 que em (a) e (b) as variáveis e as pseudo-variáveis estão mais bem divididas

do que em (c) e (d). A Tabela 3.4 traz os valores dos hiperparâmetros para cada um dos modelos apresentados.

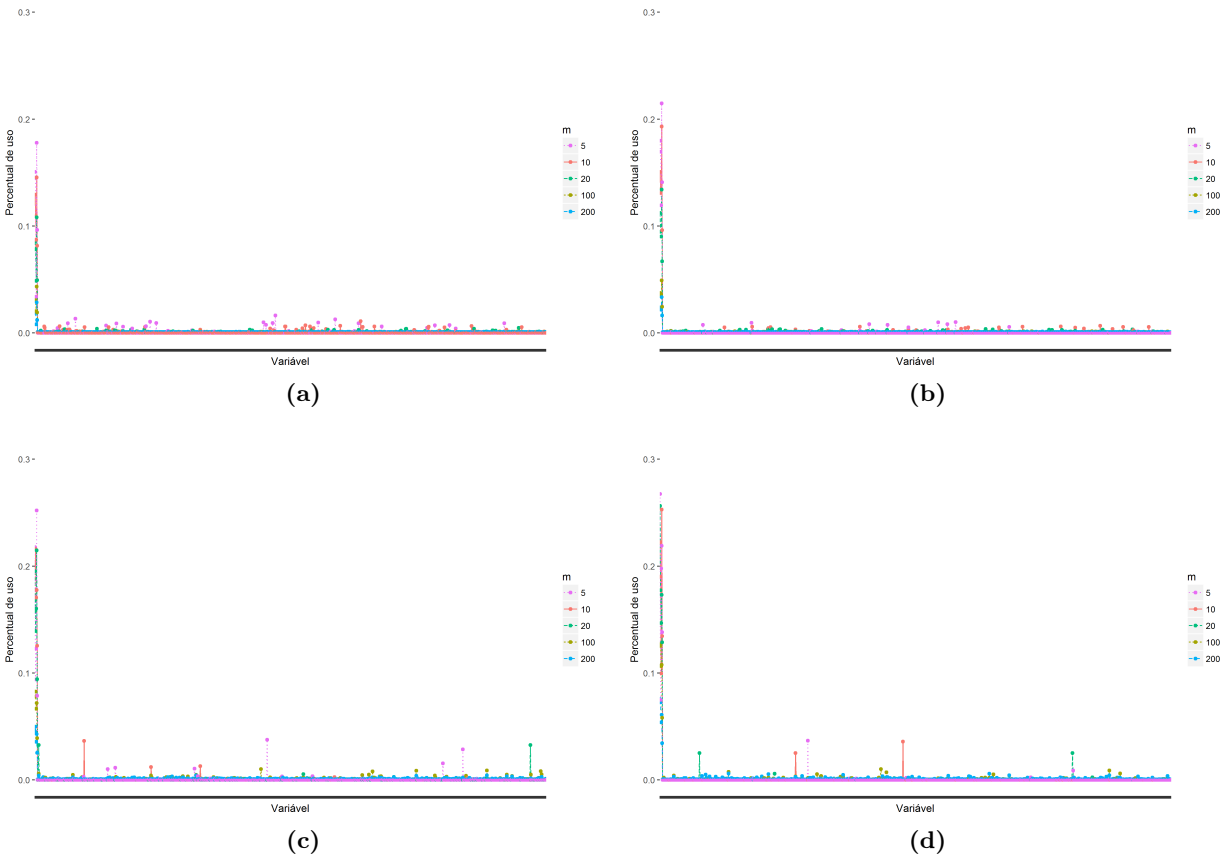
Modelo	m	α	β	k
CMC-BART	5	0,90	2	3
CMC-BART	10	0,95	2	2
CMC-BART	20	0,90	2	5
CMC-BART	100	0,90	2	5
CMC-BART	200	0,90	2	5
BART	5	0,95	1	3
BART	10	0,90	2	2
BART	20	0,90	2	2
BART	100	0,90	2	3
BART	200	0,90	2	5

(a) $n = 5000$

Modelo	m	α	β	k
CMC-BART	5	0,90	2	2
CMC-BART	10	0,90	2	2
CMC-BART	20	0,90	2	5
CMC-BART	100	0,90	2	5
CMC-BART	200	0,90	2	5
BART	5	0,90	1	2
BART	10	0,90	2	3
BART	20	0,90	2	2
BART	100	0,90	2	2
BART	200	0,90	2	2

(b) $n = 10000$

Tabela 3.4: *Modelos com melhor desempenho para FTF com* $p = 1000$.



(a) **(b)** **(c)** **(d)**

Figura 3.4: *Percentual de uso de cada variável para FTF com 1000 dimensões, para diferentes tamanhos de amostra e modelo: (a) CMC-BART com* $n_s = 500$ *observações em cada shard; (b) CMC-BART com* $n_s = 1000$ *observações em cada shard; (c) BART com* $n = 5000$ *observações, (d) BART com* $n = 10000$ *observações.*

Os hiperparâmetros do CMC-BART para seleção de variáveis com melhor desempenho ocorrem quando temos poucas árvores, isto é, para $m = 5$ ou 10. Já os valores de α variaram mais, porém o valor mais típico foi 0,90. $\beta = 2$ foi o hiperparâmetro em todos os modelos escolhidos. Para k, o valor mais comum foi 2, mas também houve modelos com $k = 2$ e até mesmo 5. Dessa forma,

recomendam-se os seguintes hiperparâmetros para seleção de variáveis utilizando o CMC-BART: $m = 10$; $\alpha = 0,90$; $\beta = 2$ e $k = 2$.

Desempenho de predição do CMC-BART

O modelo BART mostra-se com execelente capacidade preditiva, como visto em diversos exemplos dados por Chipman *et al.* [CGM10]. Nessa subseção foi avaliado o desempenho do modelo CMC-BART na predição, bem como os possíveis ganhos de velocidade dessa abordagem. A avaliação do desempenho do modelo CMC-BART é realizada por meio da raiz do erro quadrático médio (RMSE), dada por

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}, \tag{3.2}$$

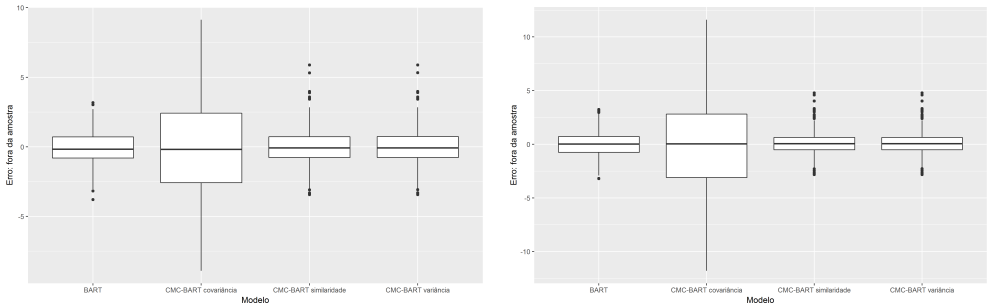
onde $\hat{y}$ é o vetor das predições do modelo sob análise dado um x observado. Os dados da FTF foram simulados para cada combinação de dimensão ($p = 10$, 20, 100 e 1000) e tamanho da amostra ($n = 5500$ e 11000). Esses dados foram divididos em 11 partes, sendo uma delas utilizada como conjunto de teste e as demais como treino, isto é, quando $n = 5500$ (ou 11000) tem-se 500 (ou 1000) observações escolhidas ao acaso formando o conjunto de teste e as outras 5000 (ou 10000) observações utilizadas como conjunto de treino. Caso o modelo ajustado seja um CMC-BART, essas 5000 observações de treino são divididas em 10 partes, de forma que cada *shard* recebe $n_s = 500$ para treinar o modelo e gera as predições para o conjunto de teste, que é o mesmo para todos os *shards*. Assim, todo RMSE apresentado é de predições fora-da-amostra, já que os valores a serem preditos não são observações que se encontravam no conjunto de treino. Dentre todas combinações de hiperparâmetros foi escolhido o modelo com menor RMSE para cada tamanho de árvore ($m =$ 5, 10, 20, 100 e 200) e dimensão ($p =$ 10, 20, 100 e 1000), e também, foi analisado o tempo de execução, bem como a correlação entre a $E(Y|x)$ e as predições dos modelos BART e CMC-BART para diferentes tipos de agregação.

Desempenho de predição do CMC-BART com FTF com $p = 10$

Para a FTF com $p = 10$, como mostrado na Figura 3.5, os erros se comportam de maneira esperada. A agregação pela matriz de covariância não se mostrou adequada para agregação, o que é esperado, dado não estar se formando um consenso para um parâmetro e sim para a distribuição preditiva. Os modelos mais maior m possuem melhor ajuste, e aumentar a amostra em cada *shard* também diminui os erros. Comparando o BART com o CMC-BART, percebe-se que apesar de o BART produzir um ajuste com menos *outliers*, a distribuição dos erros é bem similar entre ambos modelos, principalmente quando $n_s = 1000$.

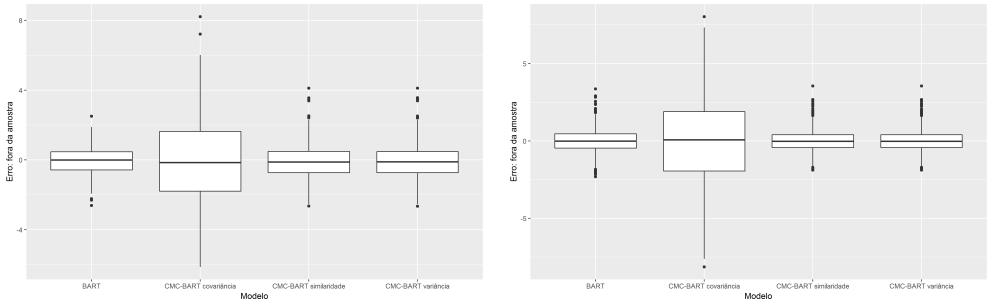
Na Tabela 3.5, pode-se observar que para modelos com $m = 5$, o RMSE do CMC-BART, com agregação por similaridade ou variância, é menor do que o do BART, quando temos um $n_s = 1000$. No entanto, quando $n_s = 500$, o BART ainda tem melhor RMSE. Um motivo para o melhor desempenho do CMC-BART quando se aumenta o tamanho da amostra é levantado por Scott *et al.* [SBB+16], pois o algoritmo para ajuste do BART encontra problemas de *mixing* conforme a amostra cresce.

O problema de *mixing* ainda é agravado pelo pequeno número de árvores no modelo. O CMC-BART tem uma posteriori com *mixing* mais rápido, enquanto o BART acaba sendo mais lento. Na Tabela 3.6 percebe-se que a correlação entre o $E(Y|x)$ e CMC-BART com agregação por similaridade ou variância possui maior correlação que o BART quando se tem $n_s = 1000$, fato que ratifica a observação dos autores. Por fim, na Tabela 3.7, verifica-se que o tempo de execução dos modelos CMC-BART pode ser até 4,8 vezes menor do que o do BART tradicional. Assim, para o CMC-BART com $m = 5$, isto é, um modelo com poucas árvores, tem-se melhor predição que o BART para amostras grandes, devido ao melhor *mixing*. Além disso, os modelos com poucas árvores possuem melhor capacidade de selecção de variáveis. Os hiperparâmetros com melhor desempenho são aqueles que priorizam árvores mais profundas, já que para $\alpha =$ 0,95 e $\beta = 1$, temos 1, 2, 3, 4 e ≥ 5 nós



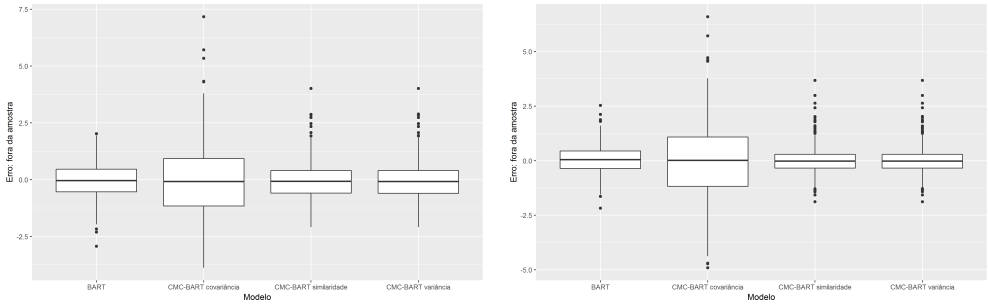
(a) $m = 5$ *e* $n_s = 500$

(b) $m = 5$ *e* $n_s = 1000$



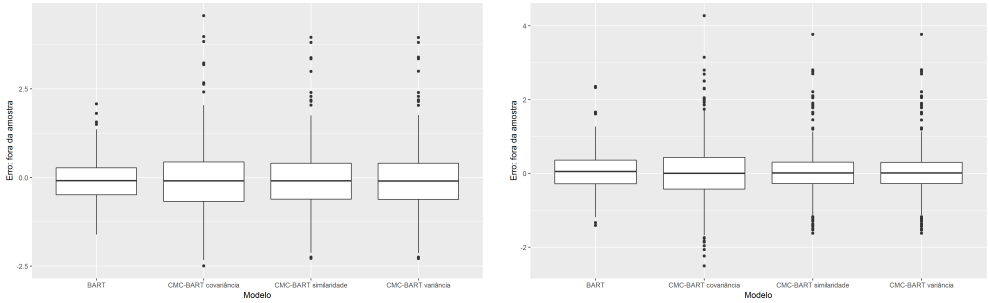
(c) $m = 10$ *e* $n_s = 500$

(d) $m = 10$ *e* $n_s = 1000$



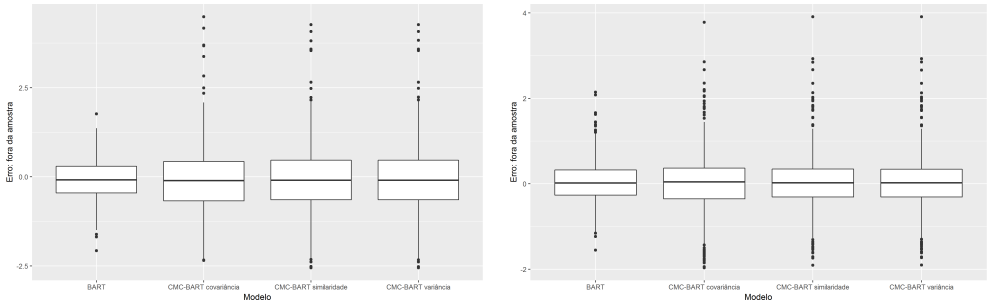
(e) $m = 20$ *e* $n_s = 500$

(f) $m = 20$ *e* $n_s = 1000$



(g) $m = 100$ *e* $n_s = 500$

(h) $m = 100$ *e* $n_s = 1000$



(i) $m = 200$ *e* $n_s = 500$

(j) $m = 200$ *e* $n_s = 1000$

Figura 3.5: *Boxplot dos erros fora da amostra da distribuição preditiva para a FTF com p = 10. Comparação entre o BART e o CMC-BART com os diferentes métodos de consenso. Note que a escala em todos as figuras é a mesma para facilitar a comparação dos erros.*

terminais com probabilidade a priori 0,05; 0,26; 0,22; 0,17 e 0,30, respectivamente. Dessa maneira, se o objetivo é realizar predições enquanto selecionam-se variáveis, é importante escolher uma priori que atribua uma maior probabilidade para árvores mais profundas.

Modelo	α	β	k	RMSE
BART	0,90	2	2	1,10
CMC-BART covariância	0,95	1	5	3,55
CMC-BART similaridade	0,90	1	3	1,27
CMC-BART variância	0,90	1	3	1,27

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,95	2	2	1,10
CMC-BART covariância	0,95	1	5	4,16
CMC-BART similaridade	0,95	1	3	0,96
CMC-BART variância	0,95	1	3	0,96

(b) $n_s = 1000$

Tabela 3.5: *Melhores modelos com $m = 5$ para FTF com $p = 10$.*

	$E(Y\|x)$	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
$E(Y\|x)$	1,00	0,97	0,89	0,97	0,97
BART	0,97	1,00	0,88	0,95	0,95
CMC-BART covariância	0,89	0,88	1,00	0,94	0,94
CMC-BART similaridade	0,97	0,95	0,94	1,00	1,00
CMC-BART variância	0,97	0,95	0,94	1,00	1,00

(a) $n = 5000$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,97	0,94	0,98	0,98
BART	0,97	1,00	0,92	0,97	0,97
CMC-BART covariância	0,94	0,92	1,00	0,96	0,96
CMC-BART similaridade	0,98	0,97	0,96	1,00	1,00
CMC-BART variância	0,98	0,97	0,96	1,00	1,00

(b) $n = 10000$

Tabela 3.6: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 5$ para FTF com $p = 10$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	2	16,22	1,00
CMC-BART covariância	0,95	1	5	19,27	0,84
CMC-BART similaridade	0,90	1	3	3,58	4,52
CMC-BART variância	0,90	1	3	3,57	4,54

(a) $n = 5000$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	2	35,47	1,00
CMC-BART covariância	0,95	1	5	67,69	0,52
CMC-BART similaridade	0,95	1	3	7,79	4,55
CMC-BART variância	0,95	1	3	7,40	4,80

(b) $n = 10000$

Tabela 3.7: *Comparativo do tempo de execução dos melhores modelos com $m = 5$ para FTF com $p = 10$.*

Ainda com $p = 10$, têm-se os resultados do RMSE das predições para $m = 10$ na Tabela 3.8. Novamente, pode-se perceber que os modelos com melhor desempenho foram aqueles em que a priori possui árvores mais profundas, principalmente nas situações em que se verifica uma amostra maior. Nota-se mais uma vez que o RMSE do CMC-BART é menor que o do BART quando $n_s = 1000$. Por outro lado, o modelo BART não teve dificuldade em atigir um *mixing* adequado em uma amostra maior, visto que sua correlação com $E(Y|x)$ é praticamente igual à do modelo CMC-BART por variância e similaridade. Conforme a Tabela 3.9, o aumento do número de árvores pode ter melhorado o *mixing* do modelo BART.

Finalmente, a razão entre o tempo de execução do BART e do CMC-BART aumentou com o aumento do número de árvores, o que fez com que a velocidade fosse até 5,95 vezes maior para a agregação por variância, conforme pode ser observado na Tabela 3.10. Como visto na subseção anterior, o modelo com $m = 10$ possui excelente capacidade de seleção de variáveis. Assim, caso o objetivo seja selecionar e estimar um modelo para predição simultaneamente à escolha de $\alpha = 0{,}90$ ou $0{,}95$ com $\beta = 1$, o CMC-BART pode ser uma boa opção para um número de dimensões reduzidos, no caso, $p = 10$.

Modelo	α	β	k	RMSE
BART	0,90	1	5	0,81
CMC-BART covariância	0,95	1	3	2,37
CMC-BART similaridade	0,90	1	2	0,96
CMC-BART variância	0,90	1	2	0,96

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	2	0,76
CMC-BART covariância	0,95	1	5	2,73
CMC-BART similaridade	0,95	1	2	0,67
CMC-BART variância	0,95	1	2	0,67

(b) $n_s = 1000$

Tabela 3.8: *Melhores modelos com $m = 10$ para FTF com $p = 10$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,94	0,98	0,98
BART	0,99	1,00	0,95	0,98	0,98
CMC-BART covariância	0,94	0,95	1,00	0,98	0,98
CMC-BART similaridade	0,98	0,98	0,98	1,00	1,00
CMC-BART variância	0,98	0,98	0,98	1,00	1,00

(a) $n = 5000$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,96	0,99	0,99
BART	0,99	1,00	0,95	0,98	0,98
CMC-BART covariância	0,96	0,95	1,00	0,97	0,97
CMC-BART similaridade	0,99	0,98	0,97	1,00	1,00
CMC-BART variância	0,99	0,98	0,97	1,00	1,00

(b) $n = 10000$

Tabela 3.9: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 10$ para FTF com $p = 10$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	5	23,23	1,00
CMC-BART covariância	0,95	1	3	19,21	1,21
CMC-BART similaridade	0,90	1	2	4,27	5,44
CMC-BART variância	0,90	1	2	4,23	5,49

(a) $n = 5000$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	2	46,68	1,00
CMC-BART covariância	0,95	1	5	68,64	0,68
CMC-BART similaridade	0,95	1	2	8,04	5,81
CMC-BART variância	0,95	1	2	7,84	5,95

(b) $n = 10000$

Tabela 3.10: *Comparativo do tempo de execução dos melhores modelos com $m = 10$ para FTF com $p = 10$.*

Já para os modelos com $m = 20$, o RMSE das predições (Tabela 3.11) comporta-se de forma semelhante ao dos modelos anteriores. Para $m = 5$ ou 10 e $n_s = 500$, o RMSE é menor para o BART do que para o CMC-BART, com agregação por similaridade ou variância, e o inverso acontece para $n_s = 1000$. Vale notar que na Tabela 3.26 (a) os modelos CMC-BART com melhor desempenho foram aqueles com prioris para árvores menos profundas que em (b). O mesmo pode ser observado anteriormente, na Tabela 3.8. A Tabela 3.12 mostra que a correlação entre $E(Y|x)$ e os modelos BART e CMC-BART é elevada, exceto para o CMC-BART com agregação por covariância, que se sai mal em relação aos demais. Outra vez, o modelo CMC-BART mostra-se mais veloz que o BART, sendo aquele até 5,89 vezes mais rápido. Os modelos com $m = 20$ são um compromisso entre seleção de variáveis e poder de predição. Perde-se um pouco da capacidade de seleção de variáveis em relação aos modelos com poucas árvores ($m \leq 10$) e ganha-se melhor poder de preditivo. Ressalta-se ainda que esses modelos são computacionalmente bem velozes, devido ao número reduzido de árvores.

Modelo	α	β	k	RMSE
BART	0,95	2	5	0,75
CMC-BART covariância	0,95	1	3	1,52
CMC-BART similaridade	0,90	1	2	0,81
CMC-BART variância	0,90	1	2	0,81

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,99	2	3	0,59
CMC-BART covariância	0,95	1	5	1,61
CMC-BART similaridade	0,95	1	3	0,56
CMC-BART variância	0,95	1	3	0,56

(b) $n_s = 1000$

Tabela 3.11: *Melhores modelos com $m = 20$ para FTF com $p = 10$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,96	0,99	0,99
BART	0,99	1,00	0,97	0,99	0,99
CMC-BART covariância	0,96	0,97	1,00	0,99	0,99
CMC-BART similaridade	0,99	0,99	0,99	1,00	1,00
CMC-BART variância	0,99	0,99	0,99	1,00	1,00

(a) $n = 5000$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,98	0,99	0,99
BART	0,99	1,00	0,98	0,99	0,99
CMC-BART covariância	0,98	0,98	1,00	0,99	0,99
CMC-BART similaridade	0,99	0,99	0,99	1,00	1,00
CMC-BART variância	0,99	0,99	0,99	1,00	1,00

(b) $n = 10000$

Tabela 3.12: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 20$ para FTF com $p = 10$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	5	29,61	1,00
CMC-BART covariância	0,95	1	3	20,75	1,43
CMC-BART similaridade	0,90	1	2	6,19	4,79
CMC-BART variância	0,90	1	2	5,92	5,00

(a) $n = 5000$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,99	2	3	63,35	1,00
CMC-BART covariância	0,95	1	5	71,29	0,89
CMC-BART similaridade	0,95	1	3	11,03	5,74
CMC-BART variância	0,95	1	3	10,75	5,89

(b) $n = 10000$

Tabela 3.13: *Comparativo do tempo de execução dos melhores modelos com $m = 20$ para FTF com $p = 10$.*

Para modelos com $m = 100$, a Tabela 3.14 apresenta os menores valores de RMSE dentre todos os modelos com $p = 10$. Para esses hiperparâmetros, o modelo CMC-BART não é melhor que o BART em nenhum dos casos. Entretanto, a diferença de RMSE entre os modelos é bem pequena, exceto para a agregação por covariância que mais uma vez se mostra um método ruim de agregação de preditivas. Por outro lado, na Tabela 3.15 em (a) tem-se que a correlação entre o $E(Y|x)$ e o modelo BART é 1,0, enquanto a correlação com o CMC-BART, com agregação por similaridade ou variância, é 0,99. Desse modo, percebe-se que os valores preditos pelo CMC-BART e o BART são muito próximos. Já em (b) a correlação entre o BART e o CMC-BART chega a 1,0. Voltando à Figura 3.5, nota-se que em (h) a distribuição dos erros é muito semelhante entre os dois modelos, exceto pelos *outliers*, que são o motivo para o RMSE ter ficado mais elevado para o CMC-BART por similaridade ou variância. Mais uma vez, o CMC-BART se mostra muito mais veloz que o BART, até mesmo para a agregação por covariância, sendo até 5,93 vezes mais rápido que o BART. Para os modelos com muitas árvores, os modelos com melhor desempenho foram aqueles com árvores menores, ou seja, com prioris mais regularizadoras, contrapondo-se aos modelos anteriores.

Modelo	α	β	k	RMSE
BART	0,90	2	5	0,59
CMC-BART covariância	0,99	1	2	0,93
CMC-BART similaridade	0,99	1	2	0,83
CMC-BART variância	0,99	1	2	0,83

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	5	0,48
CMC-BART covariância	0,99	1	2	0,69
CMC-BART similaridade	0,99	2	2	0,53
CMC-BART variância	0,99	2	2	0,53

(b) $n_s = 1000$

Tabela 3.14: *Melhores modelos com $m = 100$ para FTF com $p = 10$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,98	0,99	0,99
BART	0,99	1,00	0,99	0,99	0,99
CMC-BART covariância	0,98	0,99	1,00	1,00	1,00
CMC-BART similaridade	0,99	0,99	1,00	1,00	1,00
CMC-BART variância	0,99	0,99	1,00	1,00	1,00

(a) $n = 5000$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	1,00	0,99	0,99	0,99
BART	1,00	1,00	0,99	1,00	1,00
CMC-BART covariância	0,99	0,99	1,00	1,00	1,00
CMC-BART similaridade	0,99	1,00	1,00	1,00	1,00
CMC-BART variância	0,99	1,00	1,00	1,00	1,00

(b) $n = 10000$

Tabela 3.15: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 100$ para FTF com $p = 10$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	101,18	1,00
CMC-BART covariância	0,99	1	2	39,17	2,58
CMC-BART similaridade	0,99	1	2	24,68	4,10
CMC-BART variância	0,99	1	2	24,74	4,09

(a) $n = 5000$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	194,11	1,00
CMC-BART covariância	0,99	1	2	96,39	2,01
CMC-BART similaridade	0,99	2	2	32,86	5,91
CMC-BART variância	0,99	2	2	32,76	5,93

(b) $n = 10000$

Tabela 3.16: *Comparativo do tempo de execução dos melhores modelos com $m = 100$ para FTF com $p = 10$.*

Por fim, para a FTF com $p = 10$, têm-se os modelos com $m = 200$. A Tabela 3.17 apresenta os RMSE de cada modelo. Note que esses RMSE dos modelos CMC-BART são maiores que os apresentados na Tabela 3.14. O único modelo que apresenta melhor RMSE é o BART para amostra com $n_s = 1000$. Para esse número de árvores, o modelo BART supera o CMC-BART em qualquer método de agregação. Porém, na Tabela 3.18 verifica-se que a diferença entre as estimativas não é grande, dados os níveis de correlação entre o BART e o CMC-BART por similaridade ou variância, principalmente em (b). Cabe salientar que, apesar de o CMC-BART ser menos preciso nas predições que o BART, os ganhos de velocidade podem compensar tal diferença. Como visto na Tabela 3.19, o CMC-BART pode ser até 5,76 vezes mais rápido que o BART. Assim como para $m = 100$, os modelos com muitas árvores acabam tendo melhor desempenho quando as prioris regularizam mais a altura das árvores.

Modelo	α	β	k	RMSE
BART	0,90	1	5	0,59
CMC-BART covariância	0,99	1	2	0,94
CMC-BART similaridade	0,99	1	2	0,91
CMC-BART variância	0,99	1	2	0,91

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,95	2	5	0,46
CMC-BART covariância	0,99	1	2	0,62
CMC-BART similaridade	0,90	1	2	0,57
CMC-BART variância	0,90	1	2	0,57

(b) $n_s = 1000$

Tabela 3.17: *Melhores modelos com $m = 200$ para FTF com $p = 10$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,98	0,98	0,98
BART	0,99	1,00	0,99	0,99	0,99
CMC-BART covariância	0,98	0,99	1,00	1,00	1,00
CMC-BART similaridade	0,98	0,99	1,00	1,00	1,00
CMC-BART variância	0,98	0,99	1,00	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	1,00	0,99	0,99	0,99
BART	1,00	1,00	1,00	1,00	1,00
CMC-BART covariância	0,99	1,00	1,00	1,00	1,00
CMC-BART similaridade	0,99	1,00	1,00	1,00	1,00
CMC-BART variância	0,99	1,00	1,00	1,00	1,00

(b) $n_s = 1000$

Tabela 3.18: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 200$ para FTF com $p = 10$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	5	224,60	1,00
CMC-BART covariância	0,99	1	2	62,93	3,57
CMC-BART similaridade	0,99	1	2	48,44	4,64
CMC-BART variância	0,99	1	2	48,46	4,64

(a) $n_s = 500$

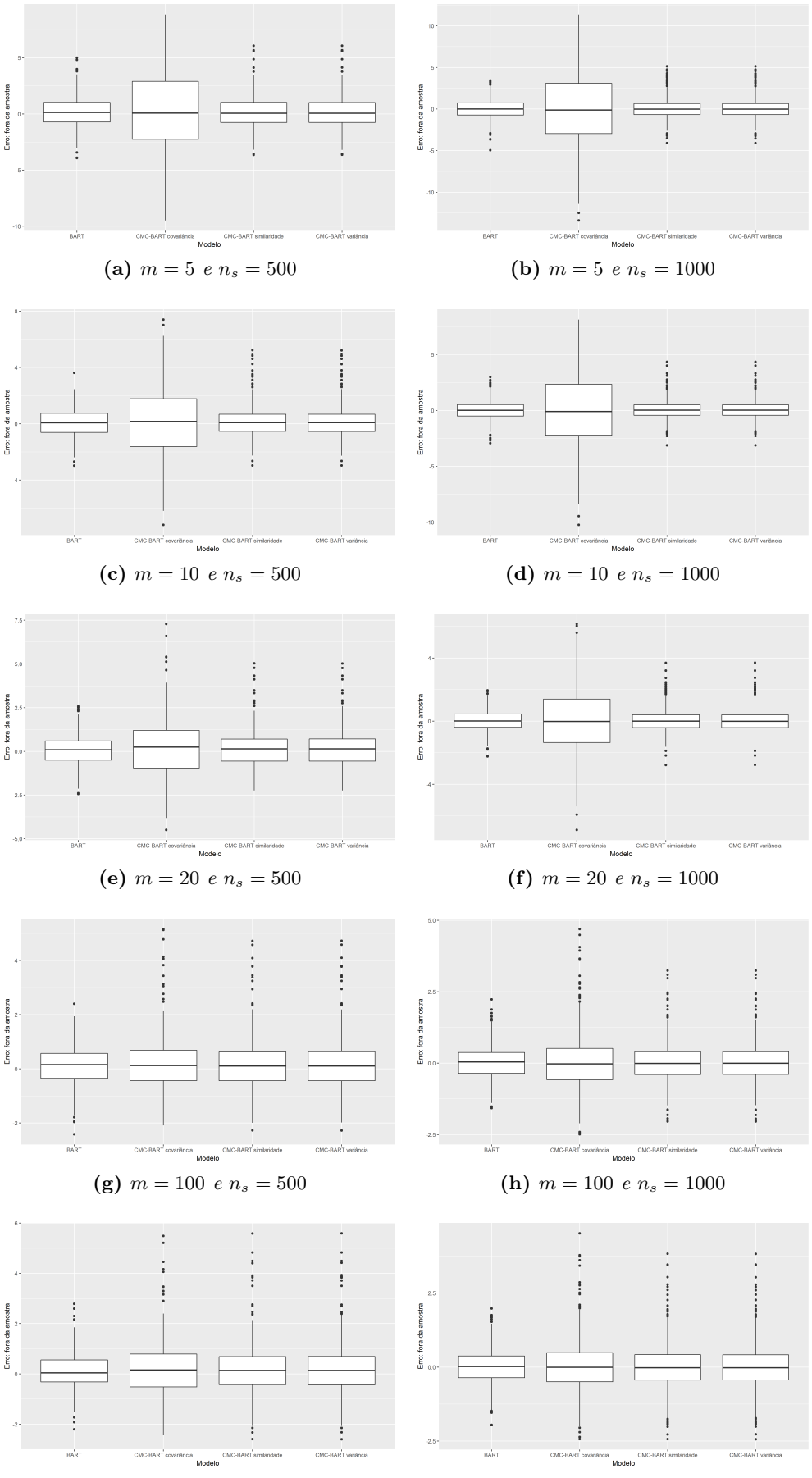
Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	5	386,83	1,00
CMC-BART covariância	0,99	1	2	133,96	2,89
CMC-BART similaridade	0,90	1	2	67,31	5,75
CMC-BART variância	0,90	1	2	67,11	5,76

(b) $n_s = 1000$

Tabela 3.19: *Comparativo do tempo de execução dos melhores modelos com $m = 200$ para FTF com $p = 10$.*

Desempenho de predição do CMC-BART com FTF com $p = 20$

Agora parte-se para a análise FTF com $p = 20$. Mais uma vez na figura 3.6 vizualiza-se que os erros diminuem com o aumento do número de árvores e com o aumento do tamanho das amostras de treino. O método de agregação pela matriz de covariância mostra-se muito ruim, produzindo erros muito maiores que os demais, tal como visto também em 3.5. A seguir averigua-se o CMC-BART para cada escolha de $m = 5$, 10, 20, 100, 200 e os resultados são contrapostos com o BART com o mesmo número de árvores.



(a) $m = 5$ *e* $n_s = 500$ **(b)** $m = 5$ *e* $n_s = 1000$

(c) $m = 10$ *e* $n_s = 500$ **(d)** $m = 10$ *e* $n_s = 1000$

(e) $m = 20$ *e* $n_s = 500$ **(f)** $m = 20$ *e* $n_s = 1000$

(g) $m = 100$ *e* $n_s = 500$ **(h)** $m = 100$ *e* $n_s = 1000$

(i) $m = 200$ *e* $n_s = 500$ **(j)** $m = 200$ *e* $n_s = 1000$

Figura 3.6: *Boxplot dos erros fora da amostra da distribuição preditiva para a FTF com p = 20. Comparação entre o BART e o CMC-BART com os diferentes métodos de consenso. Note que a escala em todos as figuras é a mesma para facilitar a comparação dos erros.*

Para $m = 5$, obtém-se um RMSE menor para o CMC-BART do que para o BART para $n_s = 500$, e ligeiramente menor para $n_s = 1000$, conforme a Tabela 3.20. Nela, observa-se que os hiperparâmetros atribuem alta probabilidade a priori para árvores mais profundas, que foram observadas também na posteriori. O aumento do número de dimensões traz maior dificuldade de predição, principalmente para um modelo com poucas árvores como esse. Tal fato pode ser observado ao se comparar o RMSE da Tabela 3.20 com os RMSE da 3.5. Nota-se um ligeiro acréscimo de erro médio observado. A correlação entre as predições do CMC-BART e o $E(Y|x)$ foi um pouco mais elevada que a do BART para todos os tamanhos de amostra (Tabela 3.22). A velocidade do CMC-BART, como esperado, é maior que a do BART (Tabela 3.22) chegando a ser 4,82 mais veloz para o consenso por similaridade e 4,93 para o consenso por variância.

Modelo	α	β	k	RMSE
BART	0,95	2	5	1,40
CMC-BART covariância	0,95	1	3	3,74
CMC-BART similaridade	0,99	1	2	1,35
CMC-BART variância	0,99	1	2	1,35

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	2	1,12
CMC-BART covariância	0,90	1	5	4,34
CMC-BART similaridade	0,95	1	3	1,11
CMC-BART variância	0,95	1	3	1,11

(b) $n_s = 1000$

Tabela 3.20: *Melhores modelos com $m = 5$ para FTF com $p = 20$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,96	0,92	0,97	0,97
BART	0,96	1,00	0,91	0,95	0,95
CMC-BART covariância	0,92	0,91	1,00	0,95	0,95
CMC-BART similaridade	0,97	0,95	0,95	1,00	1,00
CMC-BART variância	0,97	0,95	0,95	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,97	0,92	0,98	0,98
BART	0,97	1,00	0,91	0,96	0,96
CMC-BART covariância	0,92	0,91	1,00	0,95	0,95
CMC-BART similaridade	0,98	0,96	0,95	1,00	1,00
CMC-BART variância	0,98	0,96	0,95	1,00	1,00

(b) $n_s = 1000$

Tabela 3.21: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 5$ para FTF com $p = 20$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	5	16,71	1,00
CMC-BART covariância	0,95	1	3	19,14	0,87
CMC-BART similaridade	0,99	1	2	3,46	4,83
CMC-BART variância	0,99	1	2	3,33	5,01

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	2	34,48	1,00
CMC-BART covariância	0,90	1	5	66,77	0,52
CMC-BART similaridade	0,95	1	3	7,15	4,82
CMC-BART variância	0,95	1	3	6,99	4,93

(b) $n_s = 1000$

Tabela 3.22: *Comparativo do tempo de execução dos melhores modelos com $m = 5$ para FTF com $p = 20$.*

Já com $m = 10$, tem-se um melhor poder de predição do que quando $m = 5$ (Tabela 3.23). Verifica-se em (a) que o BART tem melhor desempenho que o CMC-BART; em (b), isto é, com $n_s = 1000$, o CMC-BART já passa a ter um RMSE menor. Vale ressaltar que ambos modelos têm melhor desempenho quando os hiperparâmetros priorizam árvores mais profundas, assim como foi observado quando $p = 10$. A correlação entre as predições e o $E(Y|x)$ é praticamente igual para o BART e para o CMC-BART. Por fim, a Tabela 3.25 mostra que o acréscimo de velocidade ao utilizar-se o CMC-BART é ainda maior com o aumento do número de dimensões dos dados.

Modelo	α	β	k	RMSE
BART	0,99	1	2	0,98
CMC-BART covariância	0,99	1	5	2,51
CMC-BART similaridade	0,95	1	5	1,09
CMC-BART variância	0,95	1	5	1,09

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,95	1	3	0,83
CMC-BART covariância	0,95	1	3	3,15
CMC-BART similaridade	0,95	1	2	0,78
CMC-BART variância	0,95	1	2	0,78

(b) $n_s = 1000$

Tabela 3.23: *Melhores modelos com $m = 10$ para FTF com $p = 20$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,98	0,95	0,98	0,98
BART	0,98	1,00	0,94	0,97	0,97
CMC-BART covariância	0,95	0,94	1,00	0,98	0,98
CMC-BART similaridade	0,98	0,97	0,98	1,00	1,00
CMC-BART variância	0,98	0,97	0,98	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,93	0,99	0,99
BART	0,99	1,00	0,92	0,98	0,98
CMC-BART covariância	0,93	0,92	1,00	0,95	0,95
CMC-BART similaridade	0,99	0,98	0,95	1,00	1,00
CMC-BART variância	0,99	0,98	0,95	1,00	1,00

(b) $n_s = 1000$

Tabela 3.24: *Correlação entre y e ŷ dos melhores modelos com m = 10 para FTF com p = 20.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,99	1	2	23,71	1,00
CMC-BART covariância	0,99	1	5	19,61	1,21
CMC-BART similaridade	0,95	1	5	4,46	5,32
CMC-BART variância	0,95	1	5	4,45	5,33

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	1	3	50,00	1,00
CMC-BART covariância	0,95	1	3	66,30	0,75
CMC-BART similaridade	0,95	1	2	7,96	6,28
CMC-BART variância	0,95	1	2	7,84	6,37

(b) $n_s = 1000$

Tabela 3.25: *Comparativo do tempo de execução dos melhores modelos com m = 10 para FTF com p = 20.*

Os modelos com $m = 20$ representam os modelos com razoável capacidade de seleção de variáveis e predição. Na Tabela 3.26 percebemos que a diferença entre o RMSE do BART e do CMC-BART é pequena, porém o BART tem menor RMSE tanto em (a) quanto em (b). Mais uma vez, os modelos com melhor desempenho são aqueles que atribuem maior densidade a árvores mais profundas. Já a correlação entre $E(Y|x)$ e os modelos foi maior que 0,98 para todos os modelos, exceto para o CMC-BART com agregação pela covariância (Tabela 3.27). Contudo, a velocidade do CMC-BART é até 7 vezes maior que a do que o modelo BART (quando $n_s = 1000$), conforme a Tabela 3.28.

Modelo	α	β	k	RMSE
BART	0,99	2	2	0,81
CMC-BART covariância	0,95	1	5	1,64
CMC-BART similaridade	0,90	1	2	0,98
CMC-BART variância	0,90	1	2	0,98

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,99	2	5	0,66
CMC-BART covariância	0,95	1	5	1,92
CMC-BART similaridade	0,95	1	2	0,68
CMC-BART variância	0,95	1	2	0,68

(b) $n_s = 1000$

Tabela 3.26: *Melhores modelos com $m = 20$ para FTF com $p = 20$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,96	0,98	0,98
BART	0,99	1,00	0,96	0,98	0,98
CMC-BART covariância	0,96	0,96	1,00	0,99	0,99
CMC-BART similaridade	0,98	0,98	0,99	1,00	1,00
CMC-BART variância	0,98	0,98	0,99	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,97	0,99	0,99
BART	0,99	1,00	0,97	0,99	0,99
CMC-BART covariância	0,97	0,97	1,00	0,98	0,98
CMC-BART similaridade	0,99	0,99	0,98	1,00	1,00
CMC-BART variância	0,99	0,99	0,98	1,00	1,00

(b) $n_s = 1000$

Tabela 3.27: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 20$ para FTF com $p = 20$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,99	2	2	28,62	1,00
CMC-BART covariância	0,95	1	5	21,43	1,34
CMC-BART similaridade	0,90	1	2	5,77	4,96
CMC-BART variância	0,90	1	2	5,76	4,97

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,99	2	5	71,19	1,00
CMC-BART covariância	0,95	1	5	69,90	1,02
CMC-BART similaridade	0,95	1	2	10,22	6,97
CMC-BART variância	0,95	1	2	10,12	7,03

(b) $n_s = 1000$

Tabela 3.28: *Comparativo do tempo de execução dos melhores modelos com $m = 20$ para FTF com $p = 20$.*

Para os modelos com $m = 100$, observa-se o melhor desempenho na predição por RMSE, tanto

para o BART quanto para o CMC-BART. Verifica-se na Tabela 3.29 que os modelos apresentam, em geral, os menores valores de RMSE dentre todas os valores testados de m, quando $p = 20$. Os melhores modelos BART apresentam $\alpha = 0{,}90$ e $\beta = 2$, tanto para (a) quanto para (b). Tais valores de hiperparâmetros produzem árvores menos profundas. Já os melhores CMC-BART foram aqueles com $\alpha = 0{,}95$ e $\beta = 1$, que geram árvores mais profundas a priori. Por sua vez, a correlação (Tabela 3.30) entre $E(Y|x)$ e o BART atinge 0,99 em (a) e (b); já entre $E(Y|x)$ e o CMC-BART, com agregação por variância ou similaridade, ele é de 0,98 em (a) e de 0,99 em (b). Nota-se ainda que os valores são bem elevados em ambos os modelos. Apesar de o CMC-BART ter RMSE pouco maior que o do modelo BART, uma das principais vantagens do *sharding* é o menor tempo de execução, conforme Tabela 3.31.

Modelo	α	β	k	RMSE
BART	0,90	2	5	0,71
CMC-BART covariância	0,95	1	2	1,01
CMC-BART similaridade	0,95	1	2	0,94
CMC-BART variância	0,95	1	2	0,94

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	5	0,55
CMC-BART covariância	0,99	1	2	0,86
CMC-BART similaridade	0,95	1	2	0,64
CMC-BART variância	0,95	1	2	0,64

(b) $n_s = 1000$

Tabela 3.29: *Melhores modelos com $m = 100$ para FTF com $p = 20$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,98	0,98	0,98
BART	0,99	1,00	0,98	0,99	0,99
CMC-BART covariância	0,98	0,98	1,00	1,00	1,00
CMC-BART similaridade	0,98	0,99	1,00	1,00	1,00
CMC-BART variância	0,98	0,99	1,00	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,99	0,99	0,99
BART	0,99	1,00	0,99	0,99	0,99
CMC-BART covariância	0,99	0,99	1,00	1,00	1,00
CMC-BART similaridade	0,99	0,99	1,00	1,00	1,00
CMC-BART variância	0,99	0,99	1,00	1,00	1,00

(b) $n_s = 1000$

Tabela 3.30: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 100$ para FTF com $p = 20$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	103,30	1,00
CMC-BART covariância	0,95	1	2	38,36	2,69
CMC-BART similaridade	0,95	1	2	23,91	4,32
CMC-BART variância	0,95	1	2	23,77	4,34

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	200,22	1,00
CMC-BART covariância	0,99	1	2	96,62	2,07
CMC-BART similaridade	0,95	1	2	36,74	5,45
CMC-BART variância	0,95	1	2	36,65	5,46

(b) $n_s = 1000$

Tabela 3.31: *Comparativo do tempo de execução dos melhores modelos com $m = 100$ para FTF com $p = 20$.*

Finalmente, para os modelos com $m = 200$, observa-se uma leve deterioração na capacidade de predição (RMSE) dos modelos (Tabela 3.32). Exceto para o BART com $n_s = 500$, todos os outros modelos tiveram um RMSE maior do que quando $m = 100$. Assim, vê-se que o CMC-BART começa a ter um sobreajuste para essa quantidade de árvores no modelo. Já a correlação entre $E(Y|x)$ e os modelos comportou-se da mesma maneira quando $m = 100$ (Tabela 3.33). O tempo de execução dos modelos CMC-BART é até 6,23 vezes mais rápido, como observado na Tabela 3.34.

Modelo	α	β	k	RMSE
BART	0,95	2	5	0,70
CMC-BART covariância	0,90	1	2	1,05
CMC-BART similaridade	0,99	1	2	1,05
CMC-BART variância	0,99	1	2	1,05

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	5	0,56
CMC-BART covariância	0,95	1	2	0,79
CMC-BART similaridade	0,99	2	2	0,72
CMC-BART variância	0,99	2	2	0,72

(b) $n_s = 1000$

Tabela 3.32: *Melhores modelos com $m = 200$ para FTF com $p = 20$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,98	0,98	0,98
BART	0,99	1,00	0,99	0,99	0,99
CMC-BART covariância	0,98	0,99	1,00	1,00	1,00
CMC-BART similaridade	0,98	0,99	1,00	1,00	1,00
CMC-BART variância	0,98	0,99	1,00	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,99	0,99	0,99
BART	0,99	1,00	0,99	0,99	0,99
CMC-BART covariância	0,99	0,99	1,00	1,00	1,00
CMC-BART similaridade	0,99	0,99	1,00	1,00	1,00
CMC-BART variância	0,99	0,99	1,00	1,00	1,00

(b) $n_s = 1000$

Tabela 3.33: *Correlação entre y e ŷ dos melhores modelos com m = 200 para FTF com p = 20.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	5	211,34	1,00
CMC-BART covariância	0,90	1	2	56,90	3,71
CMC-BART similaridade	0,99	1	2	48,88	4,32
CMC-BART variância	0,99	1	2	48,85	4,33

(a) $n_s = 500$

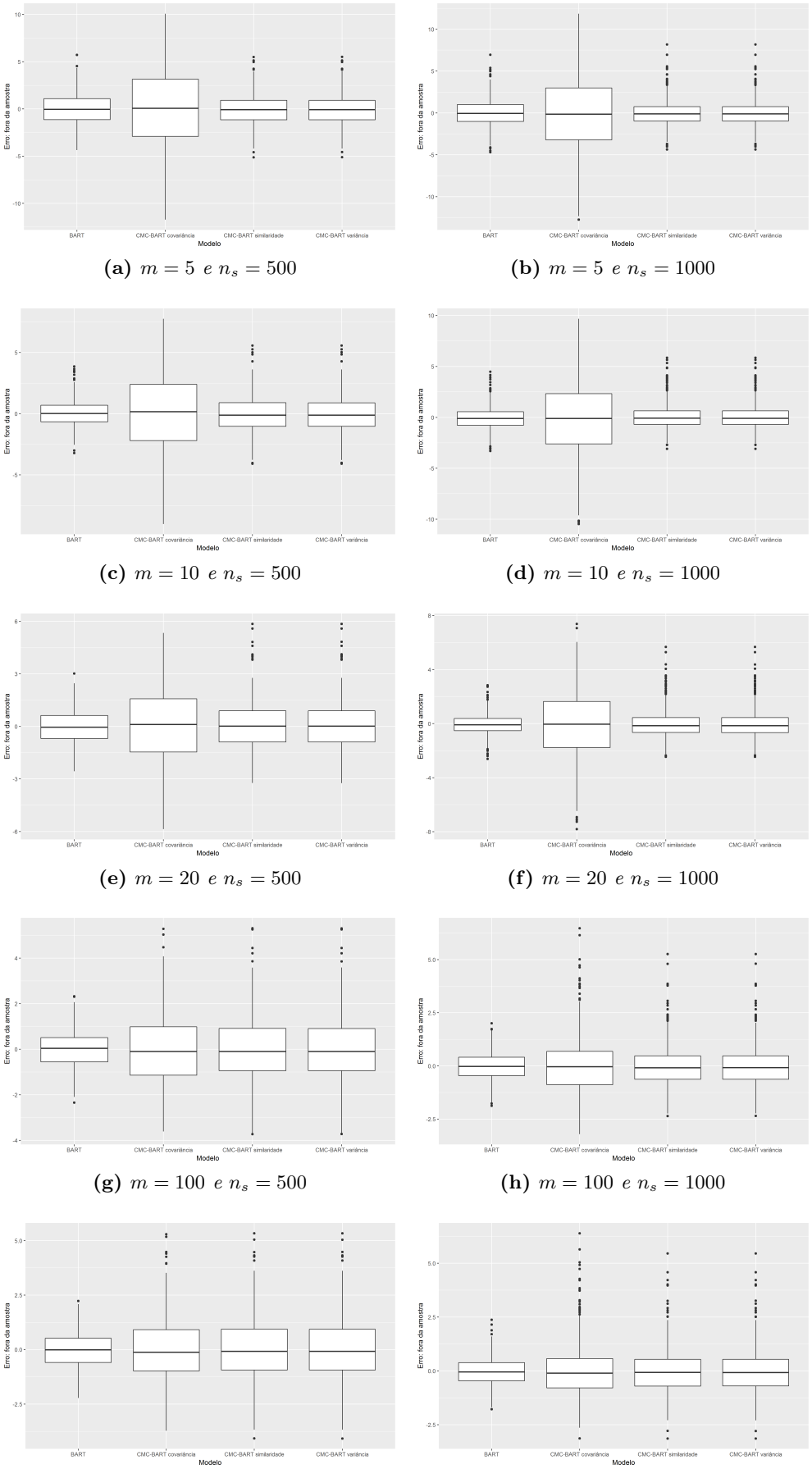
Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	394,06	1,00
CMC-BART covariância	0,95	1	2	130,30	3,02
CMC-BART similaridade	0,99	2	2	63,42	6,21
CMC-BART variância	0,99	2	2	63,30	6,23

(b) $n_s = 1000$

Tabela 3.34: *Comparativo do tempo de execução dos melhores modelos com m = 200 para FTF com p = 20.*

Investiga-se, a partir de agora, como o CMC-BART se comporta em uma dimensão ainda mais alta. Esse teste foi realizado com a FTF com $p = 100$. Na Figura 3.7 temos os *boxplots* para diversos valores de m e n_s. Mais uma vez, nota-se que a agregação pela matriz de covariância não produz boas predições, assim como para as outras dimensões já analisadas. Ainda, em geral os erros do BART são menores que os do CMC-BART, além de produzem menos *outliers*. A seguir, é feita a análise para cada valor de m.

Desempenho de predição do CMC-BART com FTF com $p = 100$



(a) $m = 5$ *e* $n_s = 500$ **(b)** $m = 5$ *e* $n_s = 1000$

(c) $m = 10$ *e* $n_s = 500$ **(d)** $m = 10$ *e* $n_s = 1000$

(e) $m = 20$ *e* $n_s = 500$ **(f)** $m = 20$ *e* $n_s = 1000$

(g) $m = 100$ *e* $n_s = 500$ **(h)** $m = 100$ *e* $n_s = 1000$

(i) $m = 200$ *e* $n_s = 500$ **(j)** $m = 200$ *e* $n_s = 1000$

Figura 3.7: *Boxplot dos erros fora da amostra da distribuição preditiva para a FTF com p = 100. Comparação entre o BART e o CMC-BART com os diferentes métodos de consenso. Note que a escala em todos as figuras é a mesma para facilitar a comparação dos erros.*

Na Tabela 3.35 encontram-se os modelos com melhor desempenho de RMSE dentre todos os modelos testados com $m = 5$. Em (a), verifica-se que o modelo BART com melhor desempenho é aquele que atribui probabilidade a priori para árvores pequenas; já para os modelos CMC-BART, os hiperparâmetros são para árvores um pouco mais profundas. Assim, observa-se uma diferença entre os modelos BART e CMC-BART quando a FTF tem dimensão menor. Nesses casos, os modelos com melhor desempenho são aqueles com hiperparâmetros que atribuem, a priori, maior probabilidade para árvores mais profundas, diferente do que ocorre para dimensões mais elevadas da FTF. Para $n_s = 500$, o modelo BART tem melhor desempenho, em termos de RMSE, que os modelos CMC-BART; todavia, a diferença de RMSE não é grande para os modelos com agregação por similaridade ou variância. Por sua vez, em (b), o CMC-BART é melhor que o BART, exceto quando a agregação é feita por covariância. Novamente, percebe-se que o modelo BART tem um mixing mais pobre quando há poucas árvores. Já a correlação entre $E(Y|x)$ e os modelos fica mais deteriorada para essa dimensão (Tabela 3.36). Em (a), tanto o BART quando o CMC-BART têm a mesma correlação com $E(Y|x)$, porém em (b) os modelos CMC-BART (variância ou similaridade) apresentam maior correlação com $E(Y|x)$, e ainda, o BART tem uma diminuição da correlação, evidenciando ainda mais os problemas de *mixing*. Por fim, os tempos de execução dos algoritmos são apresentados na Tabela 3.37. Mais uma vez, vê-se que o CMC-BART pode ser até 6,03 vezes mais rápido que os modelos BART.

Modelo	α	β	k	RMSE
BART	0,90	2	2	1,58
CMC-BART covariância	0,95	1	3	4,38
CMC-BART similaridade	0,90	1	2	1,68
CMC-BART variância	0,90	1	2	1,69

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	1	2	1,58
CMC-BART covariância	0,95	1	5	4,50
CMC-BART similaridade	0,90	2	2	1,38
CMC-BART variância	0,90	2	2	1,38

(b) $n_s = 1000$

Tabela 3.35: *Melhores modelos com $m = 5$ para FTF com $p = 100$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,95	0,91	0,95	0,95
BART	0,95	1,00	0,90	0,94	0,94
CMC-BART covariância	0,91	0,90	1,00	0,96	0,96
CMC-BART similaridade	0,95	0,94	0,96	1,00	1,00
CMC-BART variância	0,95	0,94	0,96	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,94	0,90	0,96	0,96
BART	0,94	1,00	0,88	0,93	0,93
CMC-BART covariância	0,90	0,88	1,00	0,95	0,95
CMC-BART similaridade	0,96	0,93	0,95	1,00	1,00
CMC-BART variância	0,96	0,93	0,95	1,00	1,00

(b) $n_s = 1000$

Tabela 3.36: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 5$ para FTF com $p = 100$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	2	15,10	1,00
CMC-BART covariância	0,95	1	3	19,07	0,79
CMC-BART similaridade	0,90	1	2	3,28	4,60
CMC-BART variância	0,90	1	2	3,16	4,78

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	2	36,50	1,00
CMC-BART covariância	0,95	1	5	66,76	0,55
CMC-BART similaridade	0,90	2	2	6,35	5,75
CMC-BART variância	0,90	2	2	6,05	6,03

(b) $n_s = 1000$

Tabela 3.37: *Comparativo do tempo de execução dos melhores modelos com $m = 5$ para FTF com $p = 100$.*

Analisam-se agora os modelos com $m = 10$. A Tabela 3.38 mostra que o RMSE dos melhores modelos novamente tem hiperparâmetros que controlam mais a profundidade das árvores, e também, que o modelo BART é melhor, em termos de RMSE, em (a) ou em (b). A correlação (Tabela 3.39) mostra que os modelos BART têm maior correlação com o treino que os modelos CMC-BART, tanto em (a) quanto em (b). Já o tempo de execução do CMC-BART fica 6,24 até menor que o BART (Tabela 3.40).

Modelo	α	β	k	RMSE
BART	0,90	1	2	1,08
CMC-BART covariância	0,95	1	3	3,28
CMC-BART similaridade	0,95	1	2	1,43
CMC-BART variância	0,95	1	2	1,43

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	3	1,06
CMC-BART covariância	0,95	1	5	3,61
CMC-BART similaridade	0,90	1	3	1,10
CMC-BART variância	0,90	1	3	1,11

(b) $n_s = 1000$

Tabela 3.38: *Melhores modelos com $m = 10$ para FTF com $p = 100$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,98	0,94	0,96	0,96
BART	0,98	1,00	0,93	0,96	0,96
CMC-BART covariância	0,94	0,93	1,00	0,97	0,97
CMC-BART similaridade	0,96	0,96	0,97	1,00	1,00
CMC-BART variância	0,96	0,96	0,97	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,98	0,94	0,97	0,97
BART	0,98	1,00	0,94	0,97	0,97
CMC-BART covariância	0,94	0,94	1,00	0,97	0,97
CMC-BART similaridade	0,97	0,97	0,97	1,00	1,00
CMC-BART variância	0,97	0,97	0,97	1,00	1,00

(b) $n_s = 1000$

Tabela 3.39: *Correlação entre y e ŷ dos melhores modelos com m = 10 para FTF com p = 100.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	2	21,88	1,00
CMC-BART covariância	0,95	1	3	18,72	1,17
CMC-BART similaridade	0,95	1	2	3,77	5,80
CMC-BART variância	0,95	1	2	3,77	5,81

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	3	48,18	1,00
CMC-BART covariância	0,95	1	5	67,78	0,71
CMC-BART similaridade	0,90	1	3	7,80	6,17
CMC-BART variância	0,90	1	3	7,72	6,24

(b) $n_s = 1000$

Tabela 3.40: *Comparativo do tempo de execução dos melhores modelos com m = 10 para FTF com p = 100.*

Já para $m = 20$, para esse número de árvores tem-se uma boa seleção de variáveis (como observado na subseção anterior), com capacidade de predição razoavelmente boa (Tabela 3.47). Além disso, a velocidade de execução para esse número de árvores é muito maior do que quando se aumenta para $m = 100$. Os hiperparâmetros dos modelos selecionados comportam-se de maneira mais estável para o CMC-BART, tanto para $n_=$ 500 quanto para $n_=$ 1000, $\alpha = 0{,}95$ e $\beta = 1$. Desta maneira, atribui-se maior probabilidade a priori para árvores mais profundas, porém para $n_s = 500$ os erros do modelo BART são muito menores que os erros dos modelos CMC-BART. É somente para $n_s = 1000$ que os erros do CMC-BART tornam-se um pouco menores. A correlação, exibida na Tabela 3.42, entre a $E(Y|x)$ e todos os modelos, é bastante elevada. E o tempo de execução dos modelos CMC-BART, mais uma vez, chega a ser até 6,68 vezes menor que o do BART (Tabela 3.49).

Modelo	α	β	k	RMSE
BART	0,99	1	2	0,94
CMC-BART covariância	0,90	1	5	2,18
CMC-BART similaridade	0,95	1	2	1,36
CMC-BART variância	0,95	1	2	1,36

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,95	1	2	0,74
CMC-BART covariância	0,95	1	3	2,45
CMC-BART similaridade	0,95	1	3	0,96
CMC-BART variância	0,95	1	3	0,96

(b) $n_s = 1000$

Tabela 3.41: *Melhores modelos com $m = 20$ para FTF com $p = 100$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,98	0,95	0,97	0,97
BART	0,98	1,00	0,96	0,97	0,97
CMC-BART covariância	0,95	0,96	1,00	0,99	0,99
CMC-BART similaridade	0,97	0,97	0,99	1,00	1,00
CMC-BART variância	0,97	0,97	0,99	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,96	0,98	0,98
BART	0,99	1,00	0,96	0,98	0,98
CMC-BART covariância	0,96	0,96	1,00	0,99	0,99
CMC-BART similaridade	0,98	0,98	0,99	1,00	1,00
CMC-BART variância	0,98	0,98	0,99	1,00	1,00

(b) $n_s = 1000$

Tabela 3.42: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 20$ para FTF com $p = 100$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,99	1	2	31,82	1,00
CMC-BART covariância	0,90	1	5	21,11	1,51
CMC-BART similaridade	0,95	1	2	5,76	5,52
CMC-BART variância	0,95	1	2	5,77	5,51

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	1	2	73,13	1,00
CMC-BART covariância	0,95	1	3	69,51	1,05
CMC-BART similaridade	0,95	1	3	11,10	6,59
CMC-BART variância	0,95	1	3	10,95	6,68

(b) $n_s = 1000$

Tabela 3.43: *Comparativo do tempo de execução dos melhores modelos com $m = 20$ para FTF com $p = 100$.*

Na Tabela 3.44 estão os modelos com $m = 100$; encontram-se aqui os modelos com menor RMSE

dentre todas as configurações de árvores testadas. Os hiperparâmetros dos melhores modelos para o CMC-BART são aqueles que atribuem probabilidade maior para árvores mais rasas ($\alpha = 0{,}95$ e $\beta = 2$). A diferença de RMSE do modelo BART e o do modelo CMC-BART é aceitável para $n_s = 500$ ou 1000. Além disso, as correlações (Tabela 3.45), são semelhantes às vistas para $m = 20$. O tempo de execução do CMC-BART é ainda melhor do que os vistos anteriormente, podendo ser até 7,47 vezes mais rápido que o modelo BART.

Modelo	α	β	k	RMSE
BART	0,99	2	5	0,77
CMC-BART covariância	0,95	1	2	1,45
CMC-BART similaridade	0,95	2	2	1,32
CMC-BART variância	0,95	2	2	1,33

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	1	5	0,63
CMC-BART covariância	0,99	1	2	1,20
CMC-BART similaridade	0,95	2	2	0,88
CMC-BART variância	0,95	2	2	0,88

(b) $n_s = 1000$

Tabela 3.44: *Melhores modelos com $m = 100$ para FTF com $p = 100$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,96	0,97	0,97
BART	0,99	1,00	0,97	0,98	0,98
CMC-BART covariância	0,96	0,97	1,00	1,00	1,00
CMC-BART similaridade	0,97	0,98	1,00	1,00	1,00
CMC-BART variância	0,97	0,98	1,00	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,97	0,98	0,98
BART	0,99	1,00	0,97	0,98	0,98
CMC-BART covariância	0,97	0,97	1,00	0,99	0,99
CMC-BART similaridade	0,98	0,98	0,99	1,00	1,00
CMC-BART variância	0,98	0,98	0,99	1,00	1,00

(b) $n_s = 1000$

Tabela 3.45: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 100$ para FTF com $p = 100$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,99	2	5	119,92	1,00
CMC-BART covariância	0,95	1	2	38,80	3,09
CMC-BART similaridade	0,95	2	2	20,12	5,96
CMC-BART variância	0,95	2	2	20,09	5,97

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	5	246,64	1,00
CMC-BART covariância	0,99	1	2	99,75	2,47
CMC-BART similaridade	0,95	2	2	33,34	7,40
CMC-BART variância	0,95	2	2	33,02	7,47

(b) $n_s = 1000$

Tabela 3.46: *Comparativo do tempo de execução dos melhores modelos com $m = 100$ para FTF com $p = 100$.*

Por fim, o modelo com $m = 200$, Tabela 3.47, tem RMSE mais elevado do que quando $m = 100$. Dessa forma, percebe-se que os modelos podem estar tendo um sobreajuste, assim como observado para o tamanho dimensional anterior, isto é, $p = 20$. Em (a), os modelos com melhor predição são aqueles com prioris para árvores mais profundas do que em (b). Já as correlações, apresentadas na Tabela 3.48 (a), mostram que o desempenho do CMC-BART foi pior do que quando se tem $m = 100$; por outro lado, as correlações apresentadas na Tabela 3.48 (b) são semelhantes ao do modelo com $m = 100$. Finalmente, a velocidade do CMC-BART chega a ser até 7,87 vezes maior que a do o BART, como visto na Tabela 3.49.

Modelo	α	β	k	RMSE
BART	0,90	1	3	0,82
CMC-BART covariância	0,90	1	2	1,45
CMC-BART similaridade	0,90	1	2	1,41
CMC-BART variância	0,90	1	2	1,41

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	5	0,65
CMC-BART covariância	0,90	1	2	1,11
CMC-BART similaridade	0,99	2	2	0,94
CMC-BART variância	0,99	2	2	0,94

(b) $n_s = 1000$

Tabela 3.47: *Melhores modelos com $m = 200$ para FTF com $p = 100$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,96	0,96	0,96
BART	0,99	1,00	0,97	0,98	0,98
CMC-BART covariância	0,96	0,97	1,00	1,00	1,00
CMC-BART similaridade	0,96	0,98	1,00	1,00	1,00
CMC-BART variância	0,96	0,98	1,00	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,99	0,97	0,98	0,98
BART	0,99	1,00	0,98	0,98	0,98
CMC-BART covariância	0,97	0,98	1,00	1,00	1,00
CMC-BART similaridade	0,98	0,98	1,00	1,00	1,00
CMC-BART variância	0,98	0,98	1,00	1,00	1,00

(b) $n_s = 1000$

Tabela 3.48: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 200$ para FTF com $p = 100$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	3	226,07	1,00
CMC-BART covariância	0,90	1	2	58,29	3,88
CMC-BART similaridade	0,90	1	2	44,22	5,11
CMC-BART variância	0,90	1	2	44,14	5,12

(a) $n_s = 500$

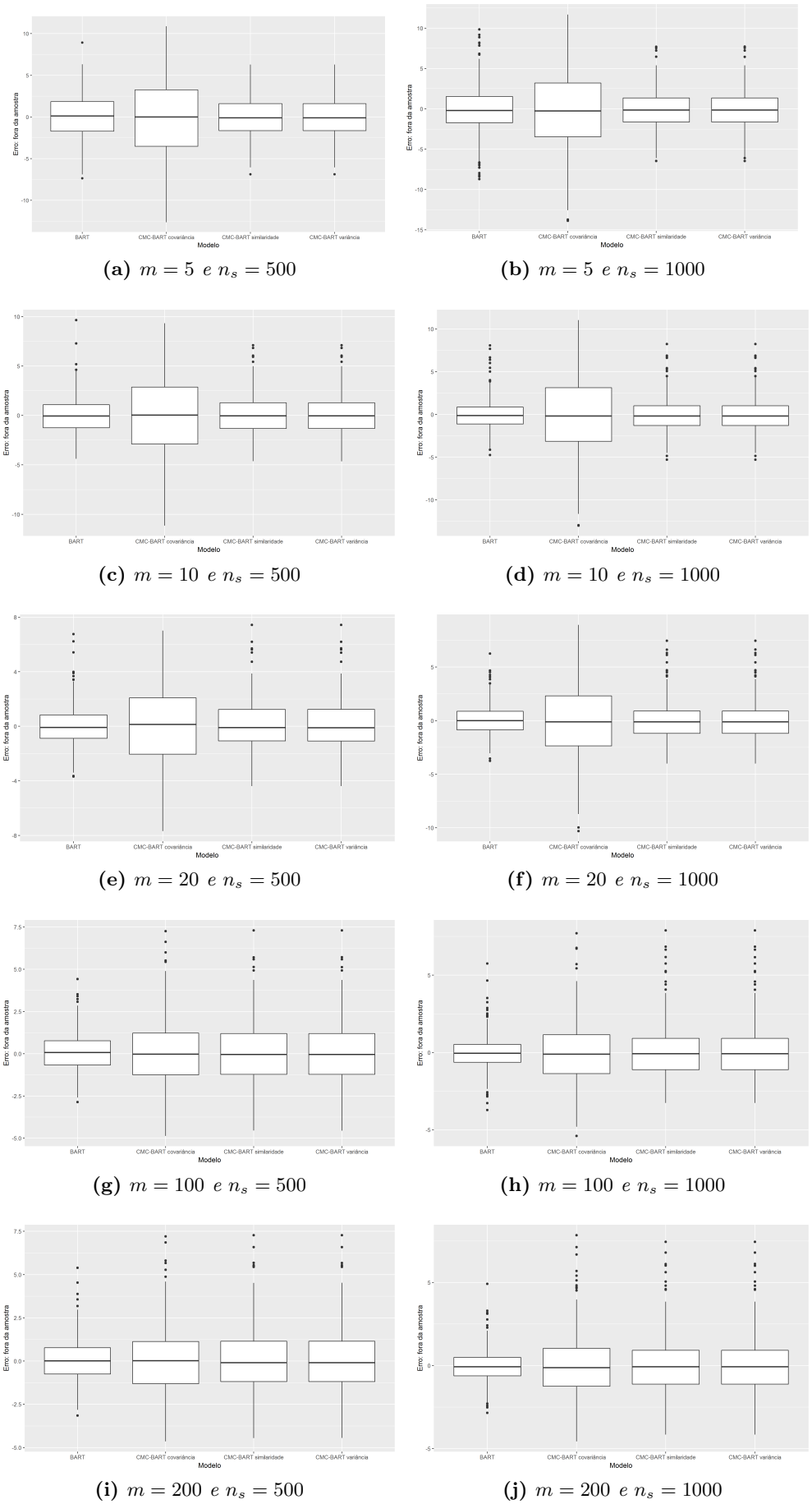
Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	519,06	1,00
CMC-BART covariância	0,90	1	2	129,38	4,01
CMC-BART similaridade	0,99	2	2	66,11	7,85
CMC-BART variância	0,99	2	2	65,96	7,87

(b) $n_s = 1000$

Tabela 3.49: *Comparativo do tempo de execução dos melhores modelos com $m = 200$ para FTF com $p = 100$.*

Ao se analisar a FTF com $p = 1000$, a maior dimensão analisada, observa-se que os erros dos modelo permanecem semelhantes aos das dimensões anteriores. A figura 3.8 representa o *boxplot* dos erros do melhor modelo para cada tamanho de árvore e tamanho da amostra para cada tipo de agregação. Os erros produzidos pelo CMC-BART com agregação pela matriz de covariância têm maior dispersão, enquanto o BART tem menor variância exceto para $m = 5$. Na sequência, foram analisados o RMSE, tempo de execução e a correlação para cada m.

Desempenho de predição do CMC-BART com FTF com $p = 1000$



(a) $m = 5$ *e* $n_s = 500$ **(b)** $m = 5$ *e* $n_s = 1000$

(c) $m = 10$ *e* $n_s = 500$ **(d)** $m = 10$ *e* $n_s = 1000$

(e) $m = 20$ *e* $n_s = 500$ **(f)** $m = 20$ *e* $n_s = 1000$

(g) $m = 100$ *e* $n_s = 500$ **(h)** $m = 100$ *e* $n_s = 1000$

(i) $m = 200$ *e* $n_s = 500$ **(j)** $m = 200$ *e* $n_s = 1000$

Figura 3.8: *Boxplot dos erros fora da amostra da distribuição preditiva para a FTF com p = 1000. Comparação entre o BART e o CMC-BART com os diferentes métodos de consenso. Note que a escala em todos as figuras é a mesma para facilitar a comparação dos erros.*

Na Tabela 3.50, nota-se que o RMSE é o maior até agora para as árvores de tamanho $m = 5$, e também que as árvores com melhor desempenho foram as com $\alpha = 0{,}90$ e $\beta = 1$ para os modelos CMC-BART e $\alpha = 0{,}95$ e $\beta = 1$ para o BART. Essas configurações de hiperparâmetros se traduzem em mais árvores profundas. Mais uma vez, o RMSE do CMC-BART com consenso por covariância é o mais elevado. Nota-se que para $m = 5$, isto é, para modelos com poucas árvores, o BART possui desempenho pior que o do CMC-BART por variância ou similaridade. Na Tabela 3.51 observam-se as correlações de $E(Y|x)$ para cada modelo. O BART tem correlação semelhante ao CMC-BART por covariância quando $n_s = 500$. O CMC-BART por similaridade ou variância possui as maiores correlações com a esperança condicional para $n_s = 500$. Já quando $n_s = 1000$, tanto o BART quando o CMC-BART por variância ou similaridade têm resultados semelhantes, porém o CMC-BART com agregação por covariância tem a correlação reduzida. Na Tabela 3.52 verifica-se que o CMC-BART por covariância tem velocidade menor que o próprio modelo BART; já as outras formas de agregação podem ser até 5,44 vezes mais rápidas que o BART.

Modelo	α	β	k	RMSE
BART	0,95	1	5	2,52
CMC-BART covariância	0,90	1	3	4,72
CMC-BART similaridade	0,90	1	3	2,34
CMC-BART variância	0,90	1	3	2,34

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	1	3	2,62
CMC-BART covariância	0,90	1	3	4,86
CMC-BART similaridade	0,90	2	2	2,18
CMC-BART variância	0,90	2	2	2,18

(b) $n_s = 1000$

Tabela 3.50: *Melhores modelos com $m = 5$ para FTF com $p = 1000$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,86	0,86	0,91	0,91
BART	0,86	1,00	0,85	0,88	0,88
CMC-BART covariância	0,86	0,85	1,00	0,96	0,96
CMC-BART similaridade	0,91	0,88	0,96	1,00	1,00
CMC-BART variância	0,91	0,88	0,96	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,85	0,75	0,92	0,92
BART	0,85	1,00	0,74	0,88	0,88
CMC-BART covariância	0,75	0,74	1,00	0,81	0,81
CMC-BART similaridade	0,92	0,88	0,81	1,00	1,00
CMC-BART variância	0,92	0,88	0,81	1,00	1,00

(b) $n_s = 1000$

Tabela 3.51: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 5$ para FTF com $p = 1000$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	1	5	17,70	1,00
CMC-BART covariância	0,90	1	3	22,02	0,80
CMC-BART similaridade	0,90	1	3	3,84	4,61
CMC-BART variância	0,90	1	3	3,78	4,68

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	3	36,17	1,00
CMC-BART covariância	0,90	1	3	69,97	0,52
CMC-BART similaridade	0,90	2	2	7,05	5,13
CMC-BART variância	0,90	2	2	6,65	5,44

(b) $n_s = 1000$

Tabela 3.52: *Comparativo do tempo de execução dos melhores modelos com $m = 5$ para FTF com $p = 1000$.*

Os modelos com $m = 10$ apresentados na Tabela 3.53 são aqueles com melhor desempenho em termos de RMSE. Para $n_s = 500$, os hiperparâmetros de regularização das árvores não mudaram em relação aos modelos com $m = 5$, porém para n_s 1000 os hiperparâmetros passaram a ser mais regularizadores das árvores, devido principalmente ao $\beta = 2$. O RMSE dos modelos BART é menor do que os do CMC-BART, para todos os tamanhos de amostras da FTF. No entanto, a diferença não é grande, principalmente quando se leva em consideração o aumento de velocidade, observado na Tabela 3.55. A Tabela 3.54 mostra que o BART e o CMC-BART com agregação por similaridade ou variância possuem correlações semelhantes com a $E(Y|x)$ para ambos tamanhos amostrais da FTF.

Modelo	α	β	k	RMSE
BART	0,95	1	2	1,81
CMC-BART covariância	0,95	1	3	4,03
CMC-BART similaridade	0,90	1	2	1,90
CMC-BART variância	0,90	1	2	1,90

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	5	1,54
CMC-BART covariância	0,95	1	3	4,53
CMC-BART similaridade	0,90	2	2	1,71
CMC-BART variância	0,90	2	2	1,71

(b) $n_s = 1000$

Tabela 3.53: *Melhores modelos com $m = 10$ para FTF com $p = 1000$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,93	0,89	0,94	0,94
BART	0,93	1,00	0,91	0,94	0,94
CMC-BART covariância	0,89	0,91	1,00	0,95	0,95
CMC-BART similaridade	0,94	0,94	0,95	1,00	1,00
CMC-BART variância	0,94	0,94	0,95	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,95	0,88	0,95	0,95
BART	0,95	1,00	0,91	0,96	0,96
CMC-BART covariância	0,88	0,91	1,00	0,93	0,93
CMC-BART similaridade	0,95	0,96	0,93	1,00	1,00
CMC-BART variância	0,95	0,96	0,93	1,00	1,00

(b) $n_s = 1000$

Tabela 3.54: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 10$ para FTF com $p = 1000$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	1	2	26,45	1,00
CMC-BART covariância	0,95	1	3	22,34	1,18
CMC-BART similaridade	0,90	1	2	5,64	4,69
CMC-BART variância	0,90	1	2	5,59	4,73

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	5	57,02	1,00
CMC-BART covariância	0,95	1	3	70,71	0,81
CMC-BART similaridade	0,90	2	2	9,32	6,12
CMC-BART variância	0,90	2	2	9,48	6,01

(b) $n_s = 1000$

Tabela 3.55: *Comparativo do tempo de execução dos melhores modelos com $m = 10$ para FTF com $p = 1000$.*

Ao elevar a quantidade de árvores para $m = 20$ (Tabela 3.56), tem-se uma melhoria no RMSE de todos os modelos. Para $n_s = 500$, tem-se árvores mais rasas para os modelos BART e CMC-BART por variância ou similaridade por conta do $\beta = 2$; já para o CMC-BART por covariância as árvores são mais profundas. Todavia, o RMSE desse modelo é o maior dentre todos os outros. Quando $n_s = 1000$, os melhores modelos já são aqueles com árvores mais profundas ($\beta = 1$). A correlação para $n_s = 500$ (Tabela 3.57 (a)) entre $E(Y|x)$ e os modelos é bastante alta, sendo a menor a do CMC-BART por covariância com 0,92 e a maior a do BART com 0,96. Por sua vez, a correlação quando $n_s = 1000$ (Tabela 3.57 (b)) da $E(Y|x)$ com os modelos BART e CMC-BART por variância ou similaridade é de 0,96. Já o tempo de execução dos algoritmos (Tabela 3.58) pode ser até 6,83 vezes menor quando é escolhido o CMC-BART por similaridade ou variância. Conforme o número de dimensões, cresce o desempenho dos modelos de consenso. Para esse número de dimensões e tamanho amostral, até mesmo o CMC-BART por covariância é mais rápido que o BART.

Modelo	α	β	k	RMSE
BART	0,90	2	3	1,39
CMC-BART covariância	0,90	1	2	2,83
CMC-BART similaridade	0,90	2	2	1,69
CMC-BART variância	0,90	2	2	1,69

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	1	5	1,30
CMC-BART covariância	0,90	1	3	3,38
CMC-BART similaridade	0,90	1	2	1,49
CMC-BART variância	0,90	1	2	1,49

(b) $n_s = 1000$

Tabela 3.56: *Melhores modelos com $m = 20$ para FTF com $p = 1000$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,96	0,93	0,95	0,95
BART	0,96	1,00	0,95	0,96	0,96
CMC-BART covariância	0,93	0,95	1,00	0,98	0,98
CMC-BART similaridade	0,95	0,96	0,98	1,00	1,00
CMC-BART variância	0,95	0,96	0,98	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,96	0,92	0,96	0,96
BART	0,96	1,00	0,95	0,97	0,97
CMC-BART covariância	0,92	0,95	1,00	0,97	0,97
CMC-BART similaridade	0,96	0,97	0,97	1,00	1,00
CMC-BART variância	0,96	0,97	0,97	1,00	1,00

(b) $n_s = 1000$

Tabela 3.57: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 20$ para FTF com $p = 1000$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	3	44,00	1,00
CMC-BART covariância	0,90	1	2	24,11	1,82
CMC-BART similaridade	0,90	2	2	7,62	5,77
CMC-BART variância	0,90	2	2	7,36	5,98

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	1	5	89,96	1,00
CMC-BART covariância	0,90	1	3	73,81	1,22
CMC-BART similaridade	0,90	1	2	13,18	6,83
CMC-BART variância	0,90	1	2	13,17	6,83

(b) $n_s = 1000$

Tabela 3.58: *Comparativo do tempo de execução dos melhores modelos com $m = 20$ para FTF com $p = 1000$.*

Aumentando o número de árvores para $m = 100$ (Tabela 3.59), temos os modelos CMC-BART

com melhor RMSE. Os CMC-BART com melhor desempenho foram aqueles onde $\beta = 2$. Já o BART tem melhor desempenho quando o $\beta = 1$ para $n_s = 500$ e $\beta = 2$ para $n_s = 1000$. A correlação dos modelos com a $E(Y|x)$ (Tabela 3.60) é elevada até mesmo para o CMC-BART por covariância, mostrando o poder dos modelos para predição mesmo em altas dimensões. Para esse número de árvores foi ainda maior a diferença de tempo de execução entre o BART e os modelos CMC-BART (Tabela 3.61). Os CMC-BART por variância ou similaridade chegaram a ser até 7,18 vezes mais rápidos que o BART.

Modelo	α	β	k	RMSE
BART	0,95	1	2	1,15
CMC-BART covariância	0,90	1	3	1,87
CMC-BART similaridade	0,90	2	2	1,70
CMC-BART variância	0,90	2	2	1,70

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,95	2	3	0,94
CMC-BART covariância	0,95	1	3	1,77
CMC-BART similaridade	0,90	2	2	1,44
CMC-BART variância	0,90	2	2	1,44

(b) $n_s = 1000$

Tabela 3.59: *Melhores modelos com $m = 100$ para FTF com $p = 1000$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,97	0,94	0,94	0,94
BART	0,97	1,00	0,96	0,96	0,96
CMC-BART covariância	0,94	0,96	1,00	0,99	0,99
CMC-BART similaridade	0,94	0,96	0,99	1,00	1,00
CMC-BART variância	0,94	0,96	0,99	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,98	0,95	0,96	0,96
BART	0,98	1,00	0,96	0,97	0,97
CMC-BART covariância	0,95	0,96	1,00	0,99	0,99
CMC-BART similaridade	0,96	0,97	0,99	1,00	1,00
CMC-BART variância	0,96	0,97	0,99	1,00	1,00

(b) $n_s = 1000$

Tabela 3.60: *Correlação entre y e $\hat{y}$ dos melhores modelos com $m = 100$ para FTF com $p = 1000$.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	1	2	178,02	1,00
CMC-BART covariância	0,90	1	3	47,88	3,72
CMC-BART similaridade	0,90	2	2	29,87	5,96
CMC-BART variância	0,90	2	2	29,89	5,96

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	3	347,70	1,00
CMC-BART covariância	0,95	1	3	120,81	2,88
CMC-BART similaridade	0,90	2	2	48,40	7,18
CMC-BART variância	0,90	2	2	48,35	7,19

(b) $n_s = 1000$

Tabela 3.61: *Comparativo do tempo de execução dos melhores modelos com $m = 100$ para FTF com $p = 1000$.*

Finalmente, para os modelos com $m = 200$ (Tabela 3.62), existe um aumento do RMSE para os modelos CMC-BART, ou seja, um pior desempenho do que quando $m = 100$. Porém, o BART tem melhor desempenho dentre todas as quantidades de árvores. Os hiperparâmetros dos melhores modelos todos apresentam $\beta = 2$. A correlação de $E(Y|x)$ com o BART e com o CMC-BART se mantém igual à do modelo com $m = 100$. A única vantagem dos modelos CMC-BART para esse número de árvores seria o ganho de velocidade, conforme observa-se na Tabela 3.64.

Modelo	α	β	k	RMSE
BART	0,95	2	5	1,13
CMC-BART covariância	0,95	2	2	1,86
CMC-BART similaridade	0,90	2	2	1,79
CMC-BART variância	0,90	2	2	1,79

(a) $n_s = 500$

Modelo	α	β	k	RMSE
BART	0,90	2	3	0,86
CMC-BART covariância	0,90	2	3	1,63
CMC-BART similaridade	0,90	2	3	1,49
CMC-BART variância	0,90	2	3	1,49

(b) $n_s = 1000$

Tabela 3.62: *Melhores modelos com $m = 200$ para FTF com $p = 1000$.*

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,97	0,94	0,94	0,94
BART	0,97	1,00	0,96	0,96	0,96
CMC-BART covariância	0,94	0,96	1,00	0,99	0,99
CMC-BART similaridade	0,94	0,96	0,99	1,00	1,00
CMC-BART variância	0,94	0,96	0,99	1,00	1,00

(a) $n_s = 500$

	E(Y|x)	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
E(Y|x)	1,00	0,98	0,95	0,96	0,96
BART	0,98	1,00	0,97	0,97	0,97
CMC-BART covariância	0,95	0,97	1,00	1,00	1,00
CMC-BART similaridade	0,96	0,97	1,00	1,00	1,00
CMC-BART variância	0,96	0,97	1,00	1,00	1,00

(b) $n_s = 1000$

Tabela 3.63: *Correlação entre y e ŷ dos melhores modelos com m = 200 para FTF com p = 1000.*

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,95	2	5	370,82	1,00
CMC-BART covariância	0,95	2	2	70,55	5,26
CMC-BART similaridade	0,90	2	2	53,92	6,88
CMC-BART variância	0,90	2	2	53,90	6,88

(a) $n_s = 500$

Modelo	α	β	k	Tempo (s)	Aumento de velocidade
BART	0,90	2	3	648,02	1,00
CMC-BART covariância	0,90	2	3	155,75	4,16
CMC-BART similaridade	0,90	2	3	95,33	6,80
CMC-BART variância	0,90	2	3	95,34	6,80

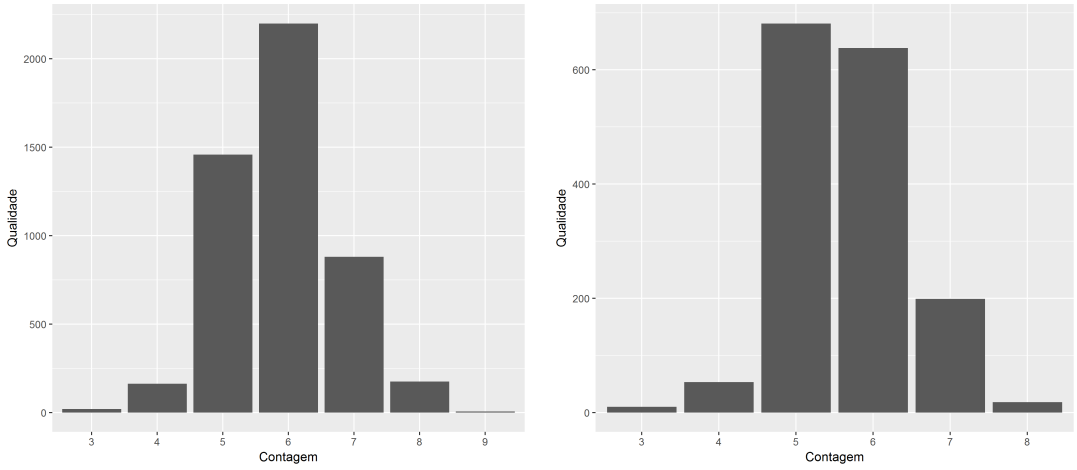
(b) $n_s = 1000$

Tabela 3.64: *Comparativo do tempo de execução dos melhores modelos com m = 200 para FTF com p = 1000.*

3.2 Dados reais de classificação

Esta seção trata do uso dos modelos BART e CMC-BART para classificação. Utilizando o conjunto de dados reais obtido em [CCA+09], foi realizado o ajuste dos modelos em estudo. Apresentam-se gráficos e medidas de resumo dos dados para melhor compreensão dos dados.

O conjunto de dados escolhido é sobre uma série de vinhos da região do Minho de Portugal, uma região produtora de vinho verde. Os dados foram coletados entre maio de 2004 e fevereiro de 2007, usando amostras apenas de vinhos com certificação de origem. As amostras foram avaliadas pelo órgão certificador "Comissão de Viticultura da Região dos Vinhos Verdes" (CVRVV). Para cada vinho foram realizadas análises físico-químicas e sensoriais. As análises sensoriais foram realizadas por pelo menos três indivíduos, que atribuíram ao vinho uma nota de 0 (muito ruim) a 10 (excelente) em um teste cego. Após as avaliações, cada vinho recebeu uma nota para sua qualidade, que é a mediana de todas as avaliações. Esses dados foram separados em dois conjuntos distintos: um para os vinhos brancos, com 4898 observações, e outro para os vinhos tintos, com 1599 observações. Na Figura 3.9 têm-se os histogramas da qualidade dos vinhos por tipo. Pouquíssimos vinhos receberam nota 3 ou 9, portanto qualquer algoritmo de classificação terá dificuldades para classificar os vinhos nessas faixas, devido ao não balanceamento dos dados.



(a) *Vinho branco* **(b)** *Vinho tinto*

Figura 3.9: *Histograma da qualidade aferida pelo análise sensorial para os vinhos tinto e branco.*

As variáveis fisíco-químicas avaliadas pelo CVRVV foram: acidez fixa (ácido tartárico) em g/dm^3, acidez volátil (ácido acético) em g/dm^3, ácido cítrico em g/dm^3, açúcar residual em g/dm^3, álcool em volume %, cloretos (cloreto de sódio) em g/dm^3, dióxido de enxofre livre em mg/dm^3, dióxido de enxofre total em mg/dm^3, pH, sulfatos (sulfato de potássio) em g/dm^3. Na figura 3.10 visualizam-se os *boxplots* de cada variável para cada tipo de vinho. Nos vinhos brancos nota-se que as variáveis com maiores variações com as notas são o teor de álcool e os cloretos. Já nos vinhos tintos, a acidez volátil, o volume de álcool e o ácido cítrico tiveram maior variação com a qualidade.

No artigo [CCA+09], foram estimados três modelos para os dados: regressão linear múltipla, rede neural e máquina de vetores de suporte (SVM). O Erro Médio Absoluto (MAE, na sigla em inglês) reportado pelos autores para os dados de vinho tinto foi de 0,50; 0,51 e 0,46, respectivamente. Já a acurácia do modelo foi de 59,1%; 59,1% e 62,4%, respectivamente. Para o vinho branco, o MAE foi de 0,59; 0,58 e 0,45, respectivamente. Por fim, a acurácia foi de 51,7%; 52,6% e 64,6%, respectivamente. Assim, o modelo com melhor desempenho foi o SVM. Os autores também reportaram as variáveis mais utilizadas pelo SVM para cada conjunto. Para os dados de vinho tinto, as quatro variáveis mais utilizadas foram: sulfatos, pH, dióxido de enxofre total e o volume de álcool, nessa ordem. Nos dados de vinho branco, as quatro variáveis mais utilizadas, em ordem, foram: sulfatos, álcool, açúcar residual e ácido cítrico.

Foram ajustados dois tipos de modelos, um BART e três CMC-BART um para cada tipo de consenso, sendo eles: covariância, variância e similaridade. O CMC-BART foi realizado com 4 *shards*; dessa forma os dados foram divididos em 5 partes, sendo uma parte o conjunto de teste e as demais conjuntos de treino, formando assim 4 *shards*. Na Tabela 3.65 pode-se verificar o número de vinhos por qualidade em cada divisão dos dados. Já para o BART foi utilizado o mesmo conjunto de teste e o conjunto de treino compreende todos os outros dados.

	3	4	5	6	7	8	9
Teste	3	33	292	436	178	38	0
Treino - Shard 1	3	30	314	410	200	20	2
Treino - Shard 2	4	34	288	447	171	35	1
Treino - Shard 3	5	37	295	434	169	38	1
Treino - Shard 4	5	29	268	471	162	44	1

(a) *Vinho branco*

	3	4	5	6	7	8
Teste	4	15	131	128	41	1
Treino - Shard 1	3	6	135	131	39	6
Treino - Shard 2	1	11	142	121	41	3
Treino - Shard 3	1	11	135	139	28	6
Treino - Shard 4	1	10	138	119	50	2

(b) *Vinho tinto*

Tabela 3.65: *Contagem do números de vinhos por qualidade em cada shard.*

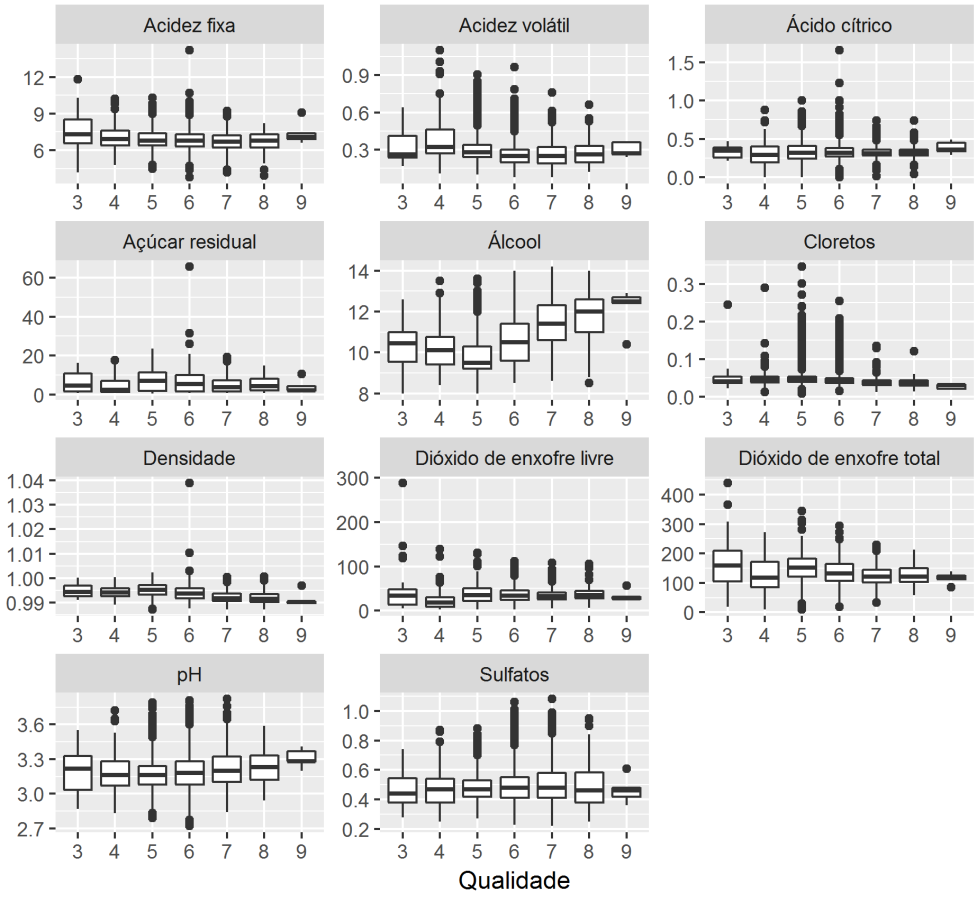
O CMC-BART foi realizado utilizando os seguintes hiperparâmetros: $m = 20$ (número de árvo-

res); para o vinho branco $k = 2$ e para o vinho tinto $k = 3$ (dispersão); $\alpha = 0,95$ (probabilidade base das árvores); $\beta = 1$ (regularizador das árvores), $\nu = 9$, $V = \frac{1}{2}\mathbb{I}_p$. Já para o BART foram escolhidos os hiperparâmetros: $m = 20$ (número de árvores); $k = 3$ (dispersão); $\alpha = 0,95$ (probabilidade base das árvores); $\beta = 1$ (regularizador das árvores), $\nu = 9$, $V = \frac{1}{2}\mathbb{I}_p$. Os hiperparâmetros foram escolhidos baseando-se nos experimentos realizados com os dados sintéticos da FTF quando $p = 10$, por ser o tamanho dimensional mais próximo dos dados de vinho verde.

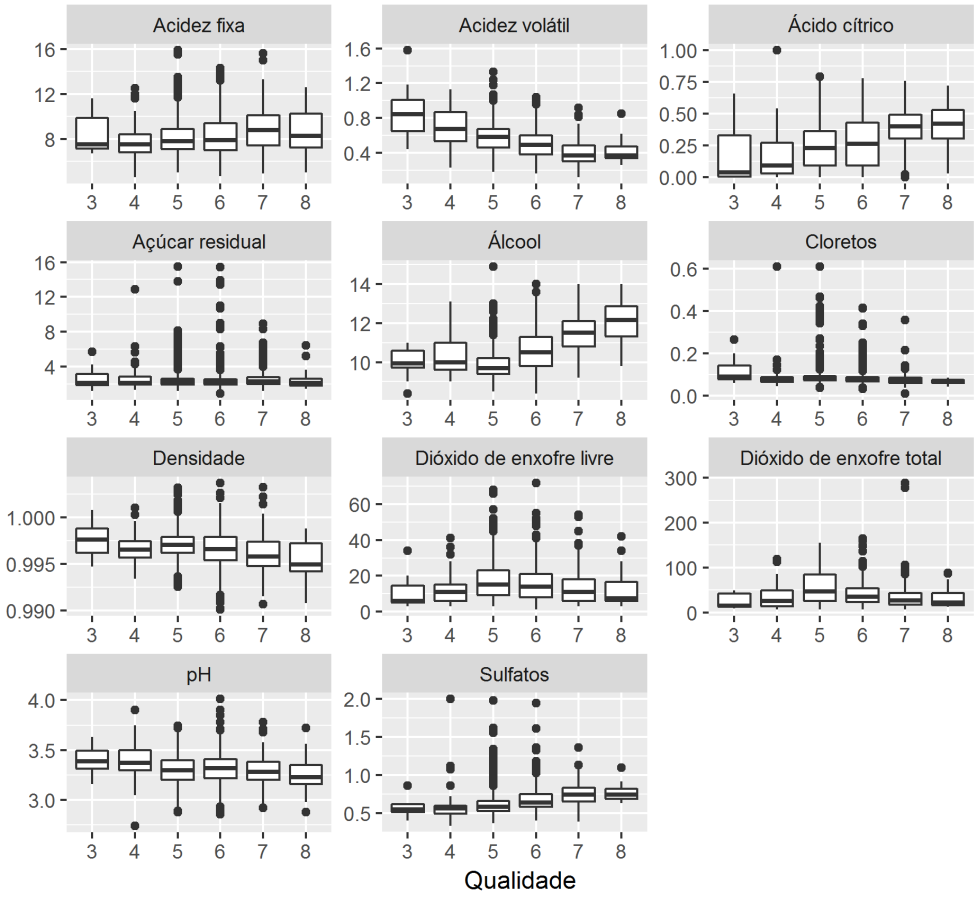
Todas as simulações, tanto para o BART quanto para o CMC-BART, foram realizadas com 500 amostras de *burn-in* e 1000 amostras para a posteriori. A partir da posteriori foram amostradas 1000 observações da distribuição preditiva para cada vinho no conjunto de treino, então calculou-se a mediana da amostra de predição para se achar a classe predita de cada vinho. No caso do CMC-BART, foram realizadas as agregações e calculada a amostra de consenso dos quatro *shards*. Obtidas as predições, foi calculado o Erro Absoluto Médio (MAE), dado pela equação

$$MAE = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j|, \tag{3.3}$$

onde y_j é a verdadeira classe e $\hat{y}_j$ é a classe predita pelo modelo avaliado. O tempo reportado de execução dos algoritmos é, no caso do BART, a soma do tempo gasto no passo de estimação e de predição; já no caso do CMC-BART, é a soma dos tempos nos passos de estimação, predição e agregação por consenso. Calculou-se também a matriz de confusão para cada modelo e a precisão por classe. A correlação entre $E(Y|x)$ e a mediana das predições de cada modelo foi utilizada para comparar os diversos modelos em análise. Também foi realizada a contagem do uso das variáveis em cada modelo. Foi avaliado também um *boxplot* dos erros de classificação e das matrizes de confusão.



(a) *Vinho branco*



(b) *Vinho tinto*

Figura 3.10: *Boxplot de cada variável por qualidade (nota da análise sensorial).*

Para o conjunto de dados sobre o vinho branco, o melhor modelo foi o CMC-BART por similaridade ou variância. Ambos tiveram desempenho praticamente igual (Tabela 3.66). É importante ressaltar que a diferença de MAE entre o BART e o CMC-BART por similaridade ou variância é praticamente zero, porém a taxa de acerto dos modelos de concenso são um pouco melhores. Além disso, o tempo computacional dos modelos CMC-BART é de 2,75 até 3,56 vezes melhor que o do BART. Já para o conjunto de dados sobre o vinho tinto, o melhor modelo é o BART, por uma diferença de 0,01 de MAE e uma taxa de acerto de 59,06%, isto é, 0,94% superior ao CMC-BART por similaridade ou variância. No entanto, o tempo de execução do CMC-BART é muito menor que o do BART, chegando a set até 3,29 vezes mais rápido. Para esses dados percebe-se que a aproximação do CMC-BART, pelos métodos de similaridade ou variância, ao BART é razoável para classificação, e com essa aproximação diminui o tempo de processsamento.

Modelo	Tempo (em s)	Tempo relativo	Ganho de vel.	MAE	Taxa de acerto (%)
BART	82,26	100,00	1,00	0,49	55,00 %
CMC-BART covarincia	29,96	36,42	2,75	0,51	53,87 %
CMC-BART similaridade	23,19	28,19	3,55	0,49	55,40 %
CMC-BART varincia	23,14	28,13	3,56	0,49	55,40 %

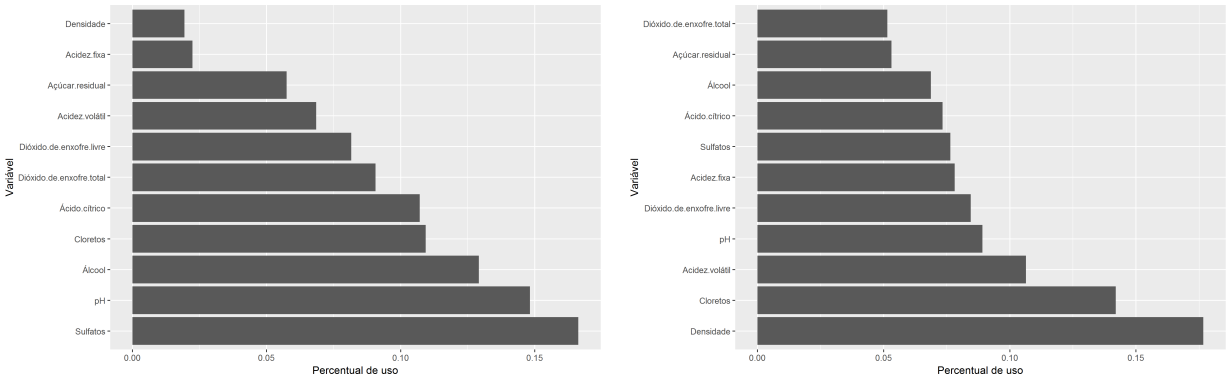
(a) *Vinho branco*

Modelo	Tempo (em s)	Tempo relativo	Ganho de vel.	MAE	Taxa de acerto (%)
BART	20,74	100,00	1,00	0,46	59,06%
CMC-BART covarincia	6,78	32,68	3,06	0,47	57,50 %
CMC-BART similaridade	6,42	30,97	3,23	0,47	58,12 %
CMC-BART varincia	6,31	30,42	3,29	0,47	58,12 %

(b) *Vinho tinto*

Tabela 3.66: *Tempo de execução e MAE para cada modelo.*

As quatro variáveis mais utilizadas pelo modelo BART foram para o vinho branco: sulfatos, pH, álcool e cloretos (Figura 3.11 (a)). Nota-se que nas variáveis reportadas em [CCA+09] apenas sulfato e álcool estão em comum. A variável açúcar residual foi uma das menos usadas no BART, mas, por outro lado, uma das mais utilizadas pelo SVM dos autores. Já para os vinhos tintos, as variáveis mais usadas pelo BART são: densidade, cloretos, acidez volátil e pH (Figura 3.11 (b)). Dessa forma, apenas pH é uma variável em comum com o modelo SVM reportado em [CCA+09].

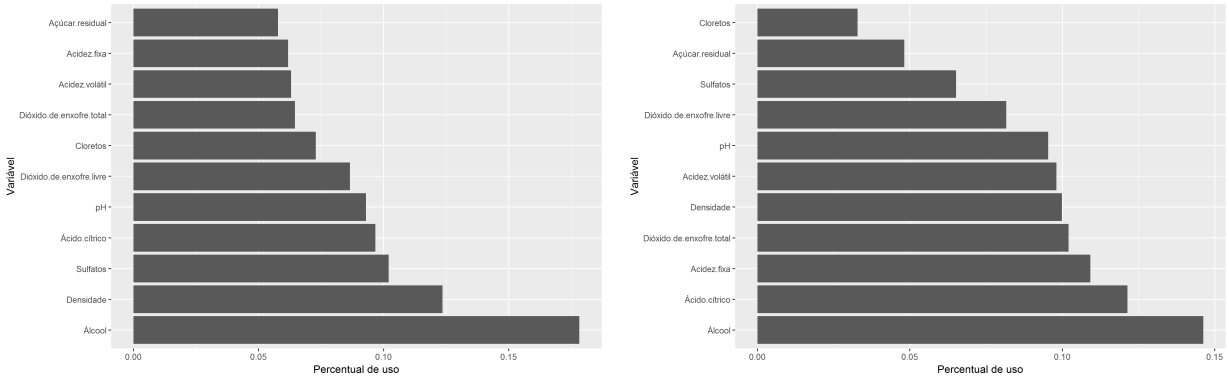


(a) *Vinho branco* **(b)** *Vinho tinto*

Figura 3.11: *Variáveis mais utilizadas pelo BART.*

Para os modelos CMC-BART as variáveis são um pouco diferentes das selecionadas pelo BART. As quatro mais usadas para o vinho branco foram: álcool, densidade, sulfatos e ácido cítrico (Figura 3.12 (a)). Sendo assim, temos três variáveis em comum com o SVM reportado pelos autores e duas

variáveis em comum com o BART. Nota-se que a variável menos empregada no CMC-BART, o açúcar residual, foi uma das mais usadas no SVM. Para os dados de vinho tinto, as quatro variáveis mais utilizadas foram: álcool, ácido cítrico, acidez fixa e dióxido de enxofre total (Figura 3.12 (b)). Logo, o CMC-BART tem apenas uma variável em comum com o SVM, o volume de álcool, e nenhuma variável em comum com o BART.



(a) *Vinho branco* **(b)** *Vinho tinto*

Figura 3.12: *Variáveis mais utilizadas pelo CMC-BART.*

A correlação para os dados de vinho branco entre $E(Y|x)$ e os modelos foi um pouco maior para o BART (0,54), já para o CMC-BART pro similaridade ou variância foi de 0,53, conforme Figura 3.67 (a). Já para o vinho tinto todos os modelos ficaram bem próximos, até mesmo o CMC-BART com agregação pela matriz de covariância, como observa-se na Figura 3.67 (b).

	$E(Y\|x)$	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
$E(Y\|x)$	1,00	0,54	0,49	0,53	0,53
BART	0,54	1,00	0,74	0,79	0,79
CMC-BART covarincia	0,49	0,74	1,00	0,86	0,86
CMC-BART similaridade	0,53	0,79	0,86	1,00	1,00
CMC-BART varincia	0,53	0,79	0,86	1,00	1,00

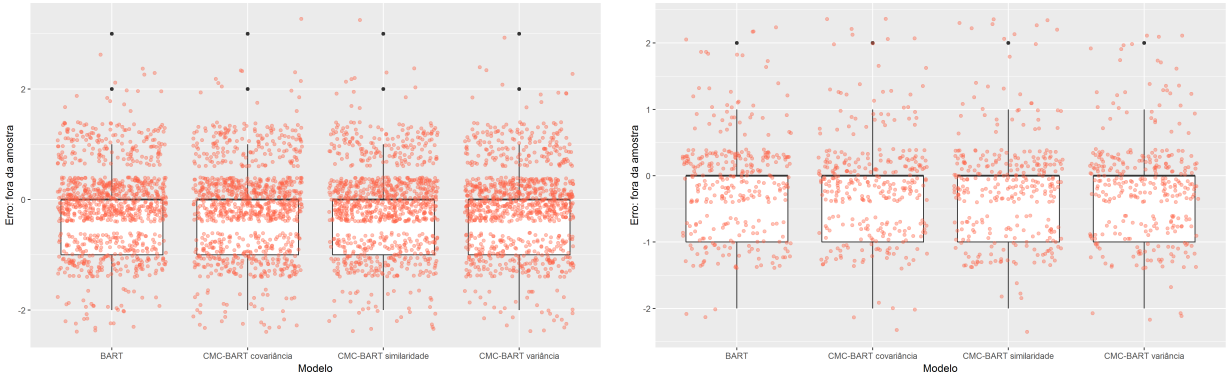
(a) *Vinho branco*

	$E(Y\|x)$	BART	CMC-BART cov.	CMC-BART sim.	CMC-BART var.
$E(Y\|x)$	1,00	0,49	0,47	0,48	0,48
BART	0,49	1,00	0,81	0,81	0,81
CMC-BART covarincia	0,47	0,81	1,00	0,91	0,91
CMC-BART similaridade	0,48	0,81	0,91	1,00	1,00
CMC-BART varincia	0,48	0,81	0,91	1,00	1,00

(b) *Vinho tinto*

Tabela 3.67: *Correlação entre $E(Y|x)$ e as predições para cada modelo.*

A Figura 3.13 mostra o *boxplot* dos erros de predição para cada modelo, em (a) para o vinho branco e em (b) para o vinho tinto. Os pontos em vermelho representam o ajuste de cada observação, x_i, com uma variação aleatória em sua localização a fim de evitar sobreposição de cada ponto. Dessa forma, evita-se um *overplotting*. Essa Figura representa as matrizes de confusão que seram apresentadas abaixo para cada modelo. Nota-se que existem muitos pontos que não foram devidamente classificados. Todavia, os diferentes modelos têm comportamento semelhante, portanto as aproximações do CMC-BART ao BART parecem razoáveis.



(a) *Vinho branco* **(b)** *Vinho tinto*

Figura 3.13: *Boxplot da dispersão dos erros para cada modelo.*

A matriz de confusão do modelo BART (Tabela 3.68) ilustra a dificuldade do modelo em classificar as classes com menor ocorrência. Observa-se que as classes de vinho com notas 3, 4, 7, 8 tiveram menor acerto, tanto para o vinho branco quanto para o vinho tinto. A precisão de cada classe também é apresentada: verifica-se que as classes com maior precisão para os dados de vinho branco são as com mais observações: 5 e 6. Já para o vinho tinto a classe com maior precisão é a classe 7, que não é a classe mais frequente nos dados observados, e as classes mais observadas permanecem com boa precisão.

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	1	2	0	0	0
4	0	1	26	6	0	0	0
5	0	1	175	111	5	0	0
6	0	0	93	304	39	0	0
7	0	0	6	113	59	0	0
8	0	0	0	21	17	0	0
9	0	0	0	0	0	0	0
Preciso	-	50,00	58,13	54,57	49,16	-	-

(a) *Vinho branco*

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	4	0	0	0	0
4	0	0	10	5	0	0	0
5	0	0	114	16	1	0	0
6	0	0	58	68	2	0	0
7	0	0	6	28	7	0	0
8	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	59,37	57,62	70,00	-	-

(b) *Vinho tinto*

Tabela 3.68: *Matriz de confusão para as predições do modelo BART.*

Ao se analisar a matriz de confusão do modelo CMC-BART com agregação pela matriz de covariância (Tabela 3.69) nota-se uma melhor precisão na classificação dos vinhos brancos (a) para as classes 5 e 7 do que do modelo BART, porém a precisão para a classe 6 é pior, o que acaba levando a uma taxa de acerto inferior. Isso se deve principalmente a esse nível de qualidade ser o mais comum na amostra analisada. Observa-se também que a classificação para o vinho tinto é maior para a classe 5 do que a do BART, e bem pior para a classe 6. Além disso, percebe-se que o CMC-BART não classificou nenhum dos vinhos nas classes: 3, 4, 8 e 9, para os vinhos brancos; e 3, 4, 7, 8 e 9 para os vinhos tintos. Dessa maneira, o CMC-BART por covariância tem maior dificuldade em identificar as classes menos frequentes.

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	2	1	0	0	0
4	0	0	23	10	0	0	0
5	0	0	161	131	0	0	0
6	0	0	80	347	9	0	0
7	0	0	5	153	20	0	0
8	0	0	0	30	8	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	59,41	51,63	54,05	-	-

(a) *Vinho branco*

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	4	0	0	0	0
4	0	0	9	6	0	0	0
5	0	0	108	23	0	0	0
6	0	0	52	76	0	0	0
7	0	0	4	37	0	0	0
8	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	61,01	53,14	-	-	-

(b) *Vinho tinto*

Tabela 3.69: *Matriz de confusão para as predições modelo CMC-BART com consenso pela matriz de covariância.*

As matrizes de confusão para as predições do modelo CMC-BART com agregação pela variância (Tabela 3.70) e pela similaridade (Tabela 3.70) são iguais. Isso ocorre devido à similaridade por cosseno tender conforme a amostra cresce. Ao analisar ambas as tabelas, verifica-se que as taxas de acerto do CMC-BART com essas agregações é superior ao CMC-BART com agregação por covariância, com pequena desvantagem para a classe 5. Já o BART tem desempenho superior quando se trata do vinho tinto e inferior quando se trata das classificações do vinho branco.

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	2	1	0	0	0
4	0	0	24	9	0	0	0
5	0	0	172	120	0	0	0
6	0	0	89	329	18	0	0
7	0	0	6	130	42	0	0
8	0	0	0	26	12	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	58,70	53,49	58,33	-	-

(a) *Vinho branco*

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	4	0	0	0	0
4	0	0	8	7	0	0	0
5	0	0	113	18	0	0	0
6	0	0	56	72	0	0	0
7	0	0	4	36	1	0	0
8	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	61,08	53,73	100	-	-

(b) *Vinho tinto*

Tabela 3.70: *Matriz de confusão para o modelo CMC-BART com consenso por similaridade.*

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	2	1	0	0	0
4	0	0	24	9	0	0	0
5	0	0	172	120	0	0	0
6	0	0	89	329	18	0	0
7	0	0	6	130	42	0	0
8	0	0	0	26	12	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	58,70	53,49	58,33	-	-

(a) *Vinho branco*

	Classe Predita						
Classe	3	4	5	6	7	8	9
3	0	0	4	0	0	0	0
4	0	0	8	7	0	0	0
5	0	0	113	18	0	0	0
6	0	0	56	72	0	0	0
7	0	0	4	36	1	0	0
8	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0
Preciso	-	-	61,08	53,73	100	-	-

(b) *Vinho tinto*

Tabela 3.71: *Matriz de confusão para o modelo CMC-BART com consenso pela variância.*

Por fim, nota-se que o CMC-BART, apesar de ter um bom ganho de velocidade e uma apro-

ximação boa, tem grande dificuldade em classificar as classes menos frequentes. Isso se deve ao processo de separação da amostra, que torna o número de indivíduos nas classes menos frequentes ainda menores, tornando ainda mais difícil o aprendizado dos parâmetros. Já a seleção de variáveis de ambos modelos, BART e CMC-BART, foi bem diferente. E a seleção das variavéis de ambos os modelos não chega a ser semelhante nem quando comparada com a seleção de variáveis do SVM de Cortez *et al.* [CCA+09].

3.3 Dados reais de regressão

Nesta seção, são apresentados os resultados para os modelos CMC-BART e BART utilizando dados reais. Os dados escolhidos para a análise são sobre os municípios brasileiros. As variáveis são:

- Índice de Desenvolvimento Humano (idh);
- média da renda domiciliar *per capita* por quintil (rendapc);
- razão entre a renda dos 10% mais ricos e os 40 % mais pobres (razao1040);
- percentual de pessoas em domicílios com água encanada (agua);
- percentual de pessoas atendidas por serviço de coleta de lixo (lixo);
- médicos residentes no município por mil habitantes (medico);
- proporção de enfermeiros com curso superior dentre os enfermeiros (enfermeiro);
- proporção dos professores de ensino fundamental com curso superior (professores);
- percentual da população municipal com mais de 15 anos alfabetizada (alfabetizado);
- esperança de vida ao nascer (esperanca);
- mortalidade até cinco anos de idade por mil nascidos vivos (mortal5);
- mortalidade infantil por mil nascidos vivos (mortal);
- probabilidade de sobrevivência até os 40 anos (prob40);
- probabilidade de sobrevivência até os 60 anos (prob60);
- taxa de fecundidade (taxafecund);
- mulheres entre 10 e 14 anos com filho (maes10a14);
- percentual do número de mulheres chefes de família sem cônjuge e com filhos menores de 15 anos (mulherchefe);
- pessoas com 65 ou mais vivendo sozinhas (LXVoumais).

O Índice de Desenvolvimento Humano (IDH) foi escolhido como variável resposta e as demais variáveis, como explicativas. O objetivo é compreender quais são as variáveis mais impactantes no IDH segundo os modelos CMC-BART e BART.

A Figura 3.14 exibe os gráficos de dispersão com o IDH no eixo y e todas as outras variáveis no eixo x. Para cada gráfico temos a regressão local (Loess) juntamente com seu intervalo de confiança.

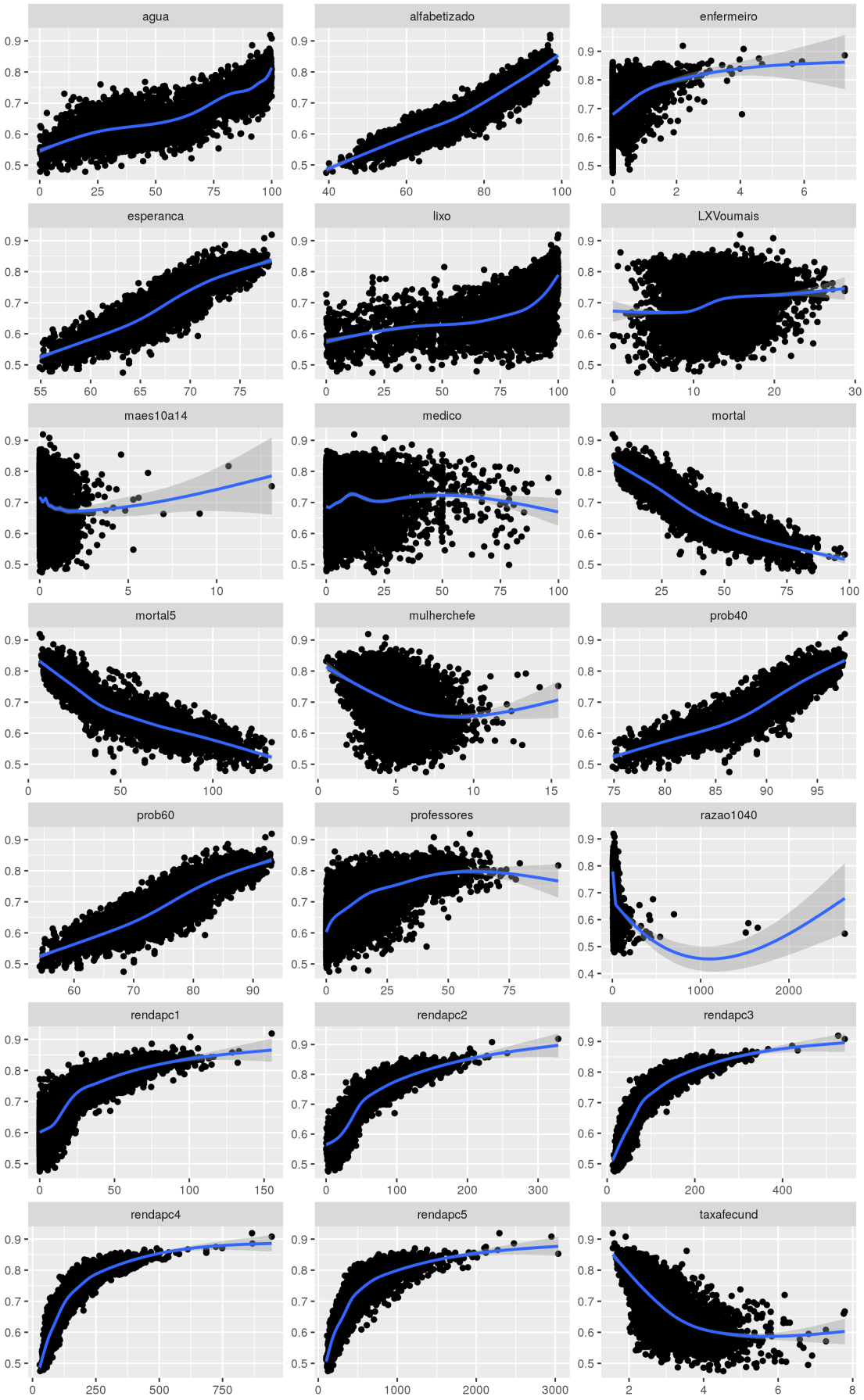


Figura 3.14: *Gráficos de dispersão entre o IDH e as demais variáveis explicativas. A linha azul é a regressão local (loess) e a sombra cinza é o intervalo de confiança da regressão.*

Na Figura 3.15 temos o histograma do IDH dos municípios. Nota-se que existem duas modas: uma primeira em 0,625 e outra em 0,775. O histograma acaba evidenciando a grande diferença que existe no desenvolvimento dos municípios brasileiros. Enquanto as regiões Sul, Centro-Oeste e Sudeste concentram os municípios com melhores níveis de IDH, as regiões Nnorte e Nordeste possuem municípios com menores níveis de IDH.

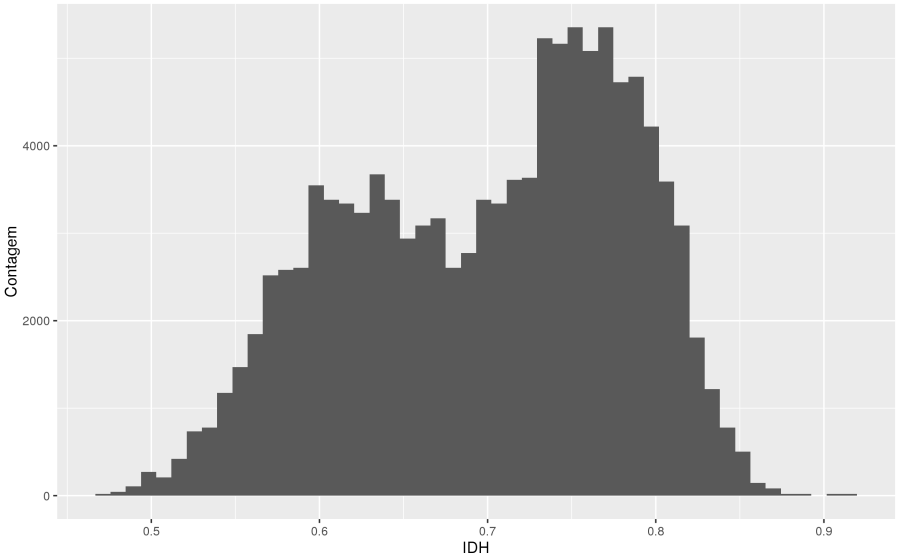
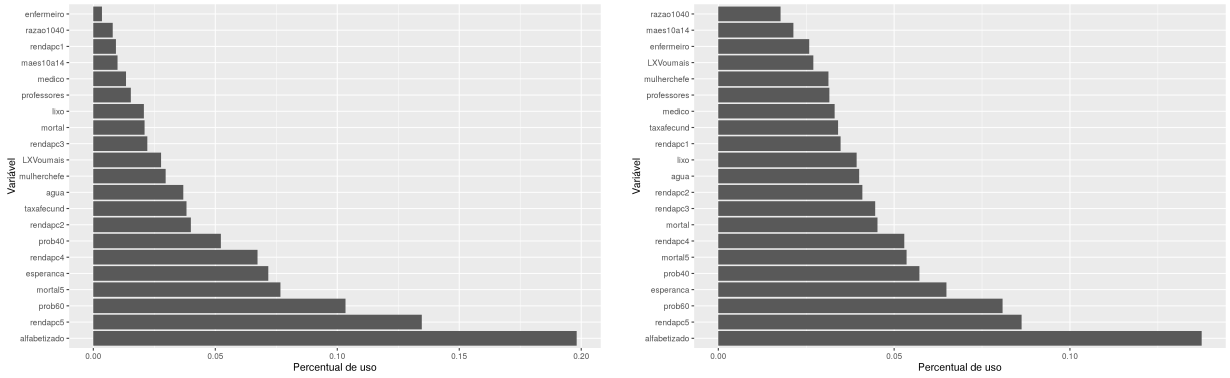


Figura 3.15: *Histograma*

Os hiperparâmetros das prioris utilizadas para o modelo BART foram $\alpha = 0{,}95$; $\beta = 2$; e para o CMC-BART $\alpha = 0{,}90$; $\beta = 1$. Já os hiperparâmetros em comum para ambos modelos foram $m = 100$; $\nu = 3$ e $q = 0{,}90$. Para essa análise o número de *shards* escolhidos foram 20, isto é, $S = 20$.

Analisando as variáveis mais selecionadas pelos modelos BART, Figura 3.16 (a), as variáveis com mais de 5% das escolhas foram: alfabetizado, a média do 5ž quintil da renda per capita, a probabilidade de viver até os 60 anos, a taxa de mortalidade até os 5 anos de idade, a esperança de vida ao nascer, a média do 4ž quintil da renda per capita e a probabilidade de viver até os 40 anos. Já na Figura 3.16 (b) tem-se que as variáveis mais selecionadas pelo modelo CMC-BART, isto é, que aparecem em mais de 5 % das escolhas foram as mesmas do BART, mas em uma ordem diferente. Dessa maneira, pode-se afirmar que para esse conjunto de dados o CMC-BART possui praticamente o mesmo desempenho na seleção de variáveis que o BART para esses hiperparâmetros.



(a) *BART* **(b)** *CMC-BART*

Figura 3.16: *Contagem de variáveis dos modelos.*

Na Figura 3.17 temos os *boxplots* dos erros de predição para cada um dos modelos. Todos os modelos possuem mediana muito próxima de zero e simetria em torno dele. O modelo BART possui

intervalo interquartílico, distância entre o 2ž quartil e o 3ž quartil maior do que os dos outros modelos, porém possui menos erros extremos (apenas quatro). Enquanto isso, dentro os modelos CMC-BART, o intervalo interquartílico é o mesmo para todos os modelos, mas para o CMC-BART por similaridade ou variância a dispersão dos dados é menor. Todos os modelos de CMC-BART aprensentam pontos onde o erro foi extremo.

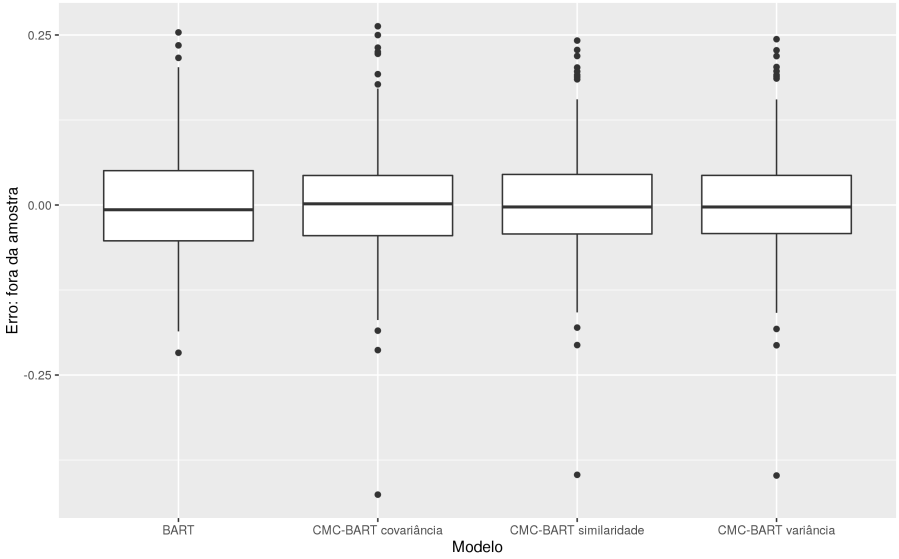


Figura 3.17: *Boxplot do RMSE para cada modelos.*

A Tabela 3.72 mostra o desempenho em termos de RMSE e de velocidade de cada modelo. O BART, apesar de possuir maior intervalo interquartílico e maior dispersão em torno da mediana, praticamente não tem erros extremos na predição, enquanto os modelos CMC-BART já são o contrário: possuem menor intervalo interquartílico que o do BART e menor dispersão em torno da mediana. Contudo, eles têm mais pontos com erros de predição extremos. Dessa maneira, o RMSE do modelo BART acaba por ser um pouco menor do que o dos modelos CMC-BART. Já a velocidade dos modelos CMC-BART é muito maior; o CMC-BART com agregação por variância chega a ser 9,48 vezes mais rápido que o BART, gastando apenas 10,55 % do tempo que o BART gasta para chegar aos mesmos resultados.

Modelo	Tempo (em s)	Tempo relativo	Ganho de vel.	RMSE
BART	110,67	100,00	1,00	0,07
CMC-BART covariância	19,61	17,72	5,64	0,08
CMC-BART similaridade	12,25	11,07	9,03	0,08
CMC-BART variância	11,68	10,55	9,48	0,08

Tabela 3.72: *Tempo de execução e RMSE para cada modelo.*

Por fim, a correlação entre $E(Y|x)$ e todas as predições tendeu a 1,00. Assim, nota-se que os modelos se tornaram muito similares. Esse exemplo com dados reais mostra que os modelos CMC-BART possuem capacidade de seleção de variáveis tão boas quanto o modelo BART com a vantagem de serem muito mais velozes. Além disso, o CMC-BART não tem o poder de predição diminuído com a formação dos consensos, apesar de acabar tendo erros mais extremos que os do BART.

Capítulo 4

Conclusões

O presente trabalho partiu de uma situação em que é necessário trabalhar com mais de uma única máquina para realizar estimações de modelos BART. Para isso, foi utilizada a divisão e a paralelização dos dados em diversos computadores; a esse tipo de estratégia é dado o nome de Consenso Monte Carlo. A combinação de BART com CMC foi chamada de CMC-BART. O principal objetivo aqui era entender e melhorar as escolhas das prioris do CMC-BART para diferentes volumes de dados e propor um novo método de agregação para o CMC. Os objetivos específicos compreendiam:

- verificar como se comporta seleção de variáveis do CMC-BART;
- estudar o impacto da escolha dos hiperparâmetros das prioris do CMC-BART para diferentes volumes de dados na capacidade preditiva do modelo;
- analisar quais métodos de agregação das amostras geram melhores predições.

Nos exemplos com dados reais e sintéticos entendeu-se que a escolha das prioris no CMC-BART é feita de acordo com o objetivo que se deseja, isto é, se o intuito é selecionar um subconjunto das variáveis explicativas e/ou se é realizar predições com baixo erro. Além disso, verificou-se que o CMC-BART, em geral, tem melhor resultado quando as prioris produzem árvores mais profundas, ou seja, quando $\beta = 1$. Isso se deva ao fato de que cada *shard* possui menos dados, e assim, é preciso aprofundar ainda mais as árvores para obtermos melhores predições.

Nos exemplos do capítulo 3 pode-se observar que o volume das amostras exerce influência na escolha das prioris, pois os modelos com melhor desempenho foram aqueles com prioris mais regularizadoras da profundidade das árvores. O mesmo pode ser dito sobre quando se tem um grande número de dimensões.

O novo método de agregação dos resultados, isto é, o CMC-BART por similaridade, não se mostrou muito diferente do CMC-BART por variância nos exemplos apresentados. Isso se deve principalmente ao grande volume de dados que foram usados no trabalho, pois era comum que a correlação entre o ponto em que se realiza a predição $\mathbf{x}_k$ e os diversos pontos de cada *shard* tendessem a 1. Assim sendo, a diferença dos pesos da agregação por similaridade e variância tende a 0. Porém, a nova proposta de agregação dos resultados pode ser mais detalhadamente estudada no futuro com outros tipos de regressão, como por exemplo, regressões discontínuas ou por vizinhança.

Dentre os objetivos específicos percebeu-se que a seleção de variáveis do CMC-BART é suficientemente boa, e que, assim como no BART, árvores menores possuem melhor capacidade de seleção de variáveis.

O impacto dos hiperparâmetros das prioris do CMC-BART para diferentes volumes de dados na capacidade preditiva do modelo é um fator importante a ser analisado ao se escolher os hiperparâmetros. Entende-se que diferentemente do BART, em que Chipman *et al.* [CGM10] propõem uma escolha automática dos hiperparâmetros sem prejuízo no desempenho de predição, isso não é possível no CMC-BART, ao menos nos exemplos estudados. Isso se deve ao fato de que tanto o

número de covariáveis (tamanho dimensional) quanto o número de observações (volume dos dados) é importante na escolha dos hiperparâmetros.

Finalmente, os métodos de agregação das amostras que geram melhores predições para o CMC-BART são o por variância e similaridade. Contudo, a agregação por variância mostra-se um pouco mais veloz. O CMC-BART com agregação por covariância mostra-se mais lento e também como tendo pior capacidade de predição, o que é um resultado esperado.

4.1 Considerações Finais

Este trabalho mostra como o CMC-BART pode ser utilizado para trabalhar com o modelo BART com volume de dados realmente grande (isto é, casos em que não é possível alocação em mémoria RAM) sem demandar um tempo de execução não factível.

Os objetivos principais e específicos foram plenamente atingidos e a contribuição principal deste trabalho encontra-se em facilitar a escolha dos hiperparâmetros das prioris para usuários futuros do modelo CMC-BART.

4.2 Sugestões para Pesquisas Futuras

Para pesquisas futuras podem ser exploradas três abordagens: testar outros métodos de amostragem, testar métodos de CMC que surgiram recentemente ou criar novas estratégias de Consenso Monte Carlo.

Sugere-se também que sejam explorados outros métodos de amostragem no *sharding*. Talvez outros esquemas de amostragem, tais como amostragem estratificada ou por agrupamentos, possam produzir melhores *shards*. Outra sugestão é testar novas estratégias de consenso, como: Sequenciais de Consenso Monte Carlo(SCMC) [AD01]; Kernel Consenso Monte Carlo (KCMC) [NWX13]; Mistura Consenso Monte Carlo (MxCMC), [MPB03] ou [TSV05]. Essas novas estratégias de Consenso Monte Carlo talvez produzam melhores aproximações da verdadeira posteriori; porém, é importante levar em consideração a velocidade na hora de se compararm o desempenho dessas estratégias. Por fim, a criação de outros métodos de Consenso Monte Carlo também pode ser explorada.

Apêndice A

Código

Segue abaixo o código das principais funções utilizadas no trabalho, principalmente a *consensus_monte_carlo* e a *sharding*.

```
#### Functions ####
setwd("C:/Users/Nina/Desktop/Lucas/Dissertacao/functions")
library(Rcpp)
library(BART)
library(data.table)
Sys.setenv("PKG_CXXFLAGS"="-std=c++11")
sourceCpp("./c_functions.cpp")


element_wise_mult <- function(M, v)
{
  return(t(t(M) * v))
}

check_packages <- function(package)
{
  new.pkg <- package[!(package %in% installed.packages()[, "Package"])]

  if (length(new.pkg))
    install.packages(new.pkg, dependencies = TRUE)
  sapply(package, require, character.only = TRUE)
}

dist2 <- function(x, y) # reescrever no Cpp
{
  x <- scale(x)
  y <- scale(y)
  distance <- matrix(rep(as.numeric(0.0), nrow(x)*nrow(y)), ncol = nrow(x), byrow
      = F)
  for(i in 1:nrow(y))
    for(j in 1:nrow(x))
      distance[i,j] <- as.numeric(sqrt(sum((y[i,] - x[j,])^2)))
  return(distance)
}

dist3 <- function(x, y) # reescrever no Cpp
{
  x <- scale(x)
  y <- scale(y)
  distance <- matrix(rep(as.numeric(0), nrow(x)*nrow(y)), ncol = nrow(x), byrow =
      F)
  for(i in 1:nrow(y))
    distance[i, ] <- apply(x, 1, function(z) sqrt(sum((z - y[i,])^2)))
  return(distance)
}

```

```
45 k_nearest_neibourgh ← function(k, x.train, x.teste)
46 {
47   min_dist ← dist2(x.train, x.test)
48   return(t(apply(min_dist, 1, function(x) sort(x, index.return = T)$ix[1:k])))
49 }
50
51 k_nearest_neibourgh_dist ← function(k, x.train, x.teste)
52 {
53   min_dist ← dist2(x.train, x.test)
54   return(t(apply(min_dist, 1, function(x) sort(x, index.return = T)$x[1:k])))
55 }
56
57 consensus_monte_carlo ← function(S_size, agg_method, samples_path, neiboughs_
       path, varcount_path, lambda = 1e-06)
58 {
59   if(!agg_method %in% c("variance", "covariance", "similarity"))
60     stop("The agg_method must be \"variance\", \"covariance\", or \"similarity\"
         ")
61   list_samples ← list(NULL)
62   list_similarity ← list(NULL)
63   list_varcount ← list(NULL)
64
65   for(i in 1:S_size)
66   {
67     list_samples[[i]] ← as.matrix(readRDS(paste0(samples_path, i, ".rds")))
68     list_similarity[[i]] ← as.matrix(readRDS(paste0(neiboughs_path, i, ".rds")))
69     list_varcount[[i]] ← as.matrix(readRDS(paste0(varcount_path, i, ".rds")))
70   }
71
72   if(agg_method == "covariance")
73   {
74     list_cov ← list()
75     list_cov ← lapply(list_samples, function(x) solve(cov(x) + lambda * diag(
           ncol(x))))
76
77     cov ← Reduce("+", list_cov)
78     weighted_cov ← Map("%*%", list_cov, lapply(list_samples, t))
79     weighted_cov ← Reduce("+", weighted_cov)
80     lambda_matrix ← lambda * diag(ncol(list_cov[[1]]))
81
82     consensus_cov ← t(solve(cov + lambda_matrix) %*% weighted_cov)
83     num ← ncol(consensus_cov)
84     colnames(consensus_cov) ← paste0(rep("f", num), 1:num)
85
86     z ← lapply(list_varcount, function(x) colSums(x))
87     z ← lapply(z, function(x) x/sum(x))
88     z ← Reduce("+", z)/S_size
89
90     cmc ← list()
91     cmc$sample ← consensus_cov
92     cmc$z ← z
93     return(cmc)
94   }
95
96   if(agg_method == "variance")
97   {
98     list_var ← list()
99     list_var ← lapply(list_samples, function(x) apply(x, 2, var))
100    list_var ← lapply(list_var, function(x) (x + 1e-03)^(-1))
101
102    weighted_sample ← Map(element_wise_mult, list_samples, list_var)
103    weighted_sample ← Reduce("+", weighted_sample)
104
105    sum_list_var ← Reduce("+", list_var)
106    sum_list_var ← sum_list_var^(-1)
```

```
107
108     consensus_var ← element_wise_mult(weighted_sample, sum_list_var)
109     num ← ncol(consensus_var)
110     colnames(consensus_var) ← paste0(rep("f", num), 1:num)
111
112     z ← lapply(list_varcount, function(x) colSums(x))
113     z ← lapply(z, function(x) x/sum(x))
114     z ← Reduce("+", z)/S_size
115
116     cmc ← list()
117     cmc$sample ← consensus_var
118     cmc$z ← z
119     return(cmc)
120   }
121
122   if(agg_method == "similarity")
123   {
124     list_var ← list()
125     list_var ← lapply(list_samples, function(x) apply(x, 2, var))
126     list_var ← lapply(list_var, function(x) (x + 1e-03)^(-1))
127
128     list_neiboughs_weight ← lapply(list_similarity, function(x) apply(x, 1, max)
          )
129     list_weights ← Map("*", list_var, list_neiboughs_weight)
130
131
132     #list_var ← list_weights
133     weighted_sample ← Map(element_wise_mult, list_samples, list_weights)
134     weighted_sample ← Reduce("+", weighted_sample)
135
136     sum_list_var ← Reduce("+", list_weights)
137     sum_list_var ← sum_list_var^(-1)
138
139     consensus_weighted_var ← element_wise_mult(weighted_sample, sum_list_var)
140     num ← ncol(consensus_weighted_var)
141     colnames(consensus_weighted_var) ← paste0(rep("f", num), 1:num)
142
143     z ← lapply(list_varcount, function(x) colSums(x))
144     z ← lapply(z, function(x) x/sum(x))
145     z ← Reduce("+", z)/S_size
146
147     cmc ← list()
148     cmc$sample ← consensus_weighted_var
149     cmc$z ← z
150     return(cmc)
151   }
152 }
153
154 sharding ← function(x, num_shards = 2, directory = getwd(), name = NULL, gen_not
      _files = NULL, gen_train = F)
155 {
156   require(data.table)
157   if(is.null(name))
158     name ← deparse(substitute(x))
159   x ← x[sample(nrow(x)), ]
160   folds ← cut(seq(1,nrow(x)), breaks = (num_shards + as.numeric(gen_train)),
        labels = FALSE) - as.numeric(gen_train)
161
162   for(i in as.numeric(!gen_train):num_shards)
163   {
164     saveRDS(x[which(folds == i), ]
165             , file = paste0(directory, "/", name, "shard_", i, ".rds"))
166     if(!is.null(gen_not_files))
167       if(i %in% gen_not_files)
168         saveRDS(x[which(folds != i), ]
```

```
169               , file = paste0(directory, "/", name, "shard_not_", i, ".rds"))
170   }
171 }
172
173 friedman_function <- function(x, FUN, ...)
174 {
175   if("data.frame" %in% is(x))
176   {
177     y <- numeric(nrow(x))
178     sigma <- numeric(nrow(x))
179     for(i in 1:nrow(x))
180     {
181       sigma[i] <- FUN(..., n = 1)
182       y[i] <- 10*sin(pi*x[i,1]*x[i,2]) + 20*(x[i,3]-0.5)^2 + 10*x[i,4] + 5*x[i,5]
183     }
184     return(data.frame(y = y + sigma))
185   }
186   else
187   {
188     print(x[1])
189     return(10*sin(pi*x[1]*x[2]) + 20*(x[3]-0.5)^2 + 10*x[4] + 5*x[5] + sigma)
190   }
191 }
192
193 friedman_function2 <- function(x, FUN, ...)
194 {
195   if("data.frame" %in% is(x))
196   {
197     y1 <- numeric(nrow(x))
198     y2 <- numeric(nrow(x))
199     sigma <- numeric(nrow(x))
200     for(i in 1:nrow(x))
201     {
202       sigma[i] <- FUN(..., n = 1)
203       y1[i] <- 10*sin(pi*x[i,1]*x[i,2]) + 20*(x[i,3]-0.5)^2 + 10*x[i,4] + 5*x[i
              ,5]
204     }
205     return(data.frame(y = y1 + sigma))
206   }
207   else
208   {
209     print(x[1])
210     return(10*sin(pi*x[1]*x[2]) + 20*(x[3]-0.5)^2 + 10*x[4] + 5*x[5] + sigma)
211   }
212 }
213
214 artificial_set_generator <- function(observations, components = 1, seed, FUN,
        ...)
215 {
216
217   if(!missing(seed))
218     set.seed(seed)
219
220   num_total <-  observations*components
221   x <- FUN(n = num_total, ...)
222   x <- as.data.frame(base::split(x, 1:components))
223   return(x)
224 }
225
226 wu_function <- function(n_pseudo_variable = 0)
227 {
228   require(dplyr)
229   x1 <- c(runif(200, 0.1, 0.4), runif(100, 0.6, 0.9))
230   x2 <- c(runif(100, 0.1, 0.4), runif(100, 0.6, 0.9), runif(100, 0.1, 0.9))
231   x3 <- c(runif(200, 0.6, 0.9), runif(100, 0.1, 0.4))
```

```
232
233   if(n_pseudo_variable > 0)
234   {
235     x_pseudo <- runif(300 * n_pseudo_variable, 0.1, 0.9)
236     x_pseudo <- matrix(x_pseudo, nrow = 300)
237     colnames(x_pseudo) <- paste0(rep("x", n_pseudo_variable), as.character(4:(3+n
          _pseudo_variable)))
238   }
239
240   {
241     if(n_pseudo_variable == 0)
242     {
243       x <- cbind(x1, x2, x3)
244     } else {
245       x <- cbind(x1, x2, x3, x_pseudo)
246     }
247   }
248
249   x <- as.data.frame(x)
250
251   x <- x %>%
252     dplyr::mutate(y = 0) %>%
253     dplyr::mutate(y = ifelse((x1 == 0.5 | x1 < 0.5) & (x2 == 0.5 | x2 < 0.5), 1
          + rnorm(n(), 0, 0.25), y)) %>%
254     dplyr::mutate(y = ifelse((x1 == 0.5 | x1 < 0.5) & (x2 > 0.5), 3 + rnorm(n(),
           0, 0.25), y)) %>%
255     dplyr::mutate(y = ifelse(x1 > 0.5, 5 + rnorm(n(), 0, 0.25), y))
256
257   return(x)
258 }
259
260 filter_model_df <- function(data, filter_model_type, filter_model)
261 {
262   model_file <- "non_cmc"
263   if(filter_model == "CMC-BART")
264     model_file <- "cmc"
265
266   data %>%
267     filter(model_type == filter_model_type, model == filter_model) %>%
268     ungroup() %>%
269     arrange(-as.numeric(X2)) %>%
270     select(model, m, alpha, beta, k, n, p) %>%
271     plyr::rename(replace = c("model" = "Modelo"
272                              , "m" = "$m$"
273                              , "alpha" = "$\\alpha$"
274                              , "beta" = "$\\beta$"
275                              , "k" = "$k$")) %>%
276     xtable(type = "latex", display = c(rep("s", 8), rep("fg", 0)), digits = 2)
          %>%
277     print(math.style.exponents = TRUE
278           ,  include.rownames=FALSE
279           ,  sanitize.colnames.function = identity
280           , file = paste0("tables/z_hyperparam_best_model_", model_file, "_p", p
                , "_n", n_obs, ".tex")
281           , Encoding = "UTF-8")
282 }
```

Referências Bibliográficas

[AC93] James H. Albert e Siddhartha Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993. 11

[AD01] Nando de Freitas & Neil Gordon Arnaud Doucet, editor. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, New York, U.S.A., 2001. 72

[ARC14] Reshu Agarwal, Pritam Ranjan e Hugh Chipman. A new bayesian ensemble of trees approach for land cover classification of satellite imagery. *Canadian Journal of Remote Sensing*, 39(6):507–520, 2014. 4

[BFOS84] Leo Breiman, Jerome H. Friedman, Richard A. Olshen e Charles J. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984. 1, 9

[BK14] Justin Bleich e Adam Kapelner. Bayesian additive regression trees with parametric models of heteroskedasticity, 2014. 4

[Bre96] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. 2, 3

[Bre01] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. 2, 3

[CCA+09] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos e José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009. 59, 60, 63, 67

[CGM98] Hugh A. Chipman, Edward I. George e Robert E. McCulloch. Bayesian CART model search. *Journal of the American Statistical Association*, 93(443):935–948, 1998. 9

[CGM10] Hugh A. Chipman, Edward I. George e Robert E. McCulloch. BART: Bayesian additive regression tree. *The Annals of Applied Statistics*, 4(1):266–298, 2010. 2, 3, 5, 6, 7, 8, 9, 11, 19, 20, 26, 71

[con17] Wikipedia contributors. Gödel Prize — Wikipedia, The Free Encyclopedia, 2017. [Online; https://en.wikipedia.org/w/index.php?title=Gacessado em 29 de janeiro de 2018]. 3

[DB17] Gitesh Dawer e Adrian Barbu. Relevant ensemble of trees, 2017. 2

[Die00] Thomas G. Dietterich. Ensemble methods in machine learning. Em Josef Kittler e Fabio Roli, editors, *Proceedings of the First International Workshop on Multiple Classifier Systems*, páginas 1–15, London, Inglaterra, 21-23 Junho 2000. Springer-Verlag. 2

[Fri91] Jerome H. Friedman. The multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991. 19

[Fri01] Jerome H. Friedman. Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. 2

[Fri02] Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002. 2

[FS96] Yoav Freund e Robert E. Schapire. Experiments with a new boosting algorithm. Em Lorenza Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning*, páginas 148–156, Bari, Itália, 3-6 Julho 1996. Morgan Kaufmann Publishers. 2, 3

[GEW06] Pierre Geurts, Damien Ernst e Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006. 2

[Gre95] Peter A. Greene. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995. 9

[HT90] Hastie e Robert Tibshirani. *Generalized Additive Models*. Chapman and Hall, New York, NY, USA, 1990. 4

[HT00] Trevor Hastie e Robert Tibshirani. Bayesian backfitting. *Statistical Science*, 15(3):196–223, 2000. 8

[HTF01] Hastie, Robert Tibshirani e Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001. 1, 3

[KWHP16] Bereket P. Kindo, Hao Wang, Timothy Hanson e Edsel A. Peña. Bayesian quantile additive regression trees, 2016. 4

[KWP16] Bereket P. Kindo, Hao Wang e Edsel A. Peña. Multinomial probit bayesian additive regression trees. *Stat (International Statistical Institute)*, 5(1):119–131, 2016. 4

[LMLS16] Purushottam Laud, Robert McCulloch, Brent Logan e Rodney Sparapani. Nonparametric survival analysis using bayesian additive regression trees (BART). *Statistics in Medicine*, 35(16):2741–2753, 2016. 4

[LRT15] Balaji Lakshminarayanan, Daniel M. Roy e Yee Whye Teh. Particle gibbs for bayesian additive regression trees. Em Guy Lebanon e S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, páginas 553–561, San Diego, Califórnia, EUA, 09-12 May 2015. PMLR. 1, 2

[MPB03] G. J. McLachlan, D. Peel e R. W. Bean. Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics & Data Analysis*, 41(3-4):379–388, 2003. 72

[Mur17] Jared S. Murray. Log-linear bayesian additive regression trees for categorical and count responses, 2017. 4

[NWX13] Willie Neiswanger, Chong Wang e Eric Xing. Asymptotically exact, embarrassingly parallel mcmc, 2013. 72

[PCGM17] Matthew Pratola, Hugh Chipman, Edward George e Robert McCulloch. Heteroscedastic BART using multiplicative regression trees, 2017. 4

[SBB+16] Steven L. Scott, Alexander W. Blocker, Fernando V. Bonassi, Hugh A. Chipman, Edward I. George e Robert E. McCulloch. Bayes and big data: The consensus monte carlo algorithm. *International Journal of Management Science and Engineering Management*, 11:78–88, 2016. 4, 14, 16, 19, 26

[SCR07] Daria Sorokina, Rich Caruana e Mirek Riedewald. Additive groves of regression trees. Em Joost N. Kok, Jacek Koronacki, Raomon Lopez de Mantaras, Stan Matwin, Dunja Mladenič e Andrzej Skowron, editors, *Machine Learning: ECML 2007*, páginas 323–334, Berlin, Heidelberg, Alemanha, 09-12 May 2007. Springer. 2

[TSV05] Mahlet G. Tadesse, Naijun Sha e Marina Vannucci. Bayesian variable selection in clustering high-dimensional data. *Journal of the American Statistical Association*, 100(470):602–617, 2005. 72

[TW87] Martin Tanner e Wing Hung Wong. The calculation of posterior distributions by data augmentation (with discussion). *Journal of the American Statistical Association*, 82(398):528–550, 1987. 4

[VGB15] Rashmi Korlakai Vinayak e Ran Gilad-Bachrach. DART: Dropouts meet multiple additive regression trees. Em Guy Lebanon e S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, páginas 489–497, San Diego, Califórnia, EUA, 09-12 May 2015. PMLR. 2

[ZH10] Junni L. Zhang e Wolfgang K. Härdle. The bayesian additive classification tree applied to credit risk modelling. *Computational Statistics & Data Analysis*, 54(5):1197–1205, 2010. 4