UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JAIRSON BARBOSA RODRIGUES

Análise de Fatores Relevantes no Desempenho de Plataformas para Processamento de *Big Data*: Uma Abordagem Baseada em Projeto de Experimentos

Recife

2020

JAIRSON BARBOSA RODRIGUES

Análise de Fatores Relevantes no Desempenho de Plataformas para Processamento de *Big Data*: Uma Abordagem Baseada em Projeto de Experimentos

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

Área de Concentração: Redes de Computadores e Sistemas Distribuídos

Orientador: Prof. Dr. Germano Crispim Vasconcelos

Coorientador: Prof. Dr. Paulo Romero Martins Maciel

Recife

2020

Catalogação na fonte
Bibliotecária Mariana de Souza Alves CRB4-2105

R696a Rodrigues, Jairson Barbosa
Análise de fatores relevantes no desempenho de plataformas para processamento de big data: uma abordagem baseada em projeto de experimentos / Jairson Barbosa Rodrigues. – 2020.
202 f., il., fig., tab.

Orientador: Germano Crispim Vasconcelos.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2020.
Inclui referências, apêndices e anexos.

1. Redes de Computadores e Sistemas Distribuídos. 2. Projeto de Experimentos (DoE). 3. Aprendizagem de Máquina. 4. Computação Distribuída. I. Vasconcelos, Germano Crispim. (orientador) II. Título.

004.6 CDD (22. ed.) UFPE-CCEN 2021-10

Jairson Barbosa Rodrigues

“Análise de Fatores Relevantes no Desempenho de Plataformas para Processamento de Big Data — Uma Abordagem Baseada em Projeto de Experimentos”

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

Aprovado em: 30/10/2020.

_____

Orientador: Prof. Dr. Germano Crispim Vasconcelos

BANCA EXAMINADORA

_____

Prof. Dr. Carlos André Guimarães Ferraz
Centro de Informática / UFPE

_____

Prof. Dr. Kiev Santos da Gama
Centro de Informática / UFPE

_____

Prof. Dr. Renato Tinós
Departamento de Computação e Matemática / USP

_____

Prof. Dr. Alexandre Magno Andrade Maciel
Escola Politécnica de Pernambuco / UPE

_____

Prof. Dr. Paulo Renato Alves Firmino
Centro de Ciências e Tecnologia / UFCA

À Maria Clara.

AGRADECIMENTOS

Hors-concours, aos meus pais, José Rodrigues da Silva e Severina Rosa Barbosa Rodrigues, pelo dom da vida, por cada gesto de cuidado, amor e zelo, por todo o empenho com a minha educação. Obrigado por pavimentarem a minha estrada de vida, me fornecendo os degraus para ir além. Minha vontade de aprender é reflexo de seus exemplos de esforço e tem alicerces no seu suporte constante. Destaco em particular a seriedade pedagógica de minha mãe — a primeira professora na educação formal. Certamente por isso, certamente por eles, a sala de aula, as letras, o aprender me agradam tanto.

Ao meu orientador, professor Germano Crispim Vasconcelos, responsável maior por esta pesquisa, por dedicar seu tempo e atenção ao longo processo de orientação. Por me ensinar a ser mais semântico, por tão generosamente emprestar seu tempo lendo esta Tese e tantos outros manuscritos. Por me induzir a pensar, por me fazer seguir e não apenas me conduzir. Pelo suporte em tudo aquilo que preciso melhorar, por sua nobreza e generosidade e por me guiar com tanta sabedoria, não há como agradecer.

Ao professor Paulo Romero Martins Maciel, do Centro de Informática da Universidade Federal de Pernambuco, co-orientador desta Tese, agradeço por sua contribuição inestimável ao me apresentar à beleza das técnicas de projeto de experimentos. Pela precisão e objetividade, por compartilhar de sua enorme experiência, minha imensa gratidão.

Meus sinceros agradecimentos ao professor Renato Tinós, da Universidade de São Paulo, pela gentil acolhida no Instituto de Computação e Matemática, pela atenção, pela disponibilidade e por todo o conhecimento compartilhado, eternamente grato.

Ao professor Joaquim Pereira Neto, da Universidade do Estado da Bahia, por sua grandeza em compartilhar com tanta paciência um lampejo de seus conhecimentos de Estatística. Por sua disponibilidade e suporte, sou profundamente agradecido.

À Maria Clara, minha filha, que nasceu junto com o projeto de doutorado e cresceu sua primeira infância perguntando "porque o trabalho do papai nunca acaba", meu eterno amor por tornar tão clara a minha razão de tudo.

"*Experimental observations are only experience carefully planned in advance, and designed to form a secure basis of new knowledge*." (FISHER, 1935, p. 8).

RESUMO

Uma série de fenômenos tecnológicos, sociais e de mercado originaram um paradigma comumente referenciado pelo termo *Big Data*: grandes conjuntos de dados, criados a taxas muito altas, em formatos diversos e adquiridos de variadas fontes. Tais circunstâncias demandam tecnologias escaláveis, redundantes e tolerantes a falhas; normalmente obtidas por modelos de computação nas nuvens. O desempenho das tarefas em termos de tempo e custo depende de fatores como *hardware*, volume de dados e tipo de algoritmo. Escolher a configuração mais adequada é um problema de notória dificuldade devido ao número de soluções possíveis e inviabilidade de investigação de todos os cenários. Esta Tese se concentra na análise experimental de algoritmos de aprendizagem de máquina em plataformas de processamento para *Big Data* e se justifica ao auxiliar o adequado aprovisionamento de recursos em nuvem. É proposta uma metodologia baseada nas técnicas 2^k fatorial, fundamentadas na abordagem de Projeto de Experimentos (*DoE — Design of Experiments*) para avaliação da influência de fatores sobre tempo e custo, a ordenação daqueles mais relevantes e a derivação de modelos preditores. O desempenho de 288 *clusters* reais foi avaliado através de seis projetos com 48 unidades experimentais, cada uma composta por arranjos de 8 e 28 máquinas, e cada máquina variando entre 12 e 32 núcleos, 1, 7 e 8 discos, 3x e 6x RAM por núcleo, totalizando um poder computacional de até 896 núcleos e 5.25 TB de RAM. Os experimentos foram conduzidos sobre duas bases de dados. Primeiramente foram construídos 1.06 TB de dados sintéticos estruturados em 3.65 milhões de instâncias e 40 mil dimensões para classificação através de Florestas Aleatórias. Posteriormente, foi construído um *Corpus* com 249 GB de dados não estruturados de 16 milhões de páginas web de sete países de língua portuguesa. A tarefa computacional consistiu na classificação de texto para distinguir o português brasileiro de outras variações. Foram examinados os algoritmos Regressão Logística, Florestas Aleatórias, Máquinas de Vetores de Suporte, Naïve Bayes e *Perceptron* de Múltiplas Camadas. Análises de regressão foram aplicadas para quantificar a influência dos fatores. Os resultados incluem modelos lineares para estimar tempo e custo e uma ferramenta de análise visual baseada em coordenadas paralelas. Por fim, o trabalho evidencia a relevância dos métodos de *DoE* como abordagem para estimar desempenho em ambientes de processamento para *Big Data*.

Palavras-chaves: Projeto de Experimentos (DoE). Aprendizagem de Máquina. Computação Distribuída. Big Data. Modelos Preditores. Inferência Estatística.

ABSTRACT

A series of technological, social, and market phenomena originated a paradigm commonly referred to as Big Data: large data sets, created at very high rates, in different formats, and acquired from different sources. Such circumstances demand scalable, redundant, and fault-tolerant technologies, often achieved on clusters in the cloud computing model. The performance of tasks concerning time and cost depends on factors such as hardware, data volume and the type of algorithm. Choosing the most appropriate configuration regarding computational power or software parameters is a problem of notorious difficulty due to the number of possible solutions and the unfeasibility of investigating all scenarios. This thesis focuses on the experimental analysis of machine learning algorithms on processing platforms for Big Data, being justified by assisting the adequate provisioning of cloud resources. It is proposed a methodology based on 2^k factorial techniques, reasoned on the Design of Experiments (DoE), to evaluate the influence of factors on time and cost, ranking the most relevant ones, and derivate predictive models. The performance of 288 real clusters was evaluated through six designs with 48 experimental units, each composed of arrangements of 8 and 28 machines, and each machine varying between 12 and 32 cores, 1, 7, and 8 disks, 3x and 6x RAM per core, totaling a computational power of up to 896 cores and 5.25 TB of RAM. The experiments were conducted on two databases. First, 1.06 TB of synthetic data was built, structured in 3.65 million instances, and 40 thousand dimensions for classification through the Random Forest algorithm. Subsequently, a Corpus was built with 249 GB of unstructured data from 16 million web pages from seven Portuguese-speaking countries. The computational task consisted of classifying text to distinguish Brazilian Portuguese from other variations. Five different machine learning algorithms were then examined: Logistic Regression, Random Forest, Support Vector Machines, Naïve Bayes, and Multilayer Perceptron. Regression analyzes were applied to quantify the influence of the factors. The results include linear models to estimate time and cost, quantify the effects of factors on the response, and a visual analysis tool based on parallel coordinates. Finally, the work provides consistent evidence of the relevance of DoE methods as an approach to estimate performance in Big Data processing environments.

Keywords: Design of Experiments (DoE). Machine Learning. Distributed Computing. Big Data. Prediction Models. Statistical Inference.

LISTA DE FIGURAS

LISTA DE CÓDIGOS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizagem de Máquina
CERN	*Conseil Européen pour la Recherche Nucléaire* Organização Européia para a Pesquisa Nuclear
CSS	*Cascading Style Sheet*
CSV	*Comma Separated Values*
DoE	*Design of Experiments* Projeto de Experimentos
E/S	Entrada e Saída
ECC	Elastic Cloud Computing
FSPN	*Fluid Stochastic Petri Networks* Redes de Petri Estocásticas Fluidas
GFS	*Google File System* Sistema de Arquivo Google
GPU	*Graphics Processing Unit* Unidade de Processamento Gráfico
HDFS	*Hadoop Distributed File System* Sistema de Arquivo Distribuído Hadoop
IaaS	*Infrastructure as a Service* Infraestrutura como Serviço
LHC	*Large Hadron Collider* Grande Colisor de Hádrons
LHS	*Latin Hypercube Sampling* Amostragem em Hipercubo Latino
LR	*Logistic Regression* Regressão Logística
LSST	*Large Synoptic Survey Telescope* Grande Telescópio de Levantamento Sinóptico

MLP	*Multilayer Perceptron* Perceptron de Múltplicas Camadas
NB	Naïve Bayes
NIID	Normais, Independentes e Identicamente Distribuídos
NIST	*National Institute of Standards and Technology* Instituto Nacional de Padrões e Tecnologia
PT7 WEB	*Corpus* WEB rotulado da Língua Portuguesa
QN	*Queueing Nets* Redes de Filas
QoS	*Quality of Service* Qualidade do Serviço
RAM	*Random-Access Memory* Memória de Acesso Aleatório
RDDs	*Resilient Distributed Datasets* Conjuntos de Dados Distribuídos Resilientes
RF	*Random Forests* Florestas Aleatórias
ROC	*Receiver Operating Characteristic*
SSD	*Solid State Drive* Disco de Estado Sólido
SVM	*Support Vector Machines* Máquina de Vetores de Suporte
SWN	*Stochastic Well-formed Petri Networks* Redes de Petri Estocásticas bem Formadas
TLD	*Top Level Domain* Domínio de Nível Superior
URL	*Universal Resource Locator*
XaaS	*Everything as a Service* Tudo como Serviço

SUMÁRIO

1 INTRODUÇÃO

> *Information is the oil of the 21st century, and analytics is the combustion engine.*
>
> — Peter Sondergaard, *Senior Vice President*, Gartner

Os problemas de análise de dados em larga escala não surgiram de repente, mas sim ao longo de vários anos, já que criar dados é muito mais fácil que encontrar utilidade para eles (TSAI et al., 2015). Embora esta problemática não seja nova, os anos 2000 foram marcados por um salto não apenas nas possibilidades técnicas de processamento, armazenamento e transmissão de dados, mas também pela explosão de fenômenos de geração de dados em volumes sem precedentes na história da Humanidade. Este contexto não indica um fenômeno necessariamente recente, mas um conjunto de questões, inéditas e clássicas que, combinadas com o cenário tecnológico, social e econômico, deram origem a um novo paradigma (LUVIZAN; MEIRELLES; DINIZ, 2014). Comumente referenciado pelo termo *BIG DATA* [1], este conceito se refere a tecnologias analíticas que têm existido por anos mas que agora podem ser aplicadas rapidamente, em grande escala, tornando-se acessíveis a mais usuários (MILLER, 2013).

1.1 CONTEXTO E MOTIVAÇÃO

A Humanidade ultrapassou a marca de *Zettabytes* (ZB) [2] a uma taxa exponencial, tornando realidade as questões de processamento de grandes volumes de dados para uma ampla gama de aplicações empresariais e acadêmicas (POSPELOVA, 2015). Neste cenário, a competitividade de uma nação foi associada por Park e Leydesdorff (2013) com a forma como esta lida com a escalabilidade de dados em uma sociedade digital sempre em crescimento, destacando a prioridade dada a pesquisas científicas conduzidas pela indústria e academia. Alguns autores apontam ainda que as primeiras duas décadas do Século XXI têm testemunhado o surgimento de revistas acadêmicas direcionadas a aspectos gerais de *Big Data*, tais como *Data Science Journal*, *Journal of Data Science*, *EPJ Data Science* e *Journal of Big Data*; enquanto outros periódicos têm foco específico, a exemplo da revista *GigaScience*, cuja atenção é voltada ao estudo do tema aplicado ao espectro das ciências biomédicas e da vida.

[1] A origem do termo e reflexões acerca da classificação em escala *Big Data* encontram-se na Seção 2.2.1.
[2] Zettabyte (ZB) $= 1.024 \times 10^{21}$ *bytes*.

Gantz e Reinsel (2012) já apontavam que a fatia do universo digital atribuída aos mercados emergentes girava em torno de 36% e prospectavam o patamar de 62% para 2020, imputando à China 21% do fluxo de bits de todo o planeta. Em 2018 a sociedade humana criou 33 ZB de dados e estima-se que este volume aumentará em até cinco vezes até 2025 (EYUPOGLU, 2020).

Isto posto, os avanços de inovação científica dependerão da capacidade de gerar conhecimento e significado a partir de enormes quantidades de informação, a exemplo dos dados coletados por projetos como o Grande Colisor de Hádrons (LHC/CERN) na Europa, o Grande Telescópio de Levantamento Sinóptico (LSST) no Chile, a busca em grafos de bilhões de arestas em redes sociais e a antecipação de tendências nas relações de consumo no mercado (SIMONET; FEDAK; RIPEANU, 2015). A pesquisa intensiva em grandes volumes de dados é apontada por Hey et al. (2009) como o quarto paradigma da Ciência, diferenciando-se da (i) ciência experimental que buscava descrever os fenômenos da natureza a partir da observação; (ii) da ciência teórica, como as Leis de Newton ou as Equações de Maxwell; e (iii) da investigação de fenômenos complexos através da simulação computacional.

Destoante da euforia com o potencial da análise de *Big Data*, Boyd e Crawford (2012) levantaram outras reflexões como: (i) *Big Data* mudou a definição do conhecimento e o que pensamos sobre pesquisa científica, (ii) as alegações sobre precisão e objetividade estão enganadas, (iii) mais dados nem sempre resultam em melhores dados, (iv) fora de contexto, *Big Data* perde o significado, (v) o acesso aos dados levanta questões éticas e (vi) o alcance limitado a *Big Data* amplia nossas desigualdades digitais.

Não obstante às implicações éticas, morais e às barreiras teóricas e práticas, é alta a expectativa com a geração de conhecimento a partir do atual volume de dados. E, uma vez estabelecida a sua importância, faz-se necessário o debate acerca das técnicas para lidar com os dados gerados na era contemporânea.

1.2 PROBLEMA

> "De muitas maneiras, a análise de *Big Data* hoje é um retorno à era da computação em *mainframe* (...) mais parece com as tarefas em lote da década de 1960, as quais eram enviadas e se saía para tomar um café, com poucas informações sobre o que realmente está acontecendo nos bastidores, quanto tempo vai demorar ou quanto vai custar (FISHER et al., 2012)."

A análise de dados em escala *Big Data* demanda poderosos aglomerados de máquinas

distribuídas (*clusters*). O desempenho das tarefas em termos de **tempo de execução** e **custo monetário** [3] depende de múltiplos fatores, dentre eles, o correto aprovisionamento de *hardware*, o volume de dados envolvido e o tipo de algoritmo a ser executado. Sobre a dificuldade de tomada de decisão nestas circunstâncias Ruan, Zheng e Dong (2015) comentam:

> "As configurações de recursos existentes dependem fortemente da seleção aleatória ou da experiência anterior do implementador, geralmente levando ao aprovisionamento de recursos de baixa qualidade. Portanto, há uma necessidade urgente de propor uma abordagem para o aprovisionamento ideal de recursos para aplicativos *Spark* em nuvens públicas."

Por sua vez, os *clusters* computacionais são construídos a partir de máquinas físicas ou através do modelo de negócio de computação nas nuvens. Neste último, a escolha dos fatores se torna especialmente crítica, uma vez que o *hardware* é tarifado por fatia de tempo. Conforme Herodotou et al. (2011) [4]:

> "A maioria dos profissionais de análise de *Big Data* — como cientistas da computação, pesquisadores de sistemas e analistas de negócios; não possui o conhecimento necessário para ajustar o sistema para obter um bom desempenho. Infelizmente, o desempenho imediato do *Hadoop* deixa muito a desejar, levando ao uso subótimo de recursos, tempo e dinheiro em ambientes de computação nas nuvens".

De acordo com Ardagna et al. (2016), a modelagem do desempenho de tais sistemas é tarefa muito desafiadora, uma vez que são muito extensos, massivamente paralelos e a maioria dos atores envolvidos não são os desenvolvedores que projetam aplicações para tais ambientes. Mais interessados são os usuários dos sistemas e — nota deste Autor, os administradores dos sistemas em produção. Ampliando o raciocínio, sobre a responsabilidade do aprovisionamento correto de estrutura para a execução de algoritmos, Wu e Lee (2012) complementam que:

> "...a habilidade para coletar dados tem avançado significativamente (...) as decisões exigidas dos gerentes de *hardware* tornaram-se mais complicadas. Os centros de dados devem alocar e agendar *software* enquanto navegam na heterogeneidade e contenção. As arquiteturas devem adaptar parâmetros operacionais e recursos estruturais ao comportamento dinâmico do aplicativo".

Reforça essa ideia o pensamento de Fischer, Gao e Bernstein (2015):

[3] O termo **tempo** será utilizado ao longo da Tese como referência ao tempo total de execução de um algoritmo em minutos com precisão de duas casas decimais. O **custo** referenciará o valor monetário (R$, Real Brasileiro) cobrado pelo provedor em nuvem devido à utilização de recursos.

[4] Nota do Autor: A plataforma *Hadoop* é apresentada na Seção 2.2.2.2. O raciocínio citado pode ser aplicado a variadas plataformas de processamento para *Big Data*.

> "Ferramentas modernas de computação distribuída, como o *Apache Hadoop*, *Spark* ou *Storm* distribuem a carga de trabalho de aplicativos em um grande número de máquinas. Enquanto abstraem os detalhes da distribuição, eles exigem que o programador defina vários parâmetros de configuração antes da implantação. Essas configurações de parâmetro (geralmente) têm um impacto substancial na eficiência da execução. Encontrar os valores certos para esses parâmetros é considerada uma tarefa difícil e requer conhecimentos de domínio, aplicativo e *framework*".

A multiplicidade de fatores — tais como número de máquinas, núcleos de processamento ou quantidade de memória RAM, e seus respectivos domínios de variação, resultam em um problema de otimização combinatória. Tais problemas possuem notória dificuldade de solução devido ao vasto número de potenciais soluções e pela inviabilidade de tempo e custo para investigação de todas as possibilidades.

Diante de tal complexidade emergem os seguintes questionamentos:

- Que fatores impactam o tempo e o custo de execução de um algoritmo distribuído?
- Como mensurar e ordenar os fatores mais importantes?
- Um fator contribui positiva ou negativamente para o desempenho final?
- Como a interação entre fatores impacta tal desempenho?
- Como minimizar os esforços experimentais maximizando os resultados?
- Como estruturar e conduzir tais experimentos?
- Quantos experimentos são necessários?
- Como conferir segurança aos resultados observados?

Para derivar conclusões válidas e objetivas dá-se a necessidade de aplicação de métodos científicos sólidos que garantam completude do processo experimental e robustez da análise dos resultados observados.

1.3 HIPÓTESE

É possível quantificar o impacto isolado de fatores de configuração de *cluster* sobre o desempenho de tempo e custo de tarefas distribuídas em escala *Big Data*, especialmente aquelas de natureza iterativa, tais como aprendizagem de máquina?

Em caso positivo, pretende-se:

- identificar e ordenar os fatores significativos em relação ao referido desempenho;
- classificar as tarefas de acordo com o seu fator dominante;
- criar modelos preditores de desempenho baseados na influência dos fatores e na interação entre estes.

1.4 FUNDAMENTAÇÃO

Esta pesquisa concentra-se na análise experimental de algoritmos distribuídos de aprendizagem de máquina executados em plataformas de processamento para *Big Data*. A mesma se justifica pela necessidade de técnicas e modelos que auxiliem cientistas de dados e administradores de plataformas a proverem recursos para execução de algoritmos computacionais sobre volumes muito grandes de dados. Neste cenário, indagam-se três questões fundamentais:

(i) quais paradigmas têm sido utilizados no processamento para *Big Data*?

(ii) quais as evidências da aplicabilidade de técnicas de aprendizagem de máquina em tão volumosos conjuntos de dados?

(iii) a otimização de tempo, custo e aprovisionamento de *hardware* é uma demanda ativa da comunidade acadêmica?

Com respeito à Questão (i), já na década de 1990 se discutia a inversão da lógica de programação tradicional — código vai até o dado [5]; como forma de lidar com o crescente volume de dados (COX; ELLSWORTH, 1997). A adoção de *clusters* de máquinas distribuídas para processamento paralelo, no contexto de *Big Data*, passou a se popularizar a partir dos anos 2000, com destaque para os trabalhos seminais que introduziram novos conceitos de armazenamento distribuído (GHEMAWAT; GOBIOFF; LEUNG, 2003) e o modelo de programação *MapReduce* (DEAN; GHEMAWAT, 2004), adotados pela empresa *Google*.

Quanto à Questão (ii), trabalhos científicos ao longo da última década evidenciam pesquisas envolvendo aprendizagem de máquina sobre *Big Data* que vão de estudos de casos e revisões de literatura ao desenvolvimento de novos algoritmos, como os trabalhos de Li,

[5] Princípio do paradigma *MapReduce*, detalhado na Seção 2.2.2.1.

Deolalikar e Pradhan (2015), Dharsandiya e Patel (2016), Maillo et al. (2017). Uma lista não exaustiva pode ser conferida na Seção 2.4.

Por fim, em relação à Questão (iii), evidências da problemática de desempenho podem ser demonstradas pela busca por modelos preditores de desempenho como função do *software* e do *hardware* (WU; LEE, 2012); estudos de caso aplicados sobre ferramentas de *benchmark* (LIANG et al., 2014); análise de desempenho de *frameworks* para *Big Data* (OUSTERHOUT et al., 2015); desenvolvimento de sofisticados modelos baseados em Redes de Petri e Redes de Fila (ARDAGNA et al., 2016); e até mesmo autores que afirmam que os modelos de otimização de custo para plataformas de *Big Data* encontram-se em sua infância (RUAN; ZHENG; DONG, 2015). Um detalhamento sobre os trabalhos relacionados pode ser conferido na Seção 3.1 desta Tese.

Partindo para o enquadramento conceitual da pesquisa, conforme Ridge (2007), pesquisas acerca de algoritmos podem ser divididos em:

- estudos de dependência – analisam o relacionamento funcional entre fatores e medidas de performance de um algoritmo;
- estudos de robustez – examinam as propriedades distributivas observadas em vários ensaios aleatórios. Respondem a perguntas como: Qual o desvio da média? Qual é a faixa de desempenho em um determinado ponto do experimento?
- estudos de sondagem – medem características internas da operação de algoritmos, procurando explicar e entender suas vantagens e desvantagens. Ex: como a inicialização de centróides pode afetar o resultado final de um algoritmo de clusterização *k-means* (JAIN, 2010)? Ou ainda, como a inicialização de ninhos em uma otimização por busca do pássaro *Cuckoo* (YANG; Suash Deb, 2009) pode afetar a solução ótima local ou global?
- estudos de teste competitivo – procuram demonstrar a superioridade de um algoritmo sobre outro através de testes de referência;
- estudos aplicados – usam um código algorítmico específico em uma aplicação específica e descrevem seu impacto nesse contexto. Ex: uso de algoritmos genéticos sobre combinações de dados clínicos sobre câncer de fígado para treinamento de algoritmos de aprendizagem supervisionada.

Estudos podem ainda ser classificados quanto à sua contribuição como (BARR et al., 1995):

- rápidos – por produzir soluções de alta qualidade mais rápido que a média;
- precisos – por identificar as melhores soluções;
- robustos – por serem menos sensíveis a diferenças na qualidade dos dados do problema e seus parâmetros de ajuste;
- simples – por serem de fácil implementação;
- de alto impacto – por resolverem um novo ou importante problema;
- generalizáveis – por permitirem aplicação a um amplo domínio de problemas;
- inovadores – por serem novos ou criativos por si só.

E seus resultados podem ser valorados como: reveladores, quando oferecem informações sobre a estrutura do problema, estabelecendo as razões de seu desempenho e explicando seu comportamento; ou teóricos, quando estabelecem informações conceituais e limites na qualidade da solução.

A partir destas classificações, pode-se conferir à presente pesquisa os rótulos de (i) estudo de **dependência**, uma vez que procura identificar o resultado de efeitos de variação de múltiplos cenários de configuração computacional; e de (ii) estudo de **robustez**, dado que entrega resultados estatísticos seguros, em intervalos previsíveis. Quanto às suas contribuições a presente pesquisa introduz um método **generalizável**, aplicável a qualquer algoritmo distribuído em *clusters* de plataformas para *Big Data*, com resultados **reveladores** da dinâmica de efeitos entre os parâmetros de configuração de plataformas e seu resultado sobre métricas de desempenho.

1.5 OBJETIVOS

Estudar os impactos da variação de fatores de configuração sobre o desempenho de algoritmos distribuídos e conferir-lhes robustez estatística. Em especial, pretende-se realizar a triagem de fatores relevantes através de métodos fundamentados nos princípios de Projeto de Experimentos (do termo original em inglês *Design of Experiments* — DoE) introduzidos por Fisher e Wishart (1930), Fisher (1935). Em sentido amplo, esta pesquisa conduz estudos de **dependência** e **robustez** sobre o desempenho em relação a tempo de execução e custo

monetário de algoritmos de aprendizagem de máquina em plataformas de processamento distribuído sobre grandes volumes de dados. Por objetivos específicos, citem-se:

- investigar o relacionamento funcional entre fatores de configuração de *hardware* e de entrada de dados sobre o desempenho de um algoritmo distribuído;
- desenvolver modelos de inferência estatística capazes de quantificar os efeitos dos fatores sobre o desempenho;
- desenvolver modelos preditores para escolha de configuração para menor tempo ou custo de um algoritmo distribuído.

1.6 DESAMBIGUAÇÃO

Devido a similaridades entre áreas de conhecimento, alguns termos podem ser objeto de interpretação dúbia ou incompleta. Para completo entendimento, segue acordo de significado para compreensão inequívoca:

- *plataforma* ou *framework* – camada de *software* que gerencia a infraestrutura subjacente do *cluster* e fornece interfaces de programação para armazenamento e manipulação de dados;
- *técnica* ou *algoritmo* – conjunto de passos utilizados para resolver um determinado problema determinístico em um número finito de passos. Os termos serão utilizados de maneira indistinta com o mesmo significado;
- *classe* – separação de alto nível entre conjuntos de *técnicas* da aprendizagem de máquina que possuem objetivos similares. A saber: a classe dos algoritmos de aprendizagem supervisionada, a classe dos problemas de associação de regras, e assim por diante;
- *cluster* – aglomerado de máquinas homogêneas ligadas em rede de alta velocidade para execução de uma tarefa computacional de maneira distribuída e paralela. O termo **não deve** ser confundido com seu significado relacionado à *classe* de problemas de aprendizagem de máquina não supervisionada (clusterização);
- *tarefa* – execução computacional de uma *técnica* nas unidades do *cluster*;

- *rótulo* – categoria semântica de um domínio específico atribuída a uma instância, observação ou indivíduo de um conjunto de dados. É muito aplicado no contexto da *classe* de problemas de aprendizagem supervisionada. Embora o termo *classe* seja muito comum para se referir ao mesmo conceito, no âmbito desta Tese será realizada a distinção entre *classe* (de técnicas) e *rótulo*.

- *nó* ou *máquina* – indivíduo computacional completo de um *cluster* composto por *hardware*, sistema operacional e *softwares* específicos, utilizado para processamento de *tarefas*;

- *desempenho* – tempo total de execução de uma *tarefa* distribuída, em minutos, ou custo monetário em moeda Real Brasileiro (R$);

- *fator* – item ajustável que permite a variação de configurações de *hardware* do *cluster* ou da escala de volume de dados de entrada. Como exemplo, tome-se: número de núcleos de processamento, quantidade de memória, ou número total de nós;

- *parâmetro* – no contexto da Computação, o termo indica uma propriedade configurável de uma plataforma. Na Estatística, *parâmetro* se refere a características de uma população. O significado estará implícito na circunstância descrita no texto;

- *hiperparâmetro* – item ajustável que permite a variação de configurações na execução de um algoritmo de aprendizagem de máquina resultando em impacto sobre o seu aprendizado. Como exemplo: tome-se: taxa de aprendizado de uma rede neural, profundidade de uma árvore de decisão ou grau de flexibilidade dos limites de decisão entre *rótulos* em máquinas de vetores de suporte.

1.7 ESCOPO

Devido à amplitude das possibilidades experimentais — diversas plataformas de processamento para *Big Data*, múltiplos provedores de serviço em nuvem, variadas técnicas de aprendizagem de máquina e amplo leque de ferramentas estatísticas e matemáticas de análise de resultados; importa destacar o escopo positivo e negativo da pesquisa.

1.7.1 Escopo Positivo

Para validação da hipótese abordou-se um subconjunto de técnicas representativas da aprendizagem de máquina da classe supervisionada binária: Regressão Logística (JR; LEMESHOW; STURDIVANT, 2013), *Florestas Aleatórias* (GENUER et al., 2017), *Máquinas de Vetores de Suporte* (VAPNIK, 1982; CORTES; VAPNIK, 1995), *Perceptron* de Múltiplas Camadas (ROSENBLATT, 1958; HUSH; HORNE, 1993) e *Naïve Bayes* (MCCALLUM; NIGAM et al., 1998). Foram conduzidos um projeto experimental sobre dados sintéticos e cinco projetos experimentais sobre dados reais, totalizando 288 experimentos sobre os fatores: número de nós do *cluster*, quantidade de núcleos de processamento por nó, capacidade de memória por nó, número de discos *SSD* por nó e volume de dados de entrada. A técnica de análise foi delimitada à inferência estatística para detecção e triagem de fatores relevantes de impacto e quantificação de efeitos baseada em análise fatorial completa e fracionada em dois níveis (FISHER; WISHART, 1930; FISHER, 1935). Os experimentos restringiram-se à execução de cargas de trabalho em *cluster* dedicado no provedor de computação em nuvem *Google Cloud Data Proc*, gerenciado pela plataforma *Apache Spark*.

1.7.2 Escopo Negativo

Demais cenários correlatos **não** estão inclusos no objeto, métodos ou resultados da pesquisa. Encontram-se subdivididos em:

(i) procedimento experimental

- projetos fatoriais com três ou mais níveis;
- projetos com pontos centrais ou axiais;
- detecção de curvaturas;
- construção de superfícies de resposta.

(ii) técnicas da aprendizagem de máquina

- medidas de precisão do resultado do algoritmo, tais como curva *ROC*, especificidade, acurácia, dentre outros indicadores;
- aprendizagem não-supervisionada, regressão, filtragem colaborativa, associação de itens frequentes e demais classes de problemas;

  - seleção de características, redução de dimensionalidade, transformação de dados, otimização de modelos e demais fases da mineração de dados;
  - processamento em tempo real, tratamento de fluxos contínuos de dados e afins.

- provedores de computação em nuvem e plataformas de processamento
  - outras plataformas para processamento de *Big Data*, tais como *Flink*, H_2O e correlatos;
  - outros provedores de computação em nuvem, tais como *Amazon AWS*, *Microsoft Azure* e *Heroku*;
  - execução de cargas de trabalho em *clusters* compartilhados.

A exclusão de escopo se deu por critérios de exequibilidade e corte semântico do problema. Todavia, o método apresentado pode ser livremente adaptado à investigação de outros fatores e variáveis de resposta, em diferentes cenários.

1.8 ESTRUTURA DA TESE

A Tese está organizada em dez seções, dois apêndices e um anexo. Esta Seção 1 - Introdução abordou o problema e os objetivos de pesquisa. A Seção 2 - Big Data discute as tecnologias de processamento para grandes volumes de dados e apresenta as técnicas distribuídas a serem investigadas. A Seção 3 - Trabalhos Relacionados enumera pesquisas similares voltadas para criação de modelos para otimização no contexto citado, debate as lacunas de experimentação estatística na Ciência da Computação e apresenta a abordagem empregada na pesquisa. Na Seção 4 - Projeto de Experimentos são detalhadas as bases científicas da organização de experimentos fatoriais; enquanto a Seção 5 - Regressão Linear discute os princípios da análise de modelos lineares. A Seção 6 - Metodologia apresenta um procedimento sequencial para organização, condução e análise de experimentos sobre métricas de desempenho e criação de modelos preditores em ambientes de processamento para *Big Data*. A metodologia proposta é demonstrada através de experimentos descritos na Seção 7 - Projeto de Experimentos $2^k r$ Fatorial Completo, com Florestas Aleatórias sobre dados sintéticos; e na Seção 8 - Projeto de Experimentos $2^{k-p} r$ Fatorial Fracionado, com Florestas Aleatórias, Regressão Logística, Máquinas de Vetores de Suporte, *Naïve Bayes* e *Perceptron* de Múltiplas Camadas sobre dados

reais. Os resultados experimentais são apresentados na Seção 9 - Discussão. A Seção 10 - Conclusões enumera as contribuições da pesquisa, descobertas, aplicações, limitações e referências a trabalhos futuros. O Apêndice A - *Corpus* da Língua Portuguesa aborda a aquisição e preparação do conjunto de dados reais para os projetos experimentais fatoriais fracionados. O Apêndice B - Implementação das Tarefas de Aprendizagem de Máquina apresenta o código-fonte utilizado nos experimentos. Por fim, o leitor interessado encontrará conteúdo introdutório acerca de conceitos experimentais, testes de hipótese e bases matemáticas da regressão linear no Anexo A - Notas Teóricas Complementares.

1.9 SÍNTESE

Esta seção introduziu a Tese de pesquisa, levantou a hipótese, definiu os objetivos do trabalho e pode ser compilada da seguinte forma:

- uma série de fenômenos tecnológicos, sociais e de mercado deram origem a um novo paradigma, comumente referenciado pelo termo *Big Data*;
- a análise de dados em escala *Big Data* demanda a utilização de poderosos aglomerados de máquinas distribuídas (*clusters*);
- o desempenho de um algoritmo distribuído é influenciado pelo *hardware*, volume de dados e implementação. Fixado o tipo do algoritmo e sua implementação, o aprovisionamento ideal do *hardware* frente ao volume de dados resulta em um problema de otimização combinatória;
- *DoE* foi identificado como um processo científico seguro e robusto para investigar empiricamente o efeito de variação de fatores de configuração de *hardware* e volume para seleção de fatores relevantes no aprovisionamento de um *cluster*.

2 BIG DATA

> *If data had mass, the Earth would be a black hole.*
>
> — Stephen Marsland, *Professor*, Marsland (2014)

Esta seção enumera os conceitos relacionados com o termo *Big Data*, os desafios para manipulação de dados em volumes muito grandes e os paradigmas que se propõem a resolvê-los. Por fim, evidencia aplicações de aprendizagem de máquina e aborda questões relacionadas.

2.1 REVISÃO SISTEMÁTICA SIMPLIFICADA

Considerando o paradigma *MapReduce* e as plataformas *Hadoop* e *Spark* como soluções empregadas na aprendizagem de máquina sobre grandes volumes de dados, realizou-se um levantamento bibliográfico sobre bases de dados de trabalhos científicos endereçando as seguintes questões:

- Que classe de aplicações de aprendizagem de máquina, em termos de conotação aplicada; e problemas, em termos de conotação técnica, têm sido resolvidos com o uso de *Hadoop MapReduce* ou *Spark*?
- Que abordagens têm sido aplicadas para otimizar a infraestrutura e os parâmetros de ambiente de tais plataformas?

Foi utilizada uma *string* de busca, derivada das questões acima, sobre o título, resumo e palavras-chave de artigos publicados entre os anos 2004 [1] a 2017 nas bases listadas no Quadro 1. Ainda foram conduzidas buscas em referências cruzadas — aquelas que aparecem no corpo do artigo, mas em número pouco significativo em detrimento do total. Não se recorreu a metanálises, medições bibliométricas ou à totalidade dos passos de uma revisão sistemática de literatura em sua completude (LEVY; ELLIS, 2006; CONFORTO; AMARAL; SILVA, 2011; WEIDT; SILVA, 2016). Contudo, foram aplicados alguns de seus métodos básicos, tais como os critérios de inclusão baseados em:

- pertinência – aqueles com demonstração de aplicação teórica ou prática em aprendizagem de máquina sobre grandes volumes de dados;

[1] O limite inferior foi estipulado pelo ano de publicação do artigo "*Mapreduce: simplified data processing on large clusters*" (DEAN; GHEMAWAT, 2004), considerado seminal na área.

- formato – artigos completos publicados em conferências ou periódicos;
- atualidade – mais novos com respeito à data de publicação e levantamento do estado da arte; ou antigos, desde que referência na área.

Foram excluídos os formatos: relatório técnico, pôster, tutorial e correlatos. Também foram removidos os estudos repetidos, incompletos ou publicados em outras línguas além de Português ou Inglês.

Quadro 1 – Bases de dados utilizadas na pesquisa

Base de Dados	URL	Artigos recuperados
IEEE Xplore	www.ieeexplore.com	1013
Compendex	www.engineeringvillage.com	1142
ScienceDirect	www.sciencedirect.com	314
Scopus	www.scopus.com	1548
Springer	www.springerlink.com	232
ACM DL	dl.acm.org	329
	Total	4.578

Fonte: O autor (2020)

O processo resultou em 807 artigos. Os mais relevantes fundamentaram os conceitos e aplicações sobre *Big Data* apresentados nesta seção e os trabalhos relacionados e lacunas experimentais debatidos na Seção 3.

2.2 CONCEITOS

Park e Leydesdorff (2013) apontam os anos 1990 como primórdios do campo de pesquisa em *Big Data*. De fato, no primeiro *workshop* de descoberta de conhecimento em bases de dados, Piatetsky-Shapiro (1991) já demonstrava a preocupação com (i) o aumento do **volume** de dados — ao afirmar que o crescimento da quantidade de bancos de dados disponíveis superava amplamente o crescimento do conhecimento correspondente; com (ii) a **complexidade** dos dados — ao mencionar a necessidade futura de lidar com dados mais complexos, não apenas relacionais, incluindo texto, informações geográficas e imagens; e com (iii) a **eficiência** de processamento — ao citar que algoritmos polinomiais, exponenciais e de alta ordem não seriam escaláveis.

Ao fim do Século XX, Cox e Ellsworth (1997) já definiam *Big Data* em termos de coleções e objetos muito grandes para serem processados por algoritmos tradicionais no *hardware* disponível. Alguns objetos não poderiam ser armazenados no maior supercomputador existente à época ou ainda não seriam passíveis de manipulação por meio dos recursos típicos que um engenheiro ou cientista teriam acesso. No mesmo estudo já eram citados desafios relacionados à distribuição dos dados entre vários locais, residentes em bases de dados heterogêneas, exigência de armazenamento muito extensas, particionamento dos dados, requisitos de largura de banda para transferência de dados do disco para a memória e desta para a rede e além dela.

2.2.1 Significado

O termo *Big Data* pode ter diferentes interpretações e ser aplicado em variadas circunstâncias dependendo do contexto no qual se encontra inserido. De acordo com Reilly, Winge e Schneider (2009), deve ser utilizado em cenários nos quais o tamanho e requisitos de desempenho tornam-se fatores de decisão relevantes para implementar o sistema de gerenciamento e análise de dados. Apontam que, para uma determinada empresa, tratar centenas de *Gigabytes* (10^9 *bytes*) de dados pela primeira vez seria um gatilho para reconsiderar opções de gestão das tecnologias de processamento. Em outras circunstâncias, este gatilho pode estar dezenas ou centenas de *Terabytes* (10^{12} *bytes*) adiante, até se tornar significativo. De forma análoga, Rousseau et al. (2012) estabelecem que o termo implica em dezenas de Terabytes para algumas aplicações, ou mesmo *Petabytes* (10^{15} *bytes*) e até *Exabytes* (10^{18} *bytes*) no contexto de projetos científicos ou grandes corporações.

Laney (2001) elaborou uma já amplamente difundida associação relacionada com a manipulação de extensos volumes de dados, coloquialmente conhecida como *os 3 V's de Big Data*: **volume** — grandes conjuntos de dados; **velocidade** — que são criados em uma taxa muito alta, e **variedade** — obtidos de fontes diversas em diversos formatos. Desde então, tornou-se comum encontrar na literatura acadêmica ou de mercado uma multiplicidade de palavras-chave propondo a importância de se definir *4 V's, 5 V's, 6 V's* e assim por diante (TSAI et al., 2015).

Na busca por uma formalização conceitual, o Instituto Nacional de Padrões e Tecnologia (*NIST*, sigla em inglês) introduziu uma definição similar a Laney (2001), adicionando uma quarta característica: **variabilidade** — a mudança nas três primeiras características. E define *Big Data* como extensos conjuntos de dados primariamente com as características de volume,

variedade, velocidade e variabilidade, que requerem arquitetura escalável para armazenamento, manipulação e análise eficientes (NIST, 2015).

2.2.2 Paradigmas e Ferramentas

Manipular extensos volumes de dados incorre em desafios em todas as fases de um processo de geração de conhecimento [2]. As técnicas tradicionais de armazenamento e processamento centralizado — um sistema computacional de nó único que endereça todas as demandas; enfrentam barreiras face à natureza dos problemas em escala *Big Data*. Sobre essa limitação, Fisher et al. (2012) comentam que:

> "... *Big Data* implica na impossibilidade de manipulação e processamento de dados pelos sistemas de informação ou métodos tradicionais não apenas pelos dados serem muito grandes para serem lidos em uma máquina individual; significa também que a maioria dos métodos de mineração de dados tradicionais, desenvolvidos para um processo de análise de dados centralizado, não pode ser aplicada a dados muito volumosos".

Tsai et al. (2015) apontam o desenvolvimento de métodos baseados em computação centralizada para contornar tais limitações, especialmente em relação a problemas de aprendizagem de máquina, tais como: classificação, clusterização, associação de regras, dentre outros. Entretanto, os autores observam que muitos dos problemas relacionados à análise de *Big Data* ainda assim não podem ser endereçados, pois incorrem em:

- ausência de escalabilidade, pois foram projetados para executar com pequenos volumes de dados não complexos, em máquinas individuais nas quais todos os dados são injetados em memória para o processo de análise;
- falta de dinamismo, dado que não se ajustam a situações em tempo real;
- estrutura uniforme, uma vez que a maioria dos métodos assume que o formato de dados será sempre o mesmo.

Assim, *Big Data* incorre em gerenciamento de dados massivos, multidimensionais, heterogêneos, complexos, não estruturados, incompletos, sujeitos a ruído e erro; o que pode levar a mudanças na abordagem estatística e de análise de dados (MA; ZHANG; WANG, 2014).

[2] Fases do processo de descoberta de conhecimento: seleção, pré-processamento, transformação, avaliação e interpretação (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

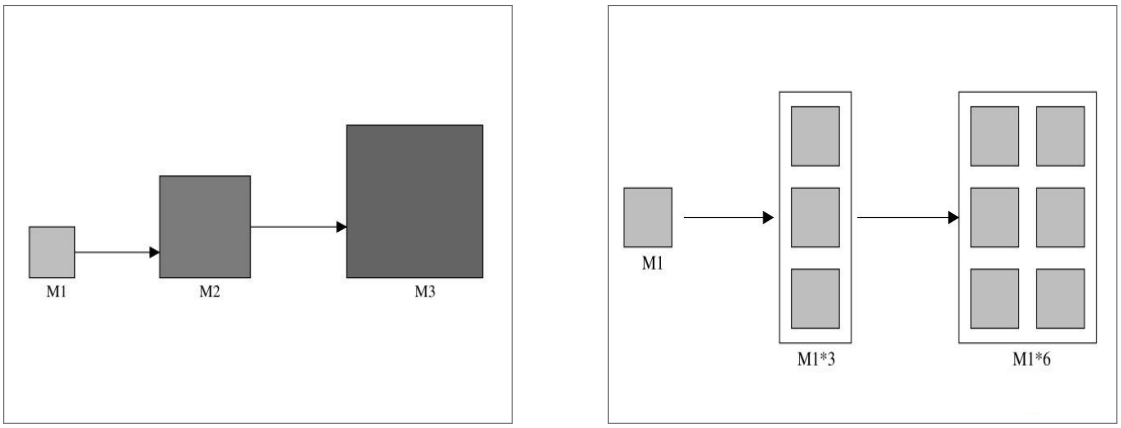
A complexidade associada com a manipulação de *Big Data* também demanda mudanças nos paradigmas de armazenamento e processamento. Surge a necessidade de empregar tecnologias capazes de lidar com problemas de escalabilidade, muitas vezes em tempo real, com suporte para redundância e tolerância a falhas. Estas características podem ser viabilizadas através de computação paralela distribuída, tirando proveito do poder computacional de agregados (*clusters*) de máquinas, normalmente compostas por *hardware* de baixo custo gerenciadas por sistema operacional de código aberto, a exemplo das diversas distribuições *Linux*. Por economia financeira, recomenda-se o aprovisionamento através de *hardware* convencional em detrimento do emprego de supercomputadores, dado que é mais barata e fácil a aquisição, manutenção e reposição em caso de falhas.

Quando a necessidade de processamento existe por um curto espaço de tempo torna-se desvantajoso aplicar recursos financeiros em infraestrutura permanente. Assim, o emprego de modelos de negócios baseados em computação nas nuvens surge como alternativa. Segundo Mell, Grance et al. (2011) trata-se de um modelo computacional para acesso conveniente, sob demanda e de qualquer localização, a uma rede compartilhada de recursos computacionais (servidores, armazenamento, aplicativos e serviços) que podem ser prontamente disponibilizados com esforço mínimo de gestão ou de interação com o provedor.

As nuvens computacionais podem ser concebidas e implantadas na forma de modelos funcionalmente distintos, representados por *Everything as a Service* — *XaaS*, onde **X** pode significar *software*, plataforma ou infraestrutura (FOX et al., 2009). Portanto, é comum que uma nuvem computacional forneça infraestrutura elástica e dinamicamente gerenciada, em geral no modelo *IaaS* (Infraestrutura como um Serviço), na forma de máquinas virtuais completas com seus respectivos recursos de *hardware* virtualizados (rede, disco, memória, processador, dentre outros).

As nuvens possuem ainda uma característica muito apropriada aos problemas em escala *Big Data*: quando o poder computacional disponível não endereça mais as necessidades de processamento geradas pelo volume de dados e pela velocidade com que tais dados crescem, faz-se necessário um aumento nos referidos recursos. Para que tal aumento sob demanda funcione, é necessário que o sistema esteja preparado para o redimensionamento. Se um sistema computacional permanece eficiente mediante o correto aprovisionamento de infraestrutura perante um aumento significativo em sua demanda, pode-se dizer que este é escalável. Basicamente, há duas formas de promover escalabilidade em sistemas computacionais (HUAI et al., 2011):

Figura 1 – Escalabilidade de sistemas



(a) Escalabilidade vertical. (b) Escalabilidade horizontal.

Fonte: O autor (2020), adaptado de Tsai et al. (2015)

- vertical, incremento individual do poder de processamento de máquinas dedicadas e da expansão de memória e capacidade de armazenamento;
- horizontal, expansão de grandes sistemas distribuídos através da adição contínua de unidades de processamento e armazenamento conectadas em rede.

A Figura 1a traz um exemplo de escalabilidade vertical no qual componentes são modificados ou mesmo o sistema computacional inteiro é substituído por uma nova versão mais poderosa. A Figura 1b representa um esquema de escalabilidade horizontal, onde várias unidades *M1* juntas ampliam a capacidade computacional total do sistema.

Sobre os requisitos de desempenho para arquiteturas no paradigma *Big Data*, NIST (2015) recomenda a adoção de uma infraestrutura horizontal, ou seja, a distribuição de sistemas de dados através de recursos independentes, horizontalmente agrupados, para alcançar a escalabilidade necessária para o processamento de extensos conjuntos de dados.

Há diversas abordagens para obter maior desempenho computacional através de escalabilidade horizontal (BAKER; BUYYA, 1988), seja com o intuito de apoiar pesquisas científicas em problemas essencialmente complexos no campo da Física ou Matemática e afins, ou para atender demandas de indústria. Dentre estes, o paradigma *MapReduce* emergiu como um padrão *de facto* para o processamento de grandes volumes de dados.

2.2.2.1 Paradigma MapReduce

Trata-se de um dos primeiros paradigmas bem-sucedidos a lidar com *Big Data* de maneira escalável. Foi introduzido pela empresa *Google* ™ em 2003 (DEAN; GHEMAWAT, 2004; DEAN; GHEMAWAT, 2008) em resposta ao crescimento vertiginoso do volume de dados de seus produtos. Executa sobre *GFS — Sistema de Arquivo Google* (GHEMAWAT; GOBIOFF; LEUNG, 2003), uma abstração para sistemas de arquivos distribuídos onde grandes volumes de dados são lidos e processados localmente em cada nó e redistribuídos por todo o *cluster*. Acredita-se que seja o paradigma mais adequado para o processamento de *Big Data*, inclusive algoritmos de aprendizagem de máquina (CHU et al., 2006).

Após seu surgimento, um conjunto de aprimoramentos e ferramentas foram desenvolvidos, tais como *Spark* (ZAHARIA et al., 2012), *Flink* (CARBONE et al., 2015), e *Storm* (IQBAL; SOOMRO, 2015). Em paralelo, surgiram bibliotecas de *software* específicas para aprendizagem de máquina distribuída, a exemplo de *SAMOA* (MORALES, 2013), *Mahout* (WITHANAWASAM, 2015) e *Spark MLLib* (MENG et al., 2016). Destaque-se que não há um conjunto de ferramentas que combine todos os recursos possíveis. Cabe ao usuário decidir sobre a tecnologia, paradigma ou plataforma com base no problema a ser resolvido. As subseções a seguir detalham brevemente a implementação *open source* da abstração *MapReduce* — o *Hadoop MapReduce*, e introduzem *Spark*, a plataforma adotada para executar as cargas de trabalho desta Tese. Maiores detalhes sobre o paradigma *MapReduce* e as plataformas mencionadas podem ser encontrados em Lin e Dyer (2010) e Landset et al. (2015).

2.2.2.2 Hadoop MapReduce

É possível referir-se a *MapReduce* como um paradigma — modelo de programação; ou como um *software* que implementa esse modelo, a exemplo das implementações *Google MapReduce* ou *Hadoop MapReduce*. Análogo ao primeiro, *Hadoop* utiliza *HDFS — Sistema de Arquivo Distribuído Hadoop* (BORTHAKUR, 2007), cuja arquitetura define dois tipos de nó: (i) um *namenode* [3], para gerenciar o espaço de nomes, decidir questões de escalonamento de tarefas, distribuição de dados, dentre outras questões; e (ii) e *n datanodes*, responsáveis por armazenar e processar os blocos de dados. A portabilidade entre sistemas operacionais,

[3] Normalmente um *cluster* de processamento para *Big Data* possui apenas um *namenode*, embora arquiteturas mais complexas possam acoplar mais *namemodes* para efeitos de alta disponibilidade.

a transparência sobre a infraestrutura subjacente, os recursos de tolerância a falhas e a disponibilidade de distribuições em código aberto tornam o *Hadoop* uma solução amplamente aceita e usada pela academia e indústria, bem como por grandes empresas de Tecnologia da Informação, como *Yahoo* ™, *Facebook* ™, *eBay* ™ e outras (MAITREY; JHA, 2015).

2.2.2.3 Conjuntos de Dados Distribuídos Resilientes

Hadoop realiza muitas operações de entrada e saída (E/S) em disco no âmbito de cada *datanode*, prejudicando seu desempenho, especialmente em algoritmos iterativos. Para solucionar esse problema, o *Spark* surgiu como um sistema de baixa latência compatível com o *HDFS* e outros mecanismos de armazenamento distribuído [4]. É um *framework* baseado em RDDs, ou Conjuntos de Dados Distribuídos Resilientes (ZAHARIA et al., 2012), uma abstração de memória que procura minimizar o gargalo de E/S do *Hadoop* com expressividade suficiente para capturar uma ampla classe de tarefas computacionais, incluindo *MapReduce* e modelos de programação especializados para tarefas iterativas. Landset et al. (2015) descrevem as vantagens que distinguem o *Spark* de outras plataformas. Por fim, Shi et al. (2015) realizam uma comparação meticulosa entre *Spark* e *Hadoop*, apontando *Spark* como a plataforma de maior desempenho geral.

2.3 ALGORITMOS DISTRIBUÍDOS DE APRENDIZAGEM DE MÁQUINA

Foge ao escopo da Tese apresentar conceitos detalhados sobre aprendizagem de máquina. Para tal, o leitor interessado encontrará conteúdo detalhado em Mitchell (1997), Mitchell, Carbonell e Michalski (2012), Hastie, Tibshirani e Friedman (2013). Para nivelamento e compreensão dos experimentos, esta seção discute brevemente os conceitos e algoritmos utilizados na pesquisa.

A aprendizagem de máquina (AM) consiste em fazer os computadores aprenderem a partir de exemplos para realizar previsões, classificar objetos ou controlar recursos. De acordo com Freiesleben, Keim e Grutsch (2020):

> "A aprendizagem de máquina — capacidade de auto-aprendizado dos al-

[4] Diversas abordagens foram concebidas para lidar com as limitações do modelo *MapReduce*, tais como *Pregel* (MALEWICZ et al., 2010), *HaLoop* (BU et al., 2010) e *Twister* (EKANAYAKE et al., 2010), dentre outras. Todavia, esses modelos oferecem padrões de comunicação restritos para classes específicas de aplicativos, enquanto a proposta *Spark* provém uma abstração mais generalizável.

> goritmos para estruturar e interpretar dados de forma autônoma; é uma abordagem metodológica para resolver problemas de otimização complicados com base em dados abundantes. AM tem ganho força à medida que aplicativos algorítmicos, poder computacional e disponibilidade de conjuntos de dados vêm aumentando nas últimas duas décadas, fornecendo um ambiente rico em informações em que o raciocínio humano pode ser parcialmente substituído pelo raciocínio computacional".

Tais técnicas computacionais passaram de uma conotação restrita para uma multiplicidade de aplicações nos primeiros anos do Séc. XXI. Muitos algoritmos ou teorias que antes eram aplicados em cenários limitados tornaram-se ferramentas para aplicações reais. De fato, conforme Lemnaru et al. (2012):

> "...aplicações empregando técnicas de mineração de dados para uma variedade de tarefas têm se tornado ubíquas, variando de sistemas triviais de seleção de propaganda até engenhos de recomendação complexos, que consideram uma grande variedade de aspectos relacionados ao usuário e aos produtos, ou até aplicações mais sensíveis como diagnóstico médico e detecção de fraude, para nomear algumas".

As diferentes técnicas de AM podem ser agrupadas em: supervisionadas — os casos mais comuns, abrangendo classificação e regressão; não supervisionadas; aprendizagem por reforço; e aprendizagem evolutiva (MARSLAND, 2014).

Conforme apresentado na Seção 1.7, a pesquisa se concentra em técnicas de aprendizagem de máquina supervisionada, especificamente: Regressão Logística, Máquinas de Vetores de Suporte, *Naïve Bayes*, Florestas Aleatórias e *Perceptron* de Múltiplas Camadas, detalhadas em sequência.

A operação de classificação implica em decidir o pertencimento de um item de dados desconhecido $x^{'}$ a um rótulo $y^{'}$ com base em um conjunto $D = (\mathbf{x_1}, y_1), ..., (\mathbf{x_n}, y_n)$ de dados $\mathbf{x_i}$ cujas associações y_i sejam conhecidas. Geralmente, $\mathbf{x_i}$ é um vetor *m*-dimensional cujos componentes são chamados de variáveis independentes, na linguagem estatística; ou variáveis de entrada, pela comunidade de aprendizagem de máquina (DREISEITL; OHNO-MACHADO, 2002).

Muitos métodos de AM podem ser formulados como um problema de otimização que busca minimizar uma função convexa que depende um vetor de parâmetros **w** com *d* dimensões. Formalmente, dado um conjunto de treinamento com pares (instância, rótulo) $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}\, \forall i$, $1 \leq i \leq n$, o problema de otimização $min_{\,\mathbf{w}\in\mathbb{R}^d}\, f(\mathbf{w})$ pode ser descrito por 2.1 (LIN et al., 2015):

$$\min_{\mathbf{w}} \quad f(\mathbf{w}) \equiv \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi(\mathbf{w}; \mathbf{x}_i, y_i), \tag{2.1}$$

onde $\xi(\mathbf{w}; \mathbf{x}_i, y_i)$ é uma função de custo e $C > 0$ é um parâmetro especificado pelo usuário. O leitor pode se aprofundar sobre funções de custo em Wald (1950).

2.3.1 Regressão Logística (LR)

Quando a função de custo é dada por $\log(1 + e^{-y_i \mathbf{w}^T \mathbf{x}_i})$, tem-se a aplicação da função logística de Verhulst (1838), empregada para modelar a probabilidade de um determinado evento ocorrer. No modelo logístico, o logaritmo das probabilidades para o rótulo positivo é uma combinação linear de uma ou mais variáveis independentes. Estas podem ser binárias ou contínuas. A probabilidade correspondente ao rótulo positivo pode variar entre 0 (certamente a instância associada ao rótulo negativo) e 1 (certamente a instância associada ao rótulo positivo). O leitor interessado pode conferir um detalhamento aprofundado em Jr, Lemeshow e Sturdivant (2013), além de um levantamento histórico sobre a regressão logística em Cramer (2003), Kint, Constales e Vanderbauwhede (2006). Os experimentos descritos na Seção 8.2 utilizam a implementação *Spark* binomial para classificação binária.

2.3.2 Máquinas de Vetores de Suporte (SVM)

Trata-se de uma técnica utilizada para classificação e regressão (VAPNIK, 1982; CORTES; VAPNIK, 1995) capaz de separar dados através de uma fronteira no espaço multidimensional. Mapeiam o espaço de entrada em um espaço de alta dimensionalidade através de mapeamentos não lineares e constroem um hiperplano ótimo de separação, tornando possível construir superfícies de decisão lineares no espaço de características que correspondem a superfícies não-lineares no espaço de entrada (STITSON et al., 1996).

O algoritmo se baseia em encontrar os vetores de suporte — os pontos mais próximos do hiperplano de separação que influenciam sua posição e orientação. O SVM é obtido substituindo-se a função de custo $\xi(\mathbf{w}; \mathbf{x}_i, y_i)$ por $max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i)$ ou $max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i)^2$, na Equação 2.1 (LIN et al., 2015).

Os experimentos da Seção 8.3 utilizam a implementação *Spark* cuja função de otimização pode ser compreendida em detalhes em Moore e DeNero (2011).

2.3.3 *Naïve Bayes* (NB)

Trata-se de uma família de classificadores probabilísticos fundamentados no Teorema de Bayes (1763) que trabalham com probabilidade condicional e pressupõem independência entre os dados. Devido a esta suposição os parâmetros para cada atributo podem ser aprendidos separadamente, simplificando bastante o aprendizado, especialmente quando o número de atributos é grande (MCCALLUM; NIGAM et al., 1998). Embora a independência seja uma fraca suposição, na prática os classificadores *bayesianos* competem bem com classificadores mais sofisticados (KIM et al., 2006).

Conforme Mitchell (1997), o classificador se aplica a tarefas de aprendizagem onde cada instância $\mathbf{x}$ é descrita por uma conjunção de valores de atributos e onde a função alvo $f(\mathbf{x})$ pode assumir qualquer valor de algum conjunto finito V. Um conjunto de exemplos de treinamento é fornecido e uma nova instância é apresentada, descrita pela tupla de valores de atributo $(a_1, a_2 \ldots a_n)$. A classificação consiste em atribuir o valor v_{map} mais provável dados os valores de atributo que descrevem a instância, de acordo com a Equação 2.2 a seguir:

$$v_{map} = \underset{\mathbf{v_j} \in \mathbf{V}}{\operatorname{argmax}} \; P(a_1, a_2 \ldots a_n | v_j) P(v_j) \tag{2.2}$$

Em virtude da suposição de que os valores dos atributos são condicionalmente independentes, a probabilidade de observar a conjunção $a_1, a_2 \ldots a_n$ é apenas o produto das probabilidades para os atributos individuais: $P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$. Dessa forma, o valor v_{nb} de saída para um classificador *Naïve Bayes* consiste em:

$$v_{nb} = \underset{\mathbf{v_j} \in \mathbf{V}}{\operatorname{argmax}} \; P(v_j) \prod_i P(a_i | v_j). \tag{2.3}$$

Detalhes sobre o *framework* probabilístico e o Teorema de *Bayes* podem ser conferidos em Mitchell (1997), McCallum, Nigam et al. (1998).

2.3.4 Florestas Aleatórias (RF)

A compreensão da técnica Florestas Aleatórias demanda o conceito de árvores de decisão (BREIMAN et al., 1984) — mecanismos de previsão e classificação de propósito geral que constam entre os primeiros algoritmos estatísticos e cujo desenvolvimento transpassa décadas, dos trabalhos de Belson (1956) até os aprimoramentos apresentados por Hawkins (1982)

e Breiman et al. (1984). Sua principal característica reside na capacidade de subdividir recursivamente um conjunto de dados de acordo com variáveis de entrada (ou preditoras), criando partições associadas (nós ou folhas). Tais partições possuem progressivamente nós/folhas similares internamente e dissimilares entre si (VILLE, 2013). Árvores de decisão preveem o rótulo de um subconjunto (nó) ou instância (folha) de um conjunto de dados D a partir da inferência sobre a melhor divisão de um grupamento de divisões possíveis a fim de maximizar o ganho de informação. Matematicamente, a divisão escolhida em cada nó se dá por $\underset{\mathbf{s}}{\operatorname{argmax}}\, IG(D, s)$, onde $IG(D, s)$ é o ganho de informação obtido. Para problemas de classificação, o ganho de informação pode ser calculado pelo índice de *Gini* ou pela entropia, conforme Quadro 2, onde f_i é a frequência do rótulo i no nó e C é o número de rótulos únicos.

Quadro 2 – Métricas de impureza e ganho de informação para árvores de decisão

Índice	**Equação**
Gini	$\sum_{i=1}^{C} f_i\,(1 - f_i)$
Entropia	$\sum_{i=1}^{C} -f_i\, log(f_i)$

Como comitês (*ensembles*) de árvores de decisão, as florestas aleatórias consistem em uma poderosa técnica estatística não-paramétrica utilizada tanto para classificação quanto para problemas de regressão. Formalmente, uma Floresta Aleatória é um classificador que consiste em uma coleção de árvores de decisão $\{h(\mathbf{w}, \Theta_k), k = 1, ...\}$ onde $\{\Theta_k\}$ são vetores randômicos identicamente distribuídos e cada árvore emite um voto unitário para o rótulo mais popular com relação ao vetor de entrada **w** (BREIMAN, 1999; BREIMAN, 2001). Uma revisão atualizada sobre Florestas Aleatórias, sua relevância em aplicações *Big Data* e as considerações para implementações distribuídas podem ser conferidas em Genuer et al. (2017). Os experimentos das Seções 7.1 e 8.5 utilizam a implementação *Spark* com ganho de informação baseado em índice de *Gini*.

2.3.5 *Perceptron* de Múltiplas Camadas (MLP)

As teorias acerca das redes neurais artificiais tiveram início com o trabalho de McCulloch e Pitts (1943) através da proposição de uma lógica *booleana* para descrição de eventos neurais e suas relações. De acordo com Abraham (2002), o trabalho foi fundamental para o conhecimento cognitivo moderno, particularmente para a Inteligência Artificial, que surgiu aderente à concepção de que o comportamento inteligente poderia ser imitado por um computador

digital. O conceito matemático *Perceptron* foi concebido por Rosenblatt (1958), todavia as capacidades do *Perceptron* eram limitadas às fronteiras de decisão lineares e a funções lógicas simples. Nesse ínterim, posteriores desenvolvimentos agregando *perceptrons* em camadas mostrou a capacidade do *Perceptron* de Múltiplas Camadas (MLP) de implementar decisões complexas e expressões booleanas arbitrárias.

Os experimentos da Seção 8.6 empregam a implementação *Spark* de um classificador baseado em rede neural *feedforward*, onde cada camada está totalmente conectada à próxima camada na rede. Os nós (neurônios) na camada de entrada representam os dados de entrada. Todos os outros nós mapeiam entradas para saídas por uma combinação linear das entradas com vetores de pesos, aplicando uma função de ativação. Os nós intermediários da rede usam função de ativação sigmoide (logística):

$$f(\mathbf{x_i}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x_i}}}. \tag{2.4}$$

Já a camada de saída com K rótulos utiliza uma função *softmax*:

$$f(\mathbf{x_i}) = \frac{e^{\mathbf{w}^T \mathbf{x}_i}}{\sum_{j=1}^{K} e^{\mathbf{w}^T \mathbf{x}_j}}, \tag{2.5}$$

O leitor interessado encontrará estudos de revisão dos aprimoramentos e aplicações das redes neurais artificiais em Hush e Horne (1993), Zhang (2000) e Tkáč e Verner (2016).

2.4 APLICAÇÕES DE APRENDIZAGEM DE MÁQUINA SOBRE *BIG DATA*

Esta seção enumera evidências de aprendizagem de máquina sobre *Big Data* presentes na literatura acadêmica e ressalta a importância do tema apresentando trabalhos que aplicam o paradigma *MapReduce* e derivados.

Problemas como clusterização, regras de associação, regressão, classificação, redução de dimensionalidade, dentre outros, já demandam computação intensiva no âmbito da computação tradicional centralizada sobre pequenos volumes de dados. Quando submetidos a dados em escala *Big Data*, tais demandas tornam-se ainda maiores, incorrendo em notório custo computacional.

A classe de problemas relacionadas à **clusterização**, por exemplo, está entre aquelas mais impactadas pelo volume dos dados. Em um estudo que procura estabelecer uma taxonomia para técnicas de clusterização aplicadas a *Big Data*, Fahad et al. (2014) indicam quais

teriam resultados satisfatórios, que seriam: *Fuzzy C-Means*, *BIRCH*, *DENCLUE*, *Expectation-Maximization* e *OptiGrid*. Entretanto, o estudo não cita *frameworks* de processamento tais como *MapReduce*. Shirkhorshidi et al. (2014) dividem as técnicas de clusterização em duas categorias, aquelas que podem lidar com *Big Data* em máquinas simples — através de amostragem e redução de dimensionalidade — e a clusterização para múltiplas máquinas em paralelo, caso para o qual indicam soluções baseadas em *MapReduce*. Diversas aplicações de clusterização sobre *Big Data* são encontradas na literatura em estudos que variam de predição de parâmetros de qualidade de serviço (*QoS*) em redes móveis (THOMAS et al., 2011) até a análise de doenças cardiovasculares (YANG; ZHANG; ZHANG, 2013); aplicações no mercado de ações (LACHIHEB; GOUIDER; SAID, 2015); e trabalhos aplicados à análise de informações anônimas sensíveis (KHAN et al., 2020).

A classe de problemas de **mineração de itens frequentes**, desde suas primeiras implementações (GOLDBERG et al., 1992), sempre constituiu um desafio dado que geralmente o processo costuma ser empregado em milhares de registros de varejo, recuperados de transações de lojas em centros de compra. Com o advento das bases em escala *Big Data* o problema tornou-se ainda maior. Neste contexto, Zhou et al. (2010) propõem um algoritmo chamado *BPFP - Balanced Parallel FP-Growth* que paraleliza o algoritmo *FP-Growth* em uma abordagem *MapReduce*. A estratégia busca dividir toda a tarefa computacional em subtarefas relativamente simples, o que possibilita a paralelização. Em Kovacs e Illés (2013), os autores propõem uma abordagem para tornar os algoritmos baseados na técnica *Apriori* mais eficientes em cenários distribuídos, incluindo pseudo-códigos da implementação das fases *map* e *reduce*. Xun et al. (2017) identificaram sérios problemas de desempenho na execução de algoritmos paralelos para mineração de itens frequentes. Os autores propuseram uma técnica chamada *FiDoop-DP* que descobre similaridades entre transações e as colocam na mesma partição de dados para tirar proveito do princípio da localidade, evitando sobrecarga no *cluster*, reduzindo o tempo de execução em até 31%. Mesmo que as técnicas de mineração de itens frequentes baseadas em *MapReduce* resultem em melhoria de eficiência em detrimento das técnicas tradicionais sobre grandes volumes de dados, a ocorrência intensiva de operações de E/S ainda provoca alta latência. Este debate é trazido à tona por Rathee, Kaul e Kashyap (2015) que relacionam implementações do algoritmo *APriori* no ecossistema *Hadoop* e apresentam a técnica denominada *R-Apriori*, baseada em computação iterativa em memória com o uso do framework *Spark*. Muitas evidências apontam o uso de regras de associação no contexto *Big Data*: sistemas de recomendação de programas para TV Digital (LAI et al., 2011); descoberta de regras

de associação em crimes cibernéticos (JIANG; SUN, 2013); suporte ao processo de mineração de dados em sistemas de relacionamento com o cliente (LI; DEOLALIKAR; PRADHAN, 2015); e detecção de doenças em grandes volumes de dados compostos por informações oriundas de colaboração médica de vários lugares do mundo (JOY; SHERLY, 2016).

Também se encontram na literatura abordagens relativas à classe de problemas de **computação evolucionária**. O estudo de Zhu e Wang (2010) sobre mineração de comunidades — muito útil para descoberta de organizações de terrorismo, comunidades web, em mecanismos de busca, e outros mais; aplicou *MapReduce* e algoritmos genéticos para implementar um novo tipo de operador de *crossover* tendo como resultado o aumento da velocidade de convergência e a melhor detecção de comunidades. Salza, Ferrucci e Sarro (2016) apresentam a plataforma de *open source elephant56* para encorajar desenvolvedores a aplicar técnicas de computação evolucionária sobre *MapReduce*. Por fim, El-Alfy e Alshammari (2016) aplicam algoritmos genéticos paralelos em problemas de detecção de intrusão em quatro bases de dados distintas de *logs* de acesso de usuários em rede.

No âmbito dos problemas de **classificação**, o tempo requerido para treinamento de redes neurais artificiais em escala *Big Data* torna proibitivo o uso de técnicas tradicionais centralizadas. Motivados por esta questão Gu, Shen e Huang (2013) propuseram a *cNeural*, uma plataforma paralela para treinamento de redes neurais com *backpropagation*, com desempenho até 50 vezes melhor que soluções similares baseadas também em *Hadoop MapReduce*. Liu et al. (2015) produziram uma vasta e detalhada especificação sobre arquiteturas de treinamento de rede neural paralela com *backpropagation*. Os autores criticam negativamente a sobrecarga causada pelo número excessivo de operações de E/S do *framework MapReduce*, destacando a necessidade de aplicar a técnica sobre plataformas que permitam a computação iterativa. Maillo et al. (2017) trazem uma proposta de implementação da técnica *k-Nearest Neighbors - kNN*, denominada *KNN-IS*, que utiliza o *framework Spark* em uma base de dados com 11 milhões de instâncias. Como resultado os autores argumentam que obtiveram o mesmo nível de acurácia que métodos tradicionais de *kNN*, sem os problemas de execução e consumo de memória encontrados no modelo de computação centralizada tradicional. Alham, Li e Liu (2014) destacam o caráter computacionalmente intensivo da técnica *Support Vector Machine (SVM)*, de complexidade $O(m^2n)$, onde *n* é a dimensão da entrada e *m* representa o número de instâncias de treinamento; ou seja, é quadrático em termos do número de instâncias de treinamento. Os autores apresentaram o algoritmo *MapReduce Multiclass SVM (MRMSVM)*, uma abordagem paralela multiclasse do *SVM* processada em ambiente distribuído. São apontadas

na literatura diversas aplicações bem sucedidas de classificação sobre *Big Data*, a exemplo de: reconhecimento de imagens faciais (ZHANG; LI; JIA, 2014); previsão do tempo a partir de dados históricos de 63 anos de clima (1951 – 2013) do *India Meteorological Department* (NAMITHA; JAYAPRIYA; KUMAR, 2015); árvores de decisão (implementação paralela do algoritmo *C 4.5*) para predição de doenças do coração (MANN; KAUR, 2015); e florestas randômicas para análise de eletrocardiogramas (MOHAPATRA; MOHANTY, 2020).

Outras abordagens constam em literatura recente, tais como extração de características (KHAN et al., 2019), aprendizagem profunda (HOSSAIN; MUHAMMAD, 2019; JAN et al., 2019; AMANULLAH et al., 2020); redução de dimensionalidade (REDDY et al., 2020) e mais.

Este rol de trabalhos apresenta apenas uma amostra ilustrativa de aplicações de aprendizagem de máquina sobre *Big Data*. Com isto, buscou-se a evidência de contemporaneidade da área na qual se encontra inserido este estudo e de sua larga aplicação, especialmente na comunidade acadêmica.

2.5 SÍNTESE

Esta seção aprofundou os conceitos relacionados com *Big Data*, paradigmas e ferramentas para processamento, apresentou técnicas de aprendizagem de máquina e evidenciou o emprego destas na resolução de problemas reais. A seção pode ser sumarizada por:

- levantamento sobre o termo *Big Data*, suas origens, aplicações e conceitos formais;
- apresentação de *MapReduce* como paradigma de programação emergente para tratamento eficiente das demandas de volume, variedade e velocidade do contexto *Big Data*;
- análise evolutiva e comparativa de *frameworks* como *Hadoop* e *Spark*, inclusas as bibliotecas para execução de tarefas de aprendizagem de máquina;
- evidências de emprego de aprendizagem de máquina em problemas de clusterização, classificação, redução de dimensionalidade, associação de regras e aprendizagem profunda;
- detalhamento teórico sobre os algoritmos Regressão Logística, Máquinas de Vetores de Suporte, Florestas Aleatórias, *Naïve Bayes* e *Perceptron* de Múltiplas Camadas, todos empregados nos experimentos desta Tese.

3 TRABALHOS RELACIONADOS

> *We've got numbers by the trillions. Here and overseas.*
>
> — Paul Frederic Simon, *Musical Composer*, Simon (1983)

Esta seção enumera diferentes esforços conduzidos pela comunidade acadêmica na busca por técnicas ou modelos eficientes de tarefas computacionais em plataformas para *Big Data*. Ao final são apresentadas: uma discussão acerca das lacunas de emprego de projetos de experimentos estatisticamente válidos na Ciência da Computação, uma análise comparativa das técnicas estudadas e a abordagem experimental empregada na Tese.

3.1 ESTUDOS DE DESEMPENHO E OTIMIZAÇÃO

A otimização do desempenho de tarefas em plataformas para processamento de *Big Data* tem atraído o interesse de pesquisadores ao longo dos anos. As pesquisas envolvem desde plataformas de *benchmarking* e estudos comparativos de desempenho de algoritmos específicos até modelos preditores de desempenho, algoritmos de escalonamento de tarefas, automação de escolha de parâmetros de configuração, gerenciamento de localidade de dados entre nós, dentre outros.

3.1.1 Algoritmos de Escalonamento

As modernas plataformas de processamento distribuído possuem algoritmos para atribuição de tarefas em nós do *cluster* de acordo com heurísticas que consideram a localidade do dado no sistema de arquivo distribuído, na distribuição física dos nós, tamanho do *cluster*, dentre outros. Nesse contexto, Zaharia et al. (2008) abordam a forte relação entre o tempo de execução em *clusters Hadoop* e seu agendador de tarefas e criticam a suposição de homogeneidade do cluster e a linearidade de progressão das tarefas. Tal fato impacta na forma como o *framework* decide reexecutar tarefas que parecem ser retardatárias, causando severas degradações de desempenho em ambientes heterogêneos. Os autores propõem um novo algoritmo de agendamento denominado *Longest Approximate Time to End (LATE)* que consegue melhorar em até 2x os tempos de resposta das aplicações *MapReduce* em ambiente *Amazon*

Elastic Cloud Computing (ECC). A mesma crítica à premissa de homogeneidade do *Hadoop* foi abordada no trabalho de Arasanal e Rumani (2013) que propõem um algoritmo adaptativo que balanceia a distribuição de dados entre nós do cluster. Experimentos conduzidos em dois cenários distintos com dados reais melhorou os tempos de resposta da aplicação. Dentre as principais contribuições do trabalho, foram propostos um modelo matemático para estimar as taxas de computação com base nas especificações de *hardware*, como velocidade da CPU e tamanho da memória física; e uma ferramenta para distribuição de dados no *cluster*.

3.1.2 Configuração de Parâmetros

Considerando a multiplicidade de parâmetros de configuração das diversas plataformas de processamento para *Big Data*, o processo de autoajuste viabiliza a execução de tarefas distribuídas de maneira transparente ao usuário.

Nesse ínterim, Herodotou et al. (2011) introduziram *StarFish*, um sistema de autoajuste de cargas de trabalho para alcançar bom desempenho automaticamente, dispensando usuários de entenderem e manipularem as diversas opções de ajuste do *framework Hadoop*. A abordagem declina de oferecer o pico ótimo de desempenho através de uma hipotética configuração manual, em detrimento, viabiliza que os aplicativos obtenham desempenho aceitável de forma automática.

O ajuste de parâmetros de *framework* não é trivial e requer tempo e recursos humanos altamente qualificados. Nesse contexto, Pospelova (2015) apresenta um sintetizador automático de configuração de parâmetros para plataformas *MapReduce/Hadoop* no gerenciador de *clusters Yarn*. A autora argumenta que as abordagens de ajuste manual existentes exigem várias execuções da mesma tarefa para encontrar configurações ideais e, portanto, não são aplicáveis ao uso na maioria dos casos. A ferramenta desenvolvida intercepta uma solicitação de recurso, modifica-a de acordo com um algoritmo de ajuste e passa para o agendador de tarefas. Embora bastante sofisticada e carregue alto valor prático e teórico, a abordagem carece de intrusão, ou seja, da inserção desta nos módulos de execução do *framework*.

Zhang, Li e Hildebrand (2015) desenvolveram um mecanismo que recomenda configurações para uma tarefa analítica enviada ao *cluster*. A técnica utilizou aprendizagem de máquina baseada em uma adaptação do algoritmo *k-Nearest Neighbors (KNN)*, que encontra configurações desejáveis de tarefas anteriores semelhantes que tiveram um bom desempenho. O método resultou em ganhos de até 28% no tempo de processamento. É importante destacar

que este tipo de abordagem deriva modelos para um conjunto específico de tarefas históricas em um conjunto específico de configuração de *hardware* e tamanho de *cluster*.

Fischer, Gao e Bernstein (2015) propõem um processo automatizado para escolha de parâmetros de configuração baseado em otimização bayesiana. Em uma extensa avaliação empírica, os autores mostram que a otimização bayesiana pode efetivamente encontrar boas configurações de parâmetros para quatro diferentes topologias de processamento de dados em fluxo implementadas no Apache *Storm* [1], resultando em ganhos significativos.

O problema da alta multiplicidade de fatores de configuração do *framework Spark* (mais de 180 parâmetros) é enfrentada por Wang, Xu e He (2016) através de uma técnica de ajuste automático que testou diversos algoritmos diferentes de aprendizado de máquina em busca do padrão de ajuste de parâmetros mais eficiente. Os autores afirmam que Árvores de Decisão *C 5.0* demonstraram o melhor equilíbrio entre acurácia e performance no tempo. Como resultado, houve ganho médio de até 36% no tempo de execução da tarefa distribuída sobre a configuração *Spark* padrão. Embora promissora, uma lacuna do trabalho reside no fato de testar apenas configurações de *software*/plataforma; o impacto sobre a variação de *hardware* não é investigado.

Já Munir et al. (2019) propõem uma abordagem baseada em custos em que uma técnica multi-objetivo decide o número de tarefas e o número de máquinas para execução com base nos dados de entrada. Os autores consideram que o *framework* pode sofrer uma sobrecarga muito alta se o número de tarefas paralelas for inadequado para as nuanças do problema. A sobrecarga causada por elevado número de tarefas pode aumentar o tempo de execução total e induzir a um desperdício significativo de recursos de computação devido a atividades de suporte, tais como inicialização, escalonamento, coleta de lixo em memória, dentre outras.

3.1.3 Modelos Preditores de Desempenho

Muitas pesquisas têm produzido modelos matemáticos preditores de tempo, custo de execução ou outras métricas de consumo de recursos em plataformas para *Big Data*. Em Popescu et al. (2012), os autores propõem um modelo de previsão de tempo de execução de tarefas *Hadoop* baseado em aprendizagem de máquina para um conjunto fixo de consultas analíticas e sob a premissa de configuração constante do *cluster*. O modelo foi testado em dados sintéticos de *benchmarking* do *framework TPC-DS* (PILHO, 2014) e em dados reais. Embora promissor,

[1] *Framework* para processamento de grandes volumes de fluxos de dados em tempo real.

uma desvantagem desse estudo reside na configuração fixa de *clusters*, com taxa de acerto menor que 80%, sem apoio de ferramentas estatísticas avançadas.

Uma das primeiras pesquisas a analisar o problema da predição de desempenho em *Spark* é apresentada por Wang e Khan (2015) — um modelo controlado por simulação capaz de prever o desempenho da carga de trabalho com alta precisão. Especificamente, como as tarefas consistem em vários estágios sequenciais, o modelo simula sua execução usando apenas uma fração dos dados de entrada e coleta *logs* de execução (por exemplo, sobrecarga de E/S, consumo de memória, tempo de execução) para prever o desempenho do trabalho para cada estágio individualmente. O modelo, testado com quatro aplicações em dados reais em um *cluster* composto por treze nós mostrou alta precisão para tempo e gasto de memória, mas encontrou alta variação relacionada às demandas de E/S. Todavia, uma lacuna da pesquisa consiste na falta de testes em *clusters* maiores.

Romsaiyud e Premchaiswadi (2013) propõem um modelo de seleção de parâmetros de configuração e previsão de tempo baseado em aprendizagem de máquina. O modelo de predição chegou a reduzir o tempo de execução em até 77% na melhor configuração. Embora a técnica seja apurada e os resultados promissores, o estudo foi conduzido em apenas quatro cenários sobre um *cluster* de configuração de *hardware* fixa de vinte nós.

Por fim, em trabalhos mais recentes, tais como em Baldacci e Golfarelli (2018), os autores propõem um modelo de custo para o *Spark SQL* cobrindo a classe de consultas *Generalized Projection, Selection, Join (GPSJ)* e leva em consideração os custos de rede, E/S e CPU realizados em três *benchmarks* e dois *clusters* de diferentes tamanhos.

3.1.4 Abordagens Estatísticas

O trabalho de Nguyen et al. (2017) é baseado puramente em técnicas estatísticas e neste os autores desenvolvem uma metodologia para extração de métricas de desempenho relevantes para diferentes níveis de execução do *Spark (application, stage, task, system)* [2]. No trabalho, os autores aplicam técnicas de análise estatística simples, tais como média, máximos e mínimos para identificar os principais recursos de execução que mudam significativamente em resposta para alterações nas configurações. O estudo investigou seis aplicações *open source* para *Big Data (Word Count, Tera Sort, KMeans, Matrix Factorization, PageRank* e *Triangle Count*) em um *cluster* de seis nós, 12 núcleos e 32 GB RAM por nó. A pesquisa foi capaz

[2] Conceitos intrínsecos à arquitetura do *Spark*, exceto *system*, que se refere ao sistema operacional.

de responder questões como "(i) que conjunto de configurações afetam o comportamento da execução do Spark?" ou "(ii) por que a alteração da configuração X prejudica o desempenho do aplicativo Spark Y?". O trabalho é importante por demonstrar que é possível raciocinar sobre o impacto das configurações, caracterizando as cargas de trabalho e aplicando técnicas de análise estatística. Todavia, uma limitação do estudo reside na utilização de apenas uma configuração de *hardware* de *cluster*, sensivelmente pequena (apenas seis nós). Ademais, ferramentas estatísticas mais avançadas podem auxiliar o processo experimental bem como potencializar os resultados. Nesse sentido, o trabalho de Venkataraman et al. (2016) apresenta métodos estatísticos baseados em projeto de experimentos com *Latin Hypercube Sampling (LHS)*. Na percepção dos autores, várias tarefas distribuídas têm estrutura previsível em termos de computação e comunicação. A técnica conseguiu minimizar o número e o custo de unidades experimentais e analisar o comportamento de tarefas em pequenas amostras de dados e prever seu desempenho em conjuntos de dados maiores e diferentes tamanhos de *cluster*.

3.1.5 Modelagem com Redes de *Petri*

Tratam-se de modelos que utilizam técnicas outras além das ferramentas tradicionais de estatística básica ou das técnicas de aprendizagem de máquina. Por exemplo, Ardagna et al. (2016) traz modelos de análise de desempenho para estimar os tempos de execução de tarefas em *Hadoop* através de modelos de complexidade e precisão crescentes, que variam de Redes de Filas (*QN*) a Redes de *Petri* Estocásticas bem Formadas (*SWN*), capazes de estimar o desempenho do trabalho em vários cenários de interesse, incluindo também recursos não confiáveis. Os modelos foram avaliados considerando o *benchmark TPC-DS* (PILHO, 2014) em execução no *Amazon EC2* e no centro de supercomputação italiano *CINECA*. Os resultados mostraram precisão média entre 9% e 14%.

Já a proposta apresentada por Gianniti et al. (2017) fornece uma nova contribuição no estudo de modelagem baseada em Redes de Petri Fluidas (*FSPN*) para prever o tempo de execução de aplicativos *Hadoop* e *Spark*. Os modelos também foram validados no CINECA e na plataforma de dados *Microsoft Azure HDInsight*. Os resultados mostraram que a precisão alcançada foi, em média, de 9,5% para *Hadoop* e 10% para *Spark* em relação às medições reais.

3.1.6 Ferramentas de *Benchmarking*

A necessidade por ferramentas de medição de desempenho e consumo de recursos em plataformas *Big Data* motivou o surgimento de um leque de *frameworks* para *benchmarking*, resumidos a seguir.

Poggi et al. (2015) apresentam uma ferramenta de *benchmarking open source* contendo amplo repositório público com detalhes da execução de 42.000 *jobs* em volumes de dados de até 57 Terabytes (TB). A ferramenta oferece meios para extrair conhecimento para otimizar as opções de configuração e implantação em nuvem, selecionando máquinas virtuais e tamanhos de *cluster* mais econômicos. Também apresenta os principais resultados da avaliação de diferentes configurações para o *framework Hadoop* e sistemas operacionais, cobrindo mais de 100 implantações de *hardware*.

Há ferramentas inteiramente dedicadas ao processo de suporte ao *benchmarking* de aplicações voltadas para processamento de *Big Data*, como é o caso do *Intel HiBench* (HUANG et al., 2010; HUANG et al., 2011), que permite a análise e extração detalhada de métricas de execução para aplicações *Hadoop*, *Storm* e *Spark*. Outro importante exemplo consiste na ferramenta *SparkBench* (LI et al., 2015; AGRAWAL et al., 2016), que consiste em um conjunto de testes de desempenho expandido do Spark. A proposta endereça os nuances e requisitos do *Spark* para diferentes aplicações. Muitas outras ferramentas para benchmarking para *Big Data* foram desenvolvidas, tais como: *GridMix, PigMix, YCSB, TPC-DS, Big-Bench, LinkBench, Performance Benchmark e CloudSuite*. Uma análise comparativa aprofundada encontra-se em Han, Xiaoyi e Jiangtao (2014).

3.2 LACUNAS EXPERIMENTAIS EM CIÊNCIA DA COMPUTAÇÃO

Embora muito comuns, as análises experimentais na Ciência da Computação normalmente possuem lacunas quanto à aplicação de robustos métodos estatísticos. São frequentes os experimentos desenvolvidos através de técnicas empíricas, baseadas em tentativa e erro, com premissas frágeis acerca do plano experimental, dos limites envolvidos, do impacto dos fatores e suas interações.

Barr et al. (1995) colocam esta questão de forma impactante ao afirmar que os padrões para testes empíricos na Ciência da Computação são muito menos rigorosos quando comparados a ciências como Física, Engenharia ou Medicina. Demonstrações ou provas de conceito através

do emprego de testes simplórios ou procedimentos *ad hoc* são a norma, em detrimento da construção cuidadosa de projetos experimentais estatisticamente válidos. Ridge (2007) expõe raciocínio similar baseado em levantamento bibliográfico em trabalhos acadêmicos sobre meta-heurísticas computacionais:

> "... embora as análises empíricas sejam muitas vezes amplas e abrangentes, raramente são apoiadas pelo rigor científico que se esperaria em disciplinas mais maduras. (...) Projetos experimentais adequados raramente são usados. As interpretações dos resultados são postas mais na forma de opiniões subjetivas e não análises estatísticas sólidas."

Embora não contemporâneas, as afirmações de Barr et al. (1995) e Ridge (2007) são atuais e encontram eco em trabalhos mais recentes, tal como em Freiesleben, Keim e Grutsch (2020) que reforçam a escassez do emprego de DoE (FISHER; WISHART, 1930) no campo da aprendizagem de máquina:

> Dado o vínculo óbvio entre DoE e aprendizagem de máquina (AM) — ambos preocupam-se com a análise de dados e a aplicação de métodos estatísticos; surpreendentemente existem poucos artigos sobre a interseção dos dois campos.

Diversos autores (BARR et al., 1995; HOOKER, 1995; WINEBERG; CHRISTENSEN, 2004; RIDGE; CURRY, 2007; RIDGE, 2007; FREIESLEBEN; KEIM; GRUTSCH, 2020) defendem o emprego de DoE na pesquisa relacionada a algoritmos; ao menos na mesma medida em que isso já acontece com as ciências relacionadas à natureza. De fato, esta metodologia já é largamente empregada em áreas como a pesquisa farmacêutica (POLITIS et al., 2017; MISHRA et al., 2018; RATHOD et al., 2019); Engenharia (GUECIOUER; YOUCEF; TAREK, 2019; DUTTA; GANDOMI, 2020; KUO; LIU; CHANG, 2020); Agronomia [3] (CASLER, 2015; TEKINDAL et al., 2014) e Física (AMIN; KIANI, 2020), dentre outras.

Há aplicações de DoE em Ciência da Computação, a exemplo dos trabalhos de Bates, Sienz e Toropov (2004), Ridge (2007) e Pais et al. (2014), dedicados ao campo da computação evolucionária. O potencial combinado de DoE e aprendizagem de máquina já foi empregado em trabalhos que envolvem Máquinas de Vetores de Suporte (SVM) (STAELIN, 2003) e redes neurais (PACKIANATHER; DRAKE; ROWLANDS, 2000; BALESTRASSI et al., 2009). Todavia, os esforços científicos na área ainda são escassos, representando apenas 6% das publicações voltadas ao tema, precedidas por Medicina, Engenharia, Bioquímica, Genética, Física e Astronomia (DURAKOVIC, 2017, pg. 424).

[3] As bases para o projeto de experimentos surgiram na Agronomia com as pesquisas de Fisher e Wishart (1930), Fisher (1935) na década de 1930.

A lacuna de emprego das técnicas de DoE na pesquisa relacionada a *Big Data* é ainda maior. Em pesquisa bibliográfica realizada pelo Autor (Seção 2.1) foi encontrado apenas um trabalho (VENKATARAMAN et al., 2016) baseado em técnicas de projeto de experimentos sobre *Big Data*.

3.3 ABORDAGEM DA TESE: PROJETO DE EXPERIMENTOS FATORIAL

O Quadro 3 enumera os trabalhos relacionados para fácil comparação, em ordem temporal, destacando as abordagens de pesquisa, plataformas sob estudo, ocorrência no tempo e referências bibliográficas para consulta.

Quadro 3 – Comparação de trabalhos de análise de desempenho para *Big Data*

Abordagem	**Plataforma**	**Referência**
Algoritmo de escalonamento	*Hadoop*	Zaharia et al. (2008)
Ferramenta de Benchmark	*Hadoop, Spark, Storm*	Huang et al. (2010), Huang et al. (2011)
Ajuste de Parâmetros	Hadoop	Herodotou et al. (2011)
Modelo Preditor	*Hadoop*	Popescu et al. (2012)
Algoritmo de escalonamento	*Hadoop*	LIN et al. (2013)
Ajuste de Parâmetros	*Hadoop*	Romsaiyud e Premchaiswadi (2013)
Algoritmo de escalonamento	*Hadoop*	Arasanal e Rumani (2013)
Ferramenta de Benchmark	*Spark*	Li et al. (2015), Agrawal et al. (2016)
Ferramenta de Benchmark	*Hadoop*	Poggi et al. (2015)
Modelo Preditor	*Spark*	Wang e Khan (2015)
Ajuste de Parâmetros	*Hadoop*	Pospelova (2015)
Ajuste de Parâmetros	*Hadoop*	Zhang, Li e Hildebrand (2015)
Ajuste de Parâmetros	*Storm*	Fischer, Gao e Bernstein (2015)
Modelagem Estatística	*Spark*	Venkataraman et al. (2016)
Ajuste de Parâmetros	*Spark*	Wang, Xu e He (2016)
Redes de Petri	Hadoop	Ardagna et al. (2016)
Modelagem Estatística	*Spark*	Nguyen et al. (2017)
Redes de Petri	*Hadoop, Spark*	Gianniti et al. (2017)
Modelo Preditor	*Spark*	Baldacci e Golfarelli (2018)
Ajuste de Parâmetros	*Spark*	Munir et al. (2019)

Fonte: O autor (2020)

É possível observar que muitos trabalhos se concentram na problemática do ajuste automático ou manual de parâmetros de configuração de *software* de *framework*. Alguns são dedicados à modelagem preditiva de desempenho, em especial de tempo. Parte dos trabalhos encontrados apresentam técnicas específicas de *framework*, carecendo de mergulho profundo

na dinâmica de execução e de implementação das plataformas, realizando ações intrusivas até mesmo no nível de escalonamento de tarefas. Noutros há carência de investigações mais amplas com *clusters* de maior poder computacional. Por último, há raras abordagens que empregam modelagens matemáticas ou estatísticas como procedimento para investigação dos resultados e, principalmente, uma grande lacuna no emprego de planejamento experimental como premissa para prover completude e cobertura aos resultados a serem analisados.

Nesse contexto, esta Tese apresenta um método baseado em projeto de experimentos fatorial randomizado com replicações (FISHER; WISHART, 1930; FISHER, 1935) para triar fatores relevantes e construir modelos de previsão de desempenho de tempo e custo de algoritmos distribuídos. São demonstradas aplicações sobre algoritmos iterativos de aprendizagem de máquina no *framework Spark*. Todavia, a metodologia pode ser adaptada a qualquer *framework* ou categoria de problema sobre *Big Data*, tais como análise de grafos, consultas *SQL-like*, tarefas *Hadoop/MapReduce*, dentre outras. Ainda mais, o pesquisador não precisa conhecer as características internas da estrutura, algoritmo ou propriedades dos dados, pois a abordagem investiga fatores agnósticos de *hardware* presentes em qualquer cenário. A metodologia é capaz de selecionar e ranquear os fatores que influenciam o tempo e o custo de computação distribuída para *Big Data* descrevendo seus efeitos e explicando a contribuição proporcional sobre a variação de desempenho através de robustas técnicas estatísticas.

O método apresentado nesta Tese difere dos trabalhos citados ao empregar DoE (FISHER; WISHART, 1930; FISHER, 1935) sobre técnicas de aprendizagem de máquina em cenários envolvendo dados sintéticos e reais, executadas em *clusters* configurados de acordo com combinações de fatores de *hardware* e volume de entrada de dados.

3.4 SÍNTESE

Esta seção apresentou, de forma não exaustiva, trabalhos relacionados ao desempenho de aplicações distribuídas para processamento de *Big Data*, e pode ser sumarizada como:

- esforços relacionados a algoritmos de escalonamento de tarefas, baseados em localidade de dados, tamanho de *cluster* e propriedades físicas de cada nó;
- técnicas de escolha dos melhores parâmetros de configuração de *software* para prover o melhor desempenho;

- abordagens baseadas em modelos matemáticos, tais como Redes de *Petri* ou estudos baseados em ferramentas estatísticas;
- técnicas que utilizam aprendizagem de máquina para derivar modelos preditores de tempo e custo;
- uma discussão sobre o emprego de técnicas de projeto de experimentos (DoE) na área de Ciência da Computação, em especial em pesquisas voltadas para *Big Data*;
- justificativas do emprego de DoE como ferramenta experimental estatisticamente segura e matematicamente válida para condução, análise e proposição de afirmações sobre o comportamento do sistema face a variação de fatores;
- um contraponto aos trabalhos referentes a otimização que demandam conhecimento sobre infraestrutura, o funcionamento interno do algoritmo ou detalhes sobre os dados a serem analisados.

4 PROJETO DE EXPERIMENTOS

> “ *Standing on the shoulders of giants.* ”

— Bernard of Chartres, *Philosopher, A.D. 12c.*, Troyan (2004)

Esta Tese foi motivada pela necessidade de procedimentos rigorosos para identificação, quantificação e análise de fatores que influenciam o desempenho em plataformas de processamento para *Big Data*. Uma detalhada discussão pode ser revisada na Seção 3.2. A presente seção enumera conceitos sobre procedimentos experimentais e introduz as bases teóricas utilizadas na pesquisa. As definições e técnicas são extratos diretos ou adaptações de referências sobre o tema que pode ser aprofundado em Fisher e Wishart (1930), Fisher (1935), Yates (1936), Ostle et al. (1963), Nelder e Mather (1965), Jain (1990), Amini e Barr (1993), Barr et al. (1995), Montgomery (2005), Ridge (2007), Lawson (2014), Grönmping (2014), Durakovic (2017), Montgomery (2017) e Seltman (2018).

4.1 PREÂMBULO

O equilíbrio entre poder da técnica, generalização, validade, praticidade e custo resulta em estudos com melhores chances de entregar evidências úteis para modificar o estado corrente de conhecimento de um determinado campo científico.

Isto posto, um desafio fundamental da análise estatística consiste em tirar conclusões com segurança a partir de dados. De acordo com Lawson (2014), estes podem ser coletados através de **amostragem**, estudos de **observação** ou **experimentos**. Para o último caso — evidências experimentais colhidas em ambientes controlados; deve-se estabelecer criteriosamente a quantidade e os limites das variáveis, bem como a apropriada organização, condução e coleta de resultados.

O Projeto de Experimentos — do termo original em inglês *Desing of Experiments* (*DoE*) consiste em um processo para estruturar e analisar experimentos, projetado para minimizar o esforço de teste e maximizar as informações adquiridas. Trata-se de uma abordagem que garante que os dados coletados possam ser analisados por métodos estatísticos para alcançar conclusões válidas e objetivas (BARR et al., 1995).

Os conceitos de *DoE* tiveram suas bases construídas por Fisher e Wishart (1930) em estudos nas áreas da Agricultura e Biologia, nos quais foram elaboradas ferramentas esta-

tísticas para organizar experimentos capazes de lidar com a alta variação que torna confusa a compreensão de resultados observáveis e as condições que os provocam. Refinamentos foram posteriormente desenvolvidos por *George E. P. Box, Søren Bisgaard, William G. Hunter*, e *Genichi Taguchi*; sendo atualmente uma das metodologias estatísticas mais utilizadas por engenheiros industriais na otimização do desempenho de processos por meio de configurações experimentais projetadas de maneira inteligente (FREIESLEBEN; KEIM; GRUTSCH, 2020). É utilizado em vários tipos de sistemas, processos e projetos de produto para conduzir estudos científicos nos quais as variáveis de entrada são criteriosamente controladas para investigar seus efeitos na variável de resposta. Pode ser empregado em comparações, triagem de fatores, identificação de funções de transferência [1], ou otimização e projeto robusto de sistemas [2]. Sua utilização vem se expandindo nas indústrias manufatureira e não manufatureira, sendo a ferramenta mais popular em áreas como Medicina, Engenharia, Bioquímica e Física nos últimos vinte anos (DURAKOVIC, 2017).

Foge ao escopo da Tese uma explanação completa sobre *DoE*, limitando-se a proporcionar um entendimento introdutório sobre o tema. No Anexo A - Notas Teóricas Complementares, Seção A.1 - Conceitos Experimentais encontram-se definições para o leitor não familiarizado com os termos básicos. Aprofundamento sobre o assunto pode ser obtido seguindo as referências citadas.

4.2 PROJETOS FATORIAIS

Basicamente, especificar um projeto de experimentos consiste em estruturar um conjunto de testes, definir seus fatores, níveis e variáveis de resposta, estabelecer uma ordem de randomização, o número de replicações e as repetições de mensuração (se necessário). Nesse cenário emergem dois questionamentos imediatos: (i) o número de experimentos e (ii) como estes devem ser organizados.

Jain (1990) aponta tipos de projetos de experimentos, com ênfase naqueles mais frequentemente utilizados: (ii) fatorial completo e (iii) fatorial fracionado. Por sua vez, Lawson (2014, p. 49) apresenta uma taxonomia mais abrangente agrupando os projetos conforme propósitos de pesquisa, tipos de fatores sob estudo e condições de variação destes.

[1] Função matemática que modela a saída de um componente eletrônico ou de sistema de controle para cada entrada possível (LAUGHTON; SAY, 2013).

[2] Redução da variação de um sistema, processo ou produto, através da eliminação dos ruídos (fatores ambientes, fora de controle ou desconhecidos) que as causam (TAGUCHI; CLAUSING et al., 1990).

Na sequência são descritas as técnicas utilizadas nesta Tese: projetos experimentais fatoriais completos e fracionados, com ênfase no projeto randomizado de dois níveis com replicações.

4.2.1 Projeto de Experimentos Fatorial Completo

Utiliza todas as combinações de valores possíveis para os níveis estabelecidos de todos os fatores. O número *n* de experimentos é dado por:

$$n = r \times \prod_{i=1}^{k} \eta_i, \tag{4.1}$$

onde:

- r - número de replicações;
- k - número de fatores;
- η_i - número de níveis do i-ésimo fator.

A vantagem direta dos projetos fatoriais completos reside no seu poder para detectar o efeito isolado de todos os fatores e das interações de alta ordem (dois ou mais fatores atuando como um novo fator gerado). Uma desvantagem explícita reside no seu alto custo. Em alguns casos a dinâmica e o tempo do experimento podem levar o estudo a ser inexequível, de difícil condução ou muito caro. Um bom exemplo prático [3] de tais restrições é a realização de experimentos computacionais em um *cluster* de máquinas distribuídas. Se entre os fatores constarem definições como o número de nós do *cluster* ou propriedades individuais de de seus nós (quantidade de núcleos de processamento, volume de memória, número de discos), um novo *cluster* será necessário para cada unidade experimental.

As técnicas de DoE possuem abordagens para reduzir o número de experimentos minimizando o prejuízo semântico do resultado. Em Jain (1990) são apresentadas três formas de minimizar o custo de um projeto fatorial completo:

- reduzir o número de fatores;
- reduzir o número de níveis por fator;
- usar projetos fracionados.

[3] Exemplo do Autor, vide experimentos descritos nas Seções 7.1 e 8.2 a 8.6 desta Tese.

A redução de fatores nem sempre é possível, pois removê-los pode implicar em transformar fatores importantes em variáveis ocultas que afetam o resultado. Como geralmente não se sabe *a priori* qual o impacto de um fator — esta informação é um dos objetivos de um estudo experimental; torna-se difícil suprimi-lo. No caso da redução do número de níveis por fator é recomendado utilizar dois níveis representativos de limites inferior e superior. Essa estratégia, denominada projeto de experimentos 2^k fatorial (Seção 4.2.3), é particularmente útil para triagem de fatores relevantes. Por último, há a opção de conduzir apenas uma fração do experimento completo (seja este de dois ou mais níveis) diminuindo a capacidade de análise do modelo. Neste caso, tem-se os projetos fatoriais fracionados.

4.2.2 Projeto de Experimentos Fatorial Fracionado

Quando o projeto de experimentos fatorial completo torna-se difícil ou inexequível, é possível utilizar apenas uma fração do mesmo, diminuindo custos e tornando os experimentos possíveis na linha de tempo e na capacidade financeira do projeto. Entretanto, a informação capturada pelos projetos fracionados será menor que aquela obtida através do projeto completo. Não será possível obter os efeitos de todas as interações entre todos os fatores, especialmente aqueles de mais alta ordem. Todavia, se alguma interação for conhecida *a priori* ou puder ser negligenciada a partir de alguma ordem (ex: a partir de três fatores), isto pode não ser um problema e a utilização de frações do conjunto de experimentos se justifica em razão da exequibilidade do estudo (JAIN, 1990).

4.2.3 Projeto de Experimentos 2^k Fatorial Fracionado

Utilizado para investigar efeitos de *k* fatores no caso especial em que há apenas dois níveis para cada fator. O projeto 2^k fatorial pode ser utilizado em sua forma completa ou sua variante fracionada. Em geral, escolhem-se os valores para os níveis na forma de limites inferior e superior ou mínimo e máximo dentro de um domínio possível ou arbitrariamente estabelecido por conhecimento de especialista. Todavia, tais valores devem ser esparsados em intervalos tão extremos quão possível para enfatizar a diferença na resposta (LAWSON, 2014). Esta classe especial de projeto de experimentos é de fácil análise e permite ranquear fatores em ordem de importância, sendo muito útil no início de estudos de desempenho com muitos fatores e muitos níveis possíveis.

Um projeto fatorial fracionado com *k* fatores e 2^k experimentos pode ser reduzido para 2^{k-p} experimentos, onde $p \in \mathbb{I}$, $1 \leq p < k$. Um projeto 2^{k-1} demandaria a metade de experimentos de um projeto 2^k completo. Um projeto 2^{k-2} demandaria $\frac{1}{4}$ do total de experimentos, enquanto 2^{k-3} demandaria $\frac{1}{8}$, e assim por diante. A operação *k-p* possui limites quanto à resolução mínima possível para o projeto, dado o número de fatores. Maiores detalhes em Jain (1990), Grönmping (2014) e Montgomery (2017).

4.2.3.1 Matriz de Yates

É comum que fatores se comportem de maneira unidirecional com incremento ou decremento de efeitos de maneira proporcional direta ou inversa à mudança de seus níveis (JAIN, 1990). Assim, é conveniente expressar os níveis de variação de um fator em um experimento 2^k fatorial através de contrastes. A estruturação de um projeto de experimentos envolve a construção da Matriz de *Yates* (YATES, 1936) com *k* fatores de dois níveis (ou fatores base) e $m - k$ fatores de interação (ou fatores gerados), que representam as diversas combinações de interação entre os fatores base até a sua mais alta ordem. Os níveis dos fatores base são denotados por contrastes $-1/+1$ e as demais colunas da matriz são o resultado do produto dos contrastes, arranjados conforme a 'ordem de *Yates*'. Exemplos ilustrativos de variação de níveis de fatores em contrastes e a estruturação de projetos 2^k fatoriais podem ser encontrados nas Seções 7 - Projeto de Experimentos $2^k r$ Fatorial Completo e 8 - Projeto de Experimentos $2^{k-p} r$ Fatorial Fracionado. No Anexo A - Notas Teóricas Complementares, Seção A.2 - Construção da Matriz de Yates é detalhado o algoritmo para estruturação da matriz experimental.

4.2.3.2 Interações e Confundimento de Efeitos

Segundo Ostle et al. (1963): "... o efeito principal de um fator é a medida da mudança na variável de resposta devido a mudanças no nível do fator e a média sobre todos os níveis de todos os outros fatores". Conforme Ridge (2007), as interações de alta ordem ocorrem quando um fator depende de outro e a mudança combinada nos dois fatores produz um efeito maior (ou menor) do que a soma dos efeitos esperados dos fatores isoladamente. Efeitos de segunda ordem são devidos à interação de dois fatores, de terceira ordem, a três fatores e assim por diante. Havendo *k* fatores, há $\binom{k}{1}$ efeitos principais, $\binom{k}{2}$ efeitos de interação entre dois fatores, $\binom{k}{3}$ efeitos de interação entre três fatores e assim por diante, resultando em um modelo com

$2^k - 1$ efeitos (MONTGOMERY, 2017, p. 253). O número *n* de efeitos para *k* fatores pode ser facilmente estimado por (JAIN, 1990):

$$n = C_{k,i} = \binom{k}{i} = \frac{k!}{i!\,(k-i)!}, \tag{4.2}$$

onde:

- *k* é o número de fatores;
- *i* é o grau de interação, $1 \leq i \leq n$.

Para compreensão, o Quadro 4 quantifica todas as interações de um hipotético estudo 2^k fatorial completo ($k = 6$ fatores, dois níveis por fator).

Quadro 4 – Número de efeitos de interação para um estudo 2^6 fatorial completo

Tipo	Cálculo	Quantidade
Principais	$\binom{6}{1} = \frac{6!}{1!\,5!}$	6
Segunda ordem	$\binom{6}{2} = \frac{6!}{2!\,4!}$	15
Terceira ordem	$\binom{6}{3} = \frac{6!}{3!\,3!}$	20
Quarta ordem	$\binom{6}{4} = \frac{6!}{4!\,2!}$	15
Quinta ordem	$\binom{6}{5} = \frac{6!}{5!\,1!}$	6
Sexta ordem	$\binom{6}{6} = \frac{6!}{6!\,0!}$	1
Número total de efeitos		63

Fonte: O autor (2020), adaptado de Montgomery (2017, p. 253)

Para propósitos de seleção de fatores de maior impacto é comum que o pesquisador esteja interessado nos efeitos principais ou interações até a segunda ordem (RIDGE, 2007). De fato, no exemplo do Quandro 4, analisar vinte interações de terceira ordem ou quinze interações de quarta ordem dentre 63 efeitos se torna uma tarefa impossível, quando não, irrelevante para propósitos práticos.

Isto posto, experimentos fracionados possuem uma desvantagem direta em relação à análise de efeitos. De acordo com Jain (1990):

> "Um problema com projetos fatoriais fracionados é que alguns efeitos não podem ser determinados. Apenas a influência combinada de dois ou mais efeitos pode ser computada. Esse problema é conhecido como **confundimento** e os efeitos cuja influência não pode ser separada são considerados **confundidos**.

Apesar das limitações, fracionar projetos fatoriais viabiliza a investigação de muitos fatores, mantendo propriedades estatísticas importantes que preservam sua análise, ao preço de negligenciar interações de ordens específicas. Isto implica que os efeitos não podem ser estimados separadamente e uma análise sobre o efeito de um hipotético fator principal A pode ser confundido com o efeito da interação de dois hipotéticos fatores B:C, ou ainda, de três fatores como B:C:D, no caso de interações de mais alta ordem, e assim por diante. Detalhamento aprofundado pode ser conferido em Jain (1990, p. 325-329), Grönmping (2014) e Montgomery (2017, p. 304-318).

A forma como os efeitos são confundidos resulta em sua Resolução, a saber:

Quadro 5 – Resolução 2^k fatorial até 20 fatores

		Número de unidades experimentais										
		4	8	16	32	64	128	256	512	1024	2048	4096
Número de fatores	2	2^{2}										
	3	2^{3-1}_{III}	2^{3}									
	4		2^{4-1}_{IV}	2^{4}								
	5		2^{5-2}_{III}	2^{5-1}_{V}	2^{5}							
	6		2^{6-3}_{III}	2^{6-2}_{IV}	2^{6-1}_{VI}	2^{6}						
	7		2^{7-4}_{III}	2^{7-3}_{IV}	2^{7-2}_{IV}	2^{7-1}_{VII}	2^{7}					
	8			2^{8-4}_{IV}	2^{8-3}_{IV}	2^{8-2}_{V}	2^{8-1}_{VIII}	2^{8}				
	9			2^{9-5}_{III}	2^{9-4}_{IV}	2^{9-3}_{IV}	2^{9-2}_{VI}	2^{9-1}_{IX}	2^{9}			
	10			2^{10-6}_{III}	2^{10-5}_{IV}	2^{10-4}_{IV}	2^{10-3}_{V}	2^{10-2}_{VI}	2^{10-1}_{X}	2^{10}		
	11			2^{11-7}_{III}	2^{11-6}_{IV}	2^{11-5}_{IV}	2^{11-4}_{V}	2^{11-3}_{VI}	2^{11-2}_{VII}	2^{11-1}_{XI}	2^{11}	
	12			2^{12-8}_{III}	2^{12-7}_{IV}	2^{12-6}_{IV}	2^{12-5}_{IV}	2^{12-4}_{VI}	2^{12-3}_{VI}	2^{12-2}_{VIII}	2^{12-1}_{XII}	2^{12}
	13			2^{13-9}_{III}	2^{13-8}_{IV}	2^{13-7}_{IV}	2^{13-6}_{IV}	2^{13-5}_{V}	2^{13-4}_{VI}	2^{13-3}_{VII}	2^{13-2}_{VIII}	2^{13-1}_{XIII}
	14			2^{14-10}_{III}	2^{14-9}_{IV}	2^{14-8}_{IV}	2^{14-7}_{IV}	2^{14-6}_{V}	2^{14-5}_{VI}	2^{14-4}_{VII}	2^{14-3}_{VIII}	2^{14-2}_{IX}
	15			2^{15-11}_{III}	2^{15-10}_{IV}	2^{15-9}_{IV}	2^{15-8}_{IV}	2^{15-7}_{V}	2^{15-6}_{VI}	2^{15-5}_{VII}	2^{15-4}_{VIII}	2^{15-3}_{VIII}
	16				2^{16-11}_{IV}	2^{16-10}_{IV}	2^{16-9}_{IV}	2^{16-8}_{V}	2^{16-7}_{VI}	2^{16-6}_{VI}	2^{16-5}_{VIII}	2^{16-4}_{VIII}
	17				2^{17-12}_{III}	2^{17-11}_{IV}	2^{17-10}_{IV}	2^{17-9}_{V}	2^{17-8}_{VI}	2^{17-7}_{VI}	2^{17-6}_{VII}	2^{17-5}_{VIII}
	18				2^{18-13}_{III}	2^{18-12}_{IV}	2^{18-11}_{IV}	2^{18-10}_{IV}	2^{18-9}_{VI}	2^{18-8}_{VI}	2^{18-7}_{VII}	2^{18-6}_{VIII}
	19				2^{19-14}_{III}	2^{19-13}_{IV}	2^{19-12}_{IV}	2^{19-11}_{IV}	2^{19-10}_{V}	2^{19-9}_{VI}	2^{19-8}_{VII}	2^{19-7}_{VIII}
	20				2^{20-15}_{III}	2^{20-14}_{IV}	2^{20-13}_{IV}	2^{20-12}_{IV}	2^{20-11}_{V}	2^{20-10}_{VI}	2^{20-9}_{VII}	2^{20-8}_{VIII}

Fonte: O autor (2020), adaptado de Ridge (2007) e Grönmping (2014)

- Resolução III: nenhum efeito principal está associado a qualquer outro efeito principal, mas são confundidos com interações de segunda ordem;

- Resolução IV : nenhum efeito principal está associado a qualquer outro efeito principal ou interação de segunda ordem, mas algumas interações de segunda ordem são confundidas com outras interações de segunda ordem e os efeitos principais são confundidos com interações de terceira ordem;
- Resolução V : nenhum efeito principal ou interação de segunda ordem estão associados, mas estas últimas são confundidas com interações de terceira ordem enquanto os efeitos principais são confundidos com interações de quarta ordem;
- Completo : todos os efeitos são estimados separadamente, sem confundimento.

Usualmente opta-se pela maior resolução possível — ressalvadas as limitações de tempo total de execução dos experimentos e seu custo. Por exemplo, é preferível escolher um projeto em que os efeitos principais sejam confundidos com interações de terceira ordem (Resolução IV) em detrimento de um projeto em que os efeitos principais estejam confundidos com interações de segunda ordem (Resolução III).

O Quadro 5 mostra as resoluções de projeto experimental para até 20 fatores. Pode-se observar o resultado sobre o número de experimentos ao se reduzir o projeto para uma versão fracionada. O termo $2^{\text{k-p}}_{\text{res}}$ deve ser lido como um projeto fatorial fracionado de *k* fatores de dois níveis com resolução *res* reduzido para $\frac{1}{2^p}$ do total de experimentos; enquanto 2^k representa um projeto fatorial completo. Por exemplo, um estudo com 6 fatores e resolução IV é representado por 2^{6-2}_{IV} e demandará $2^4 = 16$ experimentos, em detrimento de sua versão completa ($2^6 = 64$ experimentos). Uma introdução didática sobre resolução de experimentos pode ser lida em Grönmping (2014). Abordagens detalhadas podem ser encontradas em Jain (1990, Cap. 19) e Montgomery (2005).

Enfim, um projeto randomizado $2^{\text{k-p}}_{\text{res}}r$ fatorial fracionado resolução *res* com replicações consiste em um projeto 2^{k} convencional reduzido para $\frac{1}{2^p}$ experimentos, conduzidos em ordem randômica; onde cada unidade experimental é replicada *r* vezes[4]. Essa organização permite que os erros sejam distribuídos de forma aleatória, impedindo viés no resultado, atendendo importantes premissas para análise estatística através de modelos lineares (Seção 5.3), diminuindo a dificuldade operacional do estudo enquanto mantém a capacidade de captura de efeitos de acordo com a sua resolução.

[4] Execuções independentes completas sob as mesmas condições.

4.3 SÍNTESE

Esta seção abordou as bases estatísticas a serem aplicadas na metodologia da pesquisa e apresentou, em síntese:

- breve levantamento sobre o surgimento das técnicas de Projeto de Experimentos (*DoE*);
- os tipos de projeto experimental e como planejar, organizar e conduzir experimentos maximizando a informação a ser obtida, enquanto se minimiza o custo temporal ou financeiro de execução do estudo;
- conceitos sobre interações, confundimento de fatores, fracionamento de projetos e suas resoluções;
- referências de leitura sobre *DoE* para o leitor interessado.

5 REGRESSÃO LINEAR

> *All models are wrong, but some are useful.*
>
> — George Edward Pelham Box, *Statistician*, Box (1979)

Esta seção detalha a análise de projetos de experimentos fatoriais através de regressão linear multivariada. Também são apresentados os procedimentos para verificação das premissas de independência, homogeneidade de variância e normalidade de resíduos; transformações de escala, interpretação de efeitos e emprego de modelos reduzidos.

5.1 PREÂMBULO

Conforme Seção 4.2.3, projetos de experimentos fatoriais completos capturam efeitos principais e interações até sua mais alta ordem, enquanto os fracionados o fazem de acordo com sua resolução. Ambos possibilitam a análise da variação de fatores na variável de resposta através do emprego de modelos de regressão linear.

Diverso de outras abordagens de modelagem — que incluem árvores de decisão, redes neurais e máquinas de vetores de suporte; a regressão linear permite a derivação de uma fórmula explicativa que auxilia a compreensão dos efeitos dos fatores sobre a resposta a partir da análise de seus coeficientes parciais de regressão e da significância estatística. Também, não se fazem necessárias informações internas sobre o sistema, sendo possível aplicar a abordagem *black-box* [1] nos resultados de *benchmark* obtidos (SIEGMUND et al., 2015). Por fim, modelos lineares ainda facilitam a incorporação do conhecimento do domínio sobre a influência de fatores, o que pode ser eficaz para evitar *overfitting* ou *underfitting* (DOMINGOS, 2012).

Os conceitos doravante apresentados são extratos diretos ou adaptações de Cook (1977), Box e Cox (1964), Jain (1990), Miller (1993), Moessner (2007), Montgomery, Peck e Vining (2012), Kuhn e Johnson (2013), Siegmund et al. (2015), Montgomery (2017) e Pek, Wong e Wong (2017).

[1] Um modelo *black-box* de um sistema é aquele que não utiliza nenhum conhecimento prévio específico do caráter ou da física dos relacionamentos envolvidos (LJUNG, 2001).

5.2 ANÁLISE DE REGRESSÃO

Técnica estatística para investigar e modelar a relação entre variáveis. As aplicações de regressão são numerosas e ocorrem em quase todos os campos, incluindo engenharia, ciências físicas e químicas, economia, administração, ciências biológicas e da vida e ciências sociais; sendo potencialmente a técnica estatística mais amplamente empregada em pesquisas (MONTGOMERY; PECK; VINING, 2012).

5.2.1 Regressão Linear Simples

Quando uma variável dependente *y* pode ser expressa como uma função linear de uma variável independente *x* diz-se que há uma relação do tipo:

$$y = \beta_0 + \beta_1 x + \epsilon, \tag{5.1}$$

onde:

- y é a variável dependente, ou resposta;
- x é a variável independente, ou preditora;
- β_0 é o intercepto da reta;
- β_1 é o coeficiente de regressão que mede o efeito da variável *x* sobre *y*;
- ϵ denota o erro aleatório.

A Equação 5.1 é conhecida por **modelo de regressão linear simples**, onde apenas um fator afeta a variável resposta.

5.2.2 Regressão Linear Multivariada

Todavia, é muito comum que problemas reais precisem ser modelados em relação a múltiplos fatores. Nesse caso, a função amplamente usada é denominada **modelo de regressão linear multivariada**, onde cada variável independente é um regressor que afeta a variável de resposta em uma determinada importância a ser mensurada. A relevância objetiva de uma análise de regressão linear, seja simples ou multivariada, reside na estimativa de parâmetros

desconhecidos do modelo, os coeficientes ou efeitos. Dessa forma, modelos de regressão linear são geralmente utilizados como modelos empíricos ou funções de aproximação.

A forma geral para a regressão linear multivariada é dada por:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \epsilon, \tag{5.2}$$

onde:

- y é a variável resposta;
- $x_j, j = 0, 1, ..., k$ são as variáveis independentes ou preditoras;
- β_0 é o intercepto da reta;
- $\beta_j, j = 0, 1, ..., k$ são os coeficientes parciais de regressão;
- ϵ denota o erro aleatório.

Para a regressão linear múltipla, o coeficiente β_0 é o intercepto do plano. Se o domínio de dados incluir valores nulos para todos os preditores, β_0 será a média de *y* quando os valores dos regressores for zero. Do contrário, não há interpretação física. De maneira similar, o coeficiente β_1 mede a variação esperada em *y* por unidade de variação de x_1, se mantidos fixos os valores de entrada para os demais preditores $x_2, ..., x_k$. Raciocínio análogo pode ser aplicado para os demais coeficientes $\beta_2, ..., \beta_k$. Ou seja, o modelo descreve um hiperplano no espaço *k*-dimensional das variáveis x_j no qual o coeficiente β_j representa a variação esperada na resposta *y* por unidade de mudança em x_j, se mantidos fixas todas as demais variáveis preditoras x_i $(i \neq j)$; e por esta razão são chamados **coeficientes parciais de regressão**.

Duas importantes considerações sobre a aplicação de regressões lineares no contexto de projetos fatoriais devem ser registradas. Primeiramente, uma vez que o coeficiente parcial de regressão β_j mede os efeitos da mudança de uma unidade do fator x_j sobre a média da variável dependente *y*, e o fator varia em duas unidades no plano experimental de –1 para +1, o efeito de um fator x_j equivale a duas vezes o valor do coeficiente de regressão β_j (LAWSON, 2014). Em segundo lugar, como o plano experimental é organizado de acordo com uma matriz ortogonal (Seção 4.2.3.1), o erro padrão estimado para cada coeficiente de regressão será idêntico para todos.

5.2.3 Modelos Lineares com Interações

Segundo Siegmund et al. (2015), a regressão linear pode ser utilizada para estimar coeficientes para termos mais complexos, isto é, para aprender funções não lineares. Montgomery, Peck e Vining (2012) comentam ainda que o termo *linear* refere-se aos parâmetros $\beta_0, \beta_1, ..., \beta_k$, e não à existência de uma função linear entre os fatores e a resposta (y), pois há casos em que mesmo uma relação não linear pode ser tratada através de modelos lineares. Dessa forma, **efeitos de interação** também podem ser analisados através de regressão linear multivariada, cuja forma geral para três hipotéticos fatores x_1, x_2, e x_3 é dada pela Equação 5.3:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_{12} x_1 x_2 + \beta_{13} x_1 x_3 + \beta_{23} x_2 x_3 + \beta_{123} x_1 x_2 x_3 + \epsilon, \quad (5.3)$$

onde:

- $x_1 x_2$, $x_1 x_3$ e $x_2 x_3$ são as interações de segunda ordem;
- $x_1 x_2 x_3$ é uma interação de terceira ordem;
- β_{12}, β_{13}, β_{23} e β_{123} são os coeficientes parciais de regressão em relação às interações;
- os demais parâmetros se explicam tal qual a Equação 5.2.

Quando da ocorrência de interações no modelo, é muito importante que fatores principais sejam analisados levando em consideração possíveis interferências. Já que uma interação entre fatores faz com que um fator falhe em produzir o mesmo efeito sobre a resposta face a mudanças nos níveis de outro fator (MONTGOMERY, 2017).

5.3 PREMISSAS PARA UTILIZAÇÃO DE MODELOS LINEARES

O emprego de modelos lineares pressupõe que $\epsilon \approx NIID(0, \sigma^2)$. Em outros termos, os resíduos devem ser **N**ormais, **I**ndependentes e **I**denticamente **D**istribuídos **(NIID)** com média zero e variância σ^2, o que implica em variáveis randômicas não relacionadas, aproximadas de uma distribuição Normal, exibindo independência em todos os níveis para os fatores. O desenvolvimento de modelos não aderentes a estas premissas pode levar a conclusões diferentes

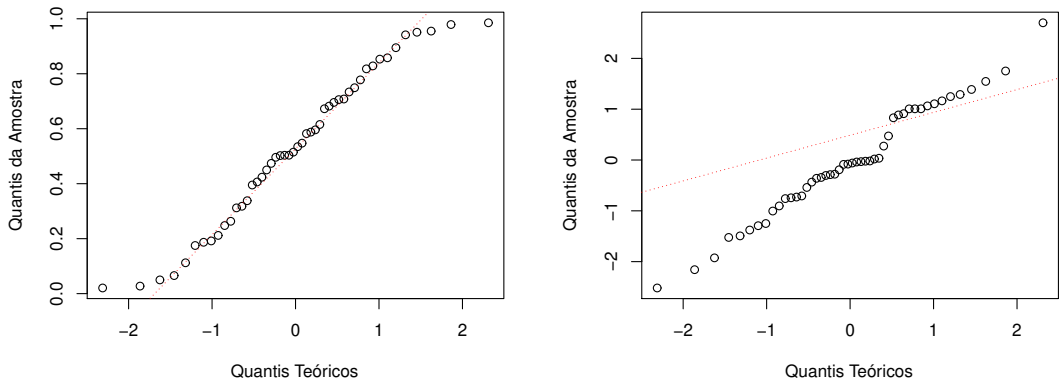
da realidade. Portanto, deve-se sempre conduzir testes de verificação. Um método elementar consiste na análise visual dos gráficos do modelo ajustado em relação aos resíduos (KUHN; JOHNSON, 2013), detalhados na sequência.

5.3.1 Gráfico Quantil-Quantil de Normalidade dos Resíduos

Embora não se possa exigir que experimentos do mundo real sigam rigorosamente uma distribuição Normal, espera-se que se aproximem desta. Do contrário, o ajuste dos valores no modelo pode ser prejudicado. Uma técnica elementar para verificação de normalidade consiste na análise do gráfico quantil-quantil que compara os quantis teóricos de uma distribuição Normal e os quantis dos resíduos da amostra em estudo.

A Figura 2 exibe gráficos quantil-quantil para dois cenários hipotéticos. Na Figura 2a os quantis seguem a linha tracejada que indica uma distribuição Normal teórica. Já na Figura 2b há um desvio muito acentuado sugerindo ausência de normalidade.

Figura 2 – Gráficos quantil-quantil de distribuição dos resíduos



(a) Distribuição Normal de resíduos. (b) Resíduos não aderentes à normalidade.

Fonte: O autor (2020)

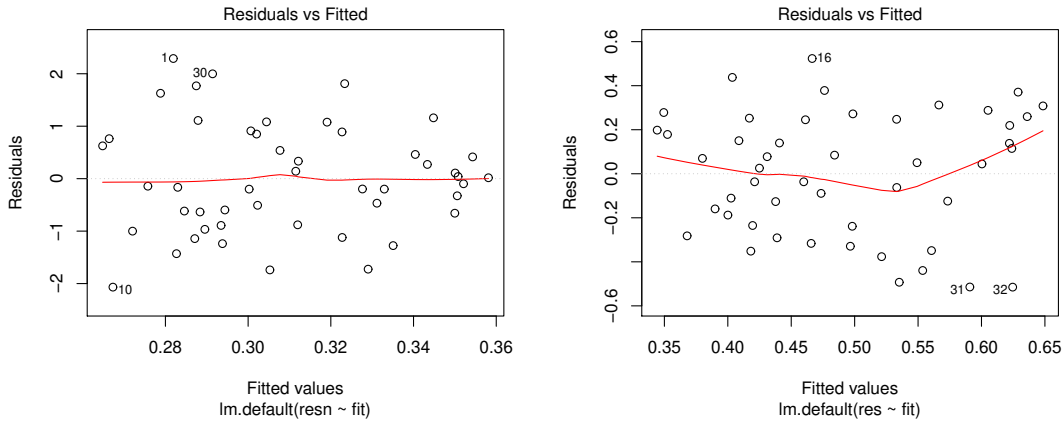
Complementar à análise gráfica, quando o teste de Shapiro e Wilk (1965) resulta em *p-value* > 0.05 admite-se que a distribuição dos resíduos não é significativamente diferente da distribuição Normal. Em outras palavras, pode-se assumir a normalidade dos resíduos.

5.3.2 Gráfico de Resíduos *versus* Valores Ajustados

Os resíduos devem se apresentar dispersos no gráfico em uma faixa horizontal do modo mais aleatório possível, sem padrões notórios. A existência de padrões sinaliza que a variância dos erros é uma função direta ou inversa da variável de resposta, ou ainda que não há linearidade, evidenciando que outras variáveis precisam fazer parte do modelo.

A Figura 3 mostra gráficos para dois modelos hipotéticos.

Figura 3 – Gráficos de resíduos *vs.* valores ajustados do modelo



(a) Homogeneidade de variância. (b) Heterogeneidade de variância.

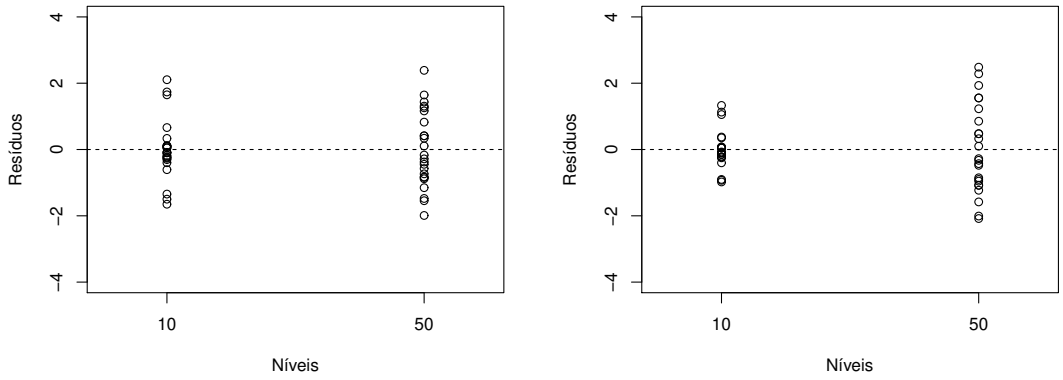
Fonte: O autor (2020)

Na Figura 3a a dispersão dos resíduos ao longo de uma faixa horizontal demonstra homogeneidade de variância. Já a Figura 3b exibe distorção, sugerindo que os resíduos variam de forma heterogênea.

5.3.3 Gráfico de Resíduos *versus* Níveis de Fatores

Espera-se que os resíduos para cada variável dependente se apresentem igualmente esparsados ao longo da abscissa do gráfico em uma faixa horizontal. Em especial, quando aplicados a projetos 2^k fatoriais (Seção 4.2.3), os resíduos devem formar duas linhas ortogonais à abscissa com variação homogênea sobre cada um dos níveis do fator em estudo. Padrões diferentes deste, tais como curvas, túneis, cones ou outras distorções visuais podem indicar ausência de homogeneidade.

Figura 4 – Gráficos de resíduos *vs.* fatores e níveis



(a) Homogeneidade de variância. (b) Variação divergente entre níveis.

Fonte: O autor (2020)

A Figura 4a exibe a variância de resíduos de forma homogênea para os dois níveis do fator em observação. Já na Figura 4b observa-se a formação de um cone ascendente entre os níveis, fato que sugere a violação das premissas para emprego de modelos lineares.

5.3.4 Gráfico de Resíduos em Sequência Temporal

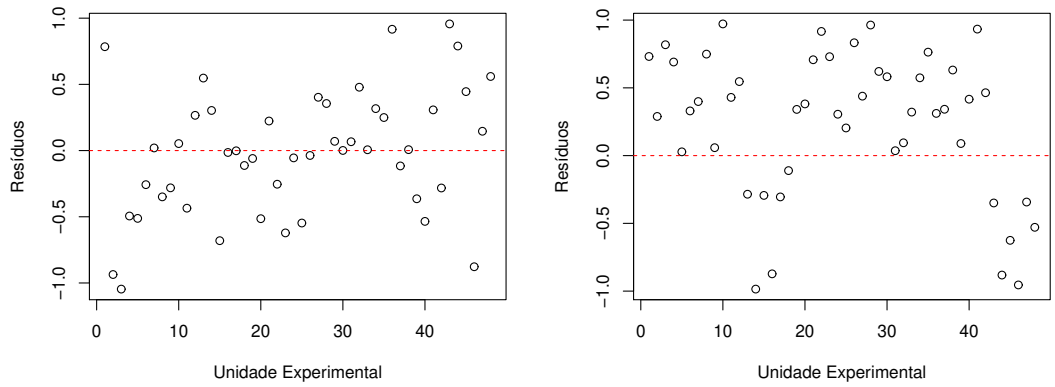
Se a relação entre os dados e sua ocorrência no tempo é conhecida, os resíduos para as unidades experimentais devem flutuar aleatoriamente na sequência temporal, evidenciando independência dos erros.

Na Figura 5a os resíduos se encontram dispersos de maneira uniforme ao longo de toda a faixa horizontal tracejada. Essa dispersão indica que os erros medidos no modelo ocorrem devido a processos aleatórios ao longo do tempo e não dependem de nenhum fator desconhecido. Já na Figura 5b, percebe-se uma maior dispersão de resíduos no quadrante de valores positivos, evidenciando problemas com o modelo.

5.3.5 Gráfico de Pontos Discrepantes

Trata-se de uma observação substancialmente diferente da maioria dos dados e que podem provocar severos efeitos no modelo de regressão. Tais pontos indicam a necessidade de revisão das condições em que ocorreu cada experimento. Pontos discrepantes costumam ser

Figura 5 – Gráficos de resíduos *vs.* unidades experimentais



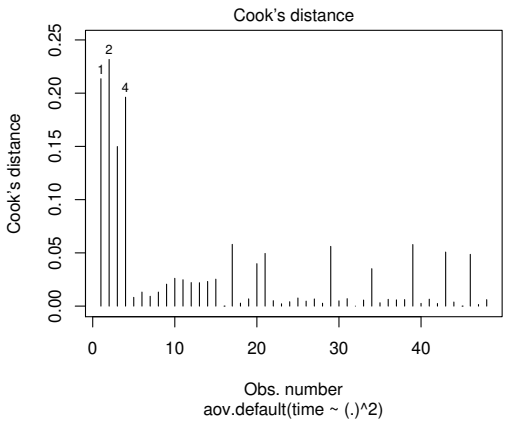
(a) Indícios de independência de erros. (b) Erros não aleatórios.

Fonte: O autor (2020)

evidenciados nos gráficos quantil-quantil de normalidade dos resíduos. Todavia, o cálculo de distância de Cook (1977) estabelece um valor que auxilia a mensurar o impacto do ponto em questão. Quão maior a distância de *Cook*, maior a discrepância entre os pontos e o conjunto total e sua consequente influência na distorção do modelo.

A Figura 6 exibe um exemplo hipotético com três pontos discrepantes claramente identificados.

Figura 6 – Gráfico de distância de *Cook* com pontos discrepantes



Fonte: O autor (2020)

A discrepância pode se dar por erros experimentais, situações anômalas ou ser indicativo de que o universo em estudo é afetado por variáveis que não estão inclusas no modelo. Quando o ponto discrepante for resultado de erros experimentais, é comum que o experimento seja

repetido. Do contrário, técnicas de correção de discrepância podem ser empregadas (MILLER, 1993; MOESSNER, 2007). Ou ainda, o ponto discrepante pode ser mantido, desde que a análise do modelo linear com ou sem sua presença não produza significativas diferenças estatísticas.

5.4 TRANSFORMAÇÕES DE DADOS

Conforme Montgomery, Peck e Vining (2012), é comum que violações nas premissas de emprego de modelos lineares deixem de existir quando os dados são submetidos a transformações de escala. Chambers et al. (1983) elencam as razões para tais transformações:

- aproximar os dados de uma distribuição com boas propriedades estatísticas;
- espalhar os dados de maneira mais uniforme;
- tornar as distribuições de dados mais simétricas;
- tornar as relações entre variáveis mais lineares; e
- tornar a variância dos dados mais constante.

A transformação de escala pode ser conduzida sobre a variável de resposta, sobre as preditoras, ou ambas. Embora haja cenários em que a decisão sobre a escala da operação pode ser tomada com base em conhecimento de domínio do especialista, há métodos formais que podem ser aplicados. Uma discussão aprofundada acerca das escalas possíveis foge ao escopo desta Tese. Isto posto, será dado foco aos métodos empregados na pesquisa, em particular, as transformações de potência propostas por Box e Cox (1964). Nestas, a falta de normalidade ou a heterogeneidade de variância dos resíduos pode ser corrigida ao se aplicar uma transformação y^{λ}, onde λ é um parâmetro estimado a partir dos dados pelo método da máxima verossimilhança. Velleman e Hoaglin (1981) sugerem que as transformações de potência de Box e Cox (1964) podem ser aproximadas de acordo com o valor de λ, conforme sumário do Quadro 6, a seguir:

É importante ressaltar que as transformações não asseguram aderência às premissas NIID, sendo necessário realizar nova análise dos resíduos do modelo ajustado sobre os dados transformados.

Quadro 6 – Transformações *Box-Cox* por aproximação de λ

$\approx \lambda$	–	Operação	Transformação
3	y^3	cúbica	$y^* = y^3$
2	y^2	quadrática	$y^* = y^2$
1	y^1	nenhuma	$y^* = y$
0.5	$y^{\frac{1}{2}}$	raiz quadrada	$y^* = \sqrt{y}$
0.33	$y^{\frac{1}{3}}$	raiz cúbica	$y^* = \sqrt[3]{y}$
0	$\log y$	logarítmica	$y^* = \log y$
-0.5	$-\frac{1}{\sqrt{y}}$	raiz quadrada recíproca	$y^* = -\frac{1}{\sqrt{y}}$
-1	$-\frac{1}{y}$	recíproca	$y^* = -\frac{1}{y}$
-2	$-\frac{1}{y^2}$	recíproca quadrática	$y^* = -\frac{1}{y^2}$

Fonte: Velleman e Hoaglin (1981)

5.5 INTERPRETAÇÃO DE MODELOS LINEARES

De posse do modelo linear, torna-se necessário interpretar o resultado, identificar e ordenar os fatores mais relevantes e estabelecer um modelo reduzido prático para utilização. Esta seção apresenta uma reflexão sobre a inferência estatística dos efeitos por fator em face da transformação de escala da resposta e um procedimento para estabelecer os modelos reduzidos a partir de projetos 2^k fatoriais.

5.5.1 Efeitos em Escala Transformada

Quando os dados estão transformados (Seção 5.4), os coeficientes de regressão também o estarão, modificando a interpretação do efeito operacionalizado pela variável original (PEK; WONG; WONG, 2017). Esta seção discute o impacto das transformações logarítmica e raiz quadrada sobre a interpretação dos efeitos dos fatores e estende o raciocínio para as demais transformações. O leitor interessado pode conferir maiores detalhes em Gelman e Hill (2006), Cohen et al. (2013) e Pek, Wong e Wong (2017).

Relativo à transformação **logarítmica** sobre a resposta, tome-se a Equação 5.2 e submeta-

se ambos os lados à constante *e*, base do logaritmo natural:

$$\begin{aligned} Y_i &= e^{\beta_0+\beta_1 X_{1i}+\ldots+\beta_K X_{Ki}+\epsilon_i} \\ &= \mathbf{B_0'}\,\mathbf{B_1}'^{X_{1i}}\,\mathbf{B_K}'^{X_{Ki}}\epsilon_{\mathbf{i}}' \,, \end{aligned} \tag{5.4}$$

onde: $\mathbf{B_0'} = e^{\beta_0}, \mathbf{B_K'} = e^{\beta_K}$ e $\epsilon_{\mathbf{i}}' = e^{\epsilon_i}$.

A Equação 5.4 mostra que não há dispersão dos coeficientes sobre outros fatores, fato que auxilia a interpretação do modelo. Observe-se que as variáveis independentes entram no modelo multiplicativamente, e não aditivamente, portanto, a mudança de uma unidade no coeficiente β_j equivale a uma mudança percentual sobre o valor de *y*.

Já a transformação para escala **raiz quadrada** dificulta a interpretação dos efeitos. Para compreensão, tome-se o modelo linear da Equação 5.1. Aplicando a transformação e submetendo ambos os lados à potência quadrática, tem-se:

$$\begin{aligned} Y_i &= (\beta_0 + \beta_1 X_i + \epsilon_i)^2 \\ &= \beta_0^2 + 2\beta_0\beta_1 X_i + \beta_1^2 X_i^2 + 2\beta_0\epsilon_i + 2\beta_1 X_i\epsilon_i + \epsilon_i^2 \\ &= \mathbf{B_0'} + \mathbf{B_1'} X_i + \mathbf{B_2'} X_i^2 + \epsilon_{\mathbf{i}}' \,, \end{aligned} \tag{5.5}$$

onde: $\mathbf{B_0'} = \beta_0^2, \mathbf{B_1'} = 2\beta_0\beta_1, \mathbf{B_2'} = \beta_1^2$ e $\epsilon_{\mathbf{i}}' = 2\beta_0\epsilon_i + 2\beta_1 X_i\epsilon_i + \epsilon_i^2$.

Percebe-se que a transformação dispersa o efeito dos coeficientes sobre as variáveis independentes, dificultando a interpretação do efeito da variação isolada de um fator. O modelo continua válido para predição, mantendo-se a proporção de variação explicada, coeficientes de determinação e demais métricas de qualidade. Exceto a transformação logarítmica, aplica-se raciocínio similar à série de transformações do Quadro 6.

5.5.2 Equação Reduzida

Uma das vantagens da utilização de projetos de experimentos no contexto aplicado nesta Tese, além da triagem de fatores relevantes, se dá na sua capacidade para estabelecer modelos reduzidos de maior usabilidade prática. Consoante Lawson (2014), o princípio da ortogonalidade do projeto permite que uma equação reduzida seja escrita a partir dos resultados da

regressão pela simples eliminação dos termos **estatisticamente não significativos** e escalando um dado fator X pela fórmula:

$$\overline{x} = \frac{X - X_{pc}}{X_{md}}\,, \tag{5.6}$$

onde:

- $X_{pc} = \frac{X_{(-)} + X_{(+)}}{2}$, ponto central entre os níveis $X_{(-)}$ e $X_{(+)}$;
- $X_{md} = \frac{X_{(-)} - X_{(+)}}{2}$, metade da distância entre os níveis $X_{(-)}$ e $X_{(+)}$.

Os projetos experimentais descritos nas Seções 7 e 8 resultam em modelos preditores lineares na forma de equação reduzida a partir da regra descrita na Equação 5.6, acima.

5.6 SÍNTESE

Esta seção apresentou o referencial teórico sobre a regressão linear, utilizada como ferramenta para a análise de resultados de projetos de experimentos estruturados conforme a Seção 4. Em suma, podem-se enumerar os principais pontos:

- bases matemáticas da regressão linear simples e multivariada;
- premissas estatísticas para emprego de modelos lineares;
- interferência das interações entre fatores nos modelos lineares multivariados;
- criação de modelo preditor reduzido;
- referências para leitura e aprofundamento dos conceitos de regressão linear.

6 METODOLOGIA

> *Research is what I'm doing when I don't know what I'm doing.*
>
> — Wernher von Braun, *Rocket Scientist*, Brown e Rodgers (2002)

A Seção 4 apresentou uma visão geral sobre experimentação em ambientes controlados e as bases conceituais para estruturação de projetos de experimentos fatoriais. A Seção 5 abordou os princípios da análise de resultados experimentais através de técnicas de regressão linear. Baseado nestes fundamentos, esta seção introduz uma metodologia para investigação do desempenho de tarefas distribuídas em *clusters* de processamento de *Big Data*. A Seção 6.1 apresenta os recursos técnicos, ferramentas e linguagens empregadas, enquanto o procedimento metodológico é descrito na Seção 6.2.

6.1 RECURSOS E FERRAMENTAS

A pesquisa utilizou o arcabouço técnico da plataforma *Spark*, os serviços de provedor em nuvem *Google Cloud* [1], as linguagens de programação *Python* e *Scala* para processamento de dados e a linguagem R para análise estatística dos resultados.

6.1.1 Plataforma de Processamento

A opção pelo *Apache Spark* se deu pela capacidade de tolerância a falhas, interface de programação bem desenvolvida para manipulação de dados e a existência de bibliotecas para aprendizagem de máquina. Detalhes conceituais constam na Seção 2.2.2.3. Os experimentos foram executados sobre *Spark 2.2.3* — projeto experimental descrito na Seção 7, e *Spark 2.4.5* — projetos experimentais descritos na Seção 8.

6.1.2 Provedor em Nuvem e *Clusters* Efêmeros

Os experimentos foram executados em *clusters* efêmeros [2] no serviço em nuvem *Google Data Proc*. Cada *cluster* em si constituiu uma unidade experimental distinta do projeto de

[1] *Google Cloud Data Proc* <https://console.cloud.google.com/dataproc/>.
[2] Criados exclusivamente para processamento de tarefas e destruídos após conclusão.

experimentos fatorial cujo desempenho se desejava medir. Cada *cluster* foi organizado com um nó mestre (*namenode*) e $l_N - 2$ nós trabalhadores preemptivos [3] (*datanodes*); onde l_N corresponde ao nível inferior ($-$) ou superior ($+$) do fator N. Dois nós **não** preemptivos foram mantidos para garantia de disponibilidade. Todos os nós foram gerenciados por um sistema operacional *Debian GNU/Linux 9*.

6.1.3 Linguagens

O *Corpus* PT7 WEB foi minerado com o uso da linguagem *Python*. Detalhes sobre a fonte de dados e o processo de mineração podem ser conferidos no Apêndice A - *Corpus* da Língua Portuguesa. Para a geração dos dados sintéticos e a execução da tarefa de aprendizagem de máquina descrita na Seção 7 foi utilizada a plataforma de *benchmark Intel HiBench 7.0* (HUANG et al., 2011). Para os experimentos descritos na Seção 8 foram desenvolvidas implementações proprietárias em linguagem *Scala* (ODERSKY; SPOON; VENNERS, 2008), e biblioteca *Spark MLLib* (MENG et al., 2016), disponibilizadas no Apêndice B - Implementação das Tarefas de Aprendizagem de Máquina. A análise estatística dos resultados foi conduzida na Linguagem R e suportada pelas bibliotecas de projetos de experimentos fatoriais `FrF2` (GRÖNMPING, 2014) e de coordenadas paralelas `parcoords` (BOSTOCK et al., 2019).

6.2 PROCEDIMENTO METODOLÓGICO

A metodologia está organizada nas fases a seguir, detalhadas no Diagrama 6.1:

- Investigação exploratória
  - Escolha de métricas de desempenho
  - Definição de variáveis experimentais
  - Fixação de intervalos de experimentação
  - Testes de viabilidade
- Condução experimental
  - Estruturação
  - Execução

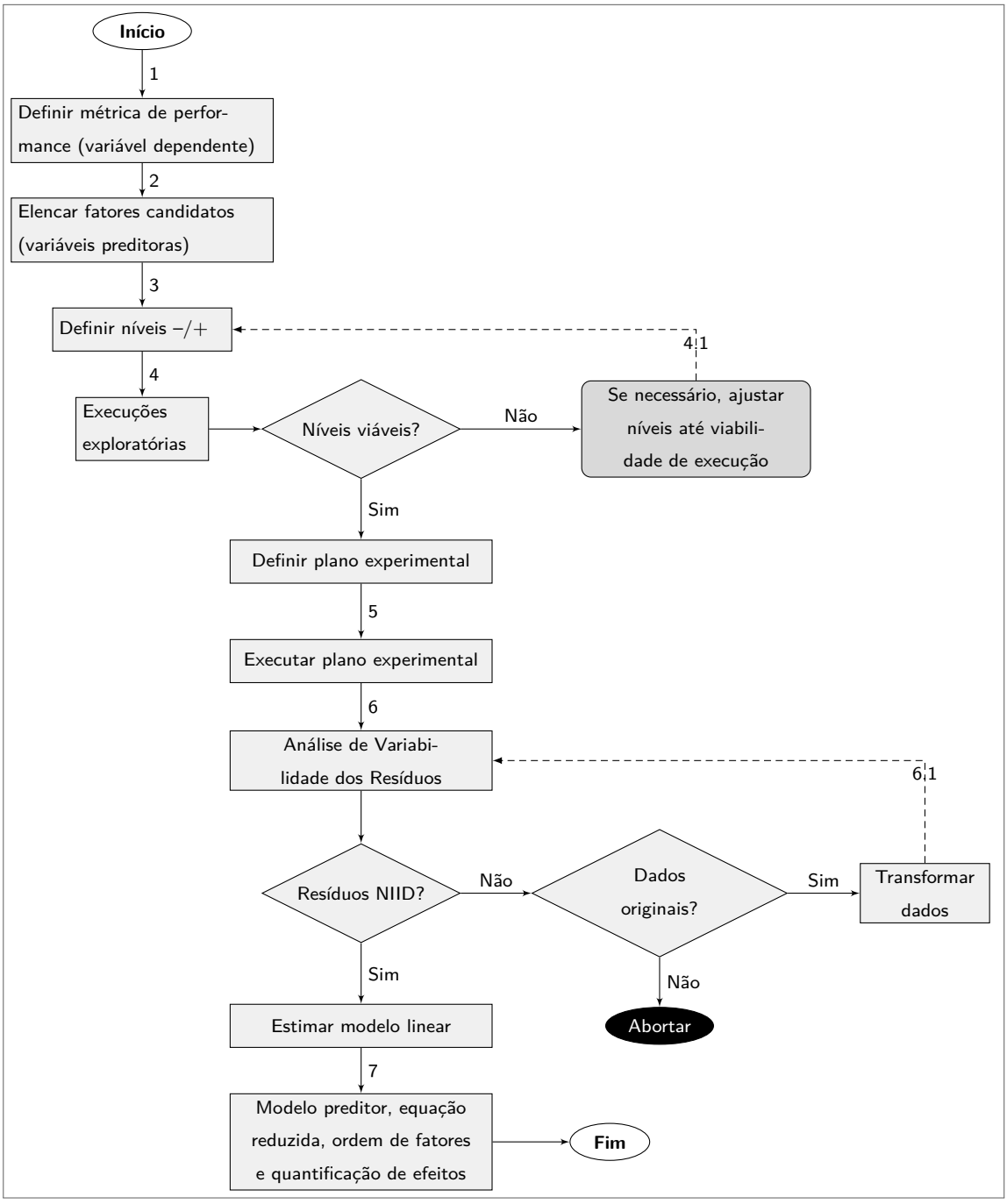
[3] Ofertados a menor custo, mas tomados sem sobreaviso em picos de demanda do provedor.

  - Coleta

- Análise de resultados
  - Análise de variabilidade dos resíduos
  - Triagem e ordenamento de efeitos
- Geração de modelos
- Tomada de decisão

Esta abordagem é capaz de detectar, triar e ordenar fatores relevantes, realizar inferência estatística de seus efeitos, e produzir modelos preditores lineares completos ou reduzidos para o desempenho de ampla gama de problemas algorítmicos sobre *Big Data* cobrindo uma variedade de *frameworks* de processamento. Essa característica é obtida ao se estabelecer procedimento baseado em métricas de desempenho e fatores genéricos. A aplicação dos procedimentos adiante descritos não demandam interferência direta nos algoritmos, em *hardware*, código-fonte dos *frameworks* de processamento ou no provedor de computação em nuvem. Conforme estabelecido na Seção 1 - Introdução, os projetos experimentais da Tese aplicam a metodologia na investigação de tempo e custo de técnicas de aprendizagem de máquina distribuída sobre plataforma *Spark*, em especial, aquelas de natureza iterativa devido ao seu intenso custo computacional.

Diagrama 6.1 – Visão geral da metodologia



Fonte: O autor (2020)

6.2.1 Investigação Exploratória

Tendo como objetivo primordial a análise de viabilidade da pesquisa, a fase de exploração é composta por tarefas que, uma vez suprimidas ou mal conduzidas, resultarão em danos ao

processo de observação, execução, coleta ou mesmo análise. Trata-se de uma etapa em que o pesquisador ainda não tem certeza sobre a viabilidade da pesquisa e faz-se necessário estipular métricas, fatores, valores de variação para cada fator e validar se uma parte ou mesmo todas as unidades experimentais são exequíveis. Sempre que possível, recomenda-se que os passos a seguir sejam seguidos em ordem, a saber: definir métrica ou métricas de desempenho, enumerar as variáveis experimentais sobre as quais se deseja realizar a inferência estatística, especificar o intervalo no qual se deseja testar a variação de níveis e proceder com testes de viabilidade das unidades experimentais.

6.2.1.1 Escolha de Métricas de Desempenho

A pesquisa experimental começa primordialmente por estabelecer uma métrica ou conjunto de métricas de desempenho para a(s) qual(is) se deseja obter conhecimento. Normalmente, o interesse pela métrica reside na necessidade por otimização. Dentre as métricas de desempenho mais comuns encontram-se tempo de execução e custo monetário. Todavia, no contexto do processamento distribuído de dados, há uma série de métricas com potencial para descoberta de conhecimento útil, a saber:

- quantidade de operações de entrada e saída em armazenamentos secundários;
- número de pacotes de rede entre nós de um *cluster* distribuído;
- volume de dados gerado em operações *shuffle* do paradigma *MapReduce*;
- número de vezes em que uma operação de coleta de lixo em memória foi requisitada;
- dentre inúmeras outras.

A escolha da métrica de desempenho é intuitiva e se justifica pelas necessidades da pesquisa e pelas restrições do ambiente no qual o pesquisador está inserido. Importa ressaltar que a métrica escolhida impacta nas variáveis experimentais (fatores) envolvidas na pesquisa. Nesta Tese, pelos motivos apresentados nas Seções 1 - Introdução e 2 - Big Data, optou-se por investigar o desempenho em termos de tempo de execução e de custo monetário de aplicações distribuídas no *framework Spark*.

6.2.1.2 Definição de Variáveis Experimentais

Neste passo da investigação exploratória o pesquisador deve elencar que variáveis experimentais se supõe ter relação com o desempenho. Os fatores podem ser quantitativos, cuja variação é expressa como valor numérico discreto ou contínuo, ou qualitativo, quando tipifica alguma situação ou condição. Os fatores podem ser ainda classificados como variáveis primárias ou variáveis ocultas. No primeiro caso são conhecidas pelo pesquisador e este pode controlar a sua variação. No segundo caso não se deseja ou mesmo não se pode controlá-las ou mantê-las constantes durante o processo de experimentação. Em muitas situações do mundo real existem variáveis ocultas introduzindo ruído e produzindo variabilidade na resposta.

Nos projetos experimentais da Tese, as variáveis primárias escolhidas foram o número de nós ou máquinas do *cluster*, a quantidade de núcleos de processamento existente em cada nó, a quantidade de memória RAM em cada nó, o número de discos SSD por nó e o volume dos dados de entrada para o algoritmo.

Dentre as variáveis ocultas conhecidas no contexto da pesquisa encontram-se a preempção de um nó pelo provedor na nuvem, a variação de oferta do preço de componentes devido a demanda de uso no momento de alocação e o tempo de restituição do nó preemptivo ao *cluster* quando de sua interrupção anterior. As variáveis ocultas desconhecidas são todas aquelas que afetam a dinâmica de processamento no âmbito do sistema operacional, do tráfico de rede e da natureza do problema, a exemplo da distribuição de dados, dentre outras. O ruído produzido pelas variáveis ocultas é contingenciado pela separação das unidades experimentais em blocos com ordem de execução randomizada, aproximando este de uma distribuição Normal, independente e aleatória.

6.2.1.3 Fixação de Intervalos de Experimentação

Após definidos os fatores a serem investigados, o pesquisador deve escolher apropriadamente o intervalo no qual a inferência estatística e os modelos serão analisados e produzidos (níveis de variação $-1/+1$). Importa que a magnitude da diferença entre níveis seja apropriada. Quando os intervalos são muito pequenos, pode-se omitir efeitos que só surgiriam em graus de variação maiores. Muitas vezes a escolha dos níveis se dá por conhecimento *a priori* do especialista, ou ainda por restrições onde determinados níveis simplesmente não são possíveis ou não fazem sentido.

No contexto desta Tese, os níveis foram escolhidos conforme Tabela 8 para Florestas Aleatórias sobre dados sintéticos e Tabela 9 para cinco algoritmos sobre dados reais. Um exemplo de limitação de nível reside na escolha por doze núcleos de processamento como nível –1 para o fator. A documentação da plataforma *Spark* sugere oito a dezesseis núcleos por nó para se obter ganho significativo do processamento paralelo em larga escala. Para a Tese, escolheu-se o ponto médio dos limites sugeridos pelo fornecedor. Importa destacar que o número de núcleos de processamento de um nó do *cluster* estabelece limites para a utilização da memória. Esta, se definida em Gigabytes (GB) produziria quatro níveis de variação distintos, o que fere princípios do projeto 2^k. Optou-se por tratar os níveis de memória como variável discreta proporcional em termos do número de núcleos. Ou seja, um nó do *cluster* pode ter 3 (–) ou 6 (+) vezes o número de núcleos de processamento, em Gigabytes (GB). No caso concreto, o fator variou nos níveis –1, $+1 =$ 3x, 6x.

6.2.1.4 Testes de Viabilidade

Estabelecidas as métricas de desempenho, os fatores e os intervalos de experimentação, o pesquisador deve se ater aos testes de viabilidade de execução. Empiricamente, a partir da escolha dos fatores já se presume que os experimentos serão exequíveis. Entretanto, alguns experimentos sobre unidades experimentais específicas podem se mostrar impossíveis, pouco práticos ou mesmo inconclusivos. Dentre as razões para tal, podem-se destacar: custo de operação, impossibilidade física, requisitos difíceis para inicialização do experimentos ou mesmo longa duração para alcançar um término. No contexto da computação distribuída, por exemplo, é intuitivo imaginar que a construção de um *cluster* com milhares de nós incorreria em tempo e custo consideráveis. Da mesma forma, um experimento viável do ponto de vista físico e monetário pode simplesmente não concluir, não retornando resultados devido a erros inerentes ao processo.

No contexto desta Tese observou-se que quantidades de memória abaixo de 3 GB por núcleo para altos volumes de dados produziam erros e interrupções de serviço devido a limites mínimos de operação. Por esta razão o nível mínimo foi reajustado para 3x. Recomenda-se que o teste de viabilidade inclua fatores ou combinações de níveis considerados críticos *a priori* em matéria de operação [4], como é o caso da já citada memória, *clusters* com reduzida quantidade

[4] Por crítico compreenda-se: "uma combinação que inviabilize a execução de uma unidade experimental comprometendo o estudo". Não confundir com fator relevante, que é parte do objetivo da pesquisa.

de nós ou cenários com poucos núcleos de processamento por nó. Esta fase é representada no Diagrama 6.1 como um *loop* no passo 4.1, indicando a necessidade de repetição de testes exploratórios e ajustes de níveis e fatores até que sejam encontrados cenários viáveis.

6.2.2 Condução Experimental

Ao iniciar esta fase as métricas de desempenho, fatores, níveis e condições de experimentação precisam estar definidas e validadas, pois servirão de entrada para a criação do plano experimental. As etapas de condução compreendem: estruturar o plano, executá-lo e coletar os resultados observados.

6.2.2.1 Estruturação do Plano Experimental

Define-se o número de experimentos, de replicações, ordem de execução de cada unidade experimental e a resolução do projeto (Seção 4.2.3.2). Tome-se por exemplo sete fatores hipotéticos A, B, ..., G, arranjados em um projeto fracionado randomizado 2^{7-4}_{III} com três replicações, conforme Tabela 1. Com base nas Equações 4.1 e 4.2, um estudo completo demandaria $2^7 \times 3 = 384$ experimentos para detecção de todos os 127 efeitos. No exemplo, o esforço experimental é reduzido para 24 experimentos, sob a penalidade de detectar apenas 28 efeitos, sendo sete efeitos relativos aos fatores principais e 21 efeitos referentes às interações de segunda ordem confundidos com os primeiros. Realce é aplicado nas replicações randomizadas da unidade experimental nº 5 para exemplificação do conceito. Perceba-se que as execuções 6, 11 e 17 do plano equivalem à mesma unidade experimental, em diferentes execuções, a saber: a 6ª do bloco 1, 3ª do bloco 2 e 1ª do bloco 3.

Os experimentos descritos na Seção 7 empregam projeto fatorial 2^4 completo, capturando todos os efeitos para quatro fatores e todas as interações até sua mais alta ordem. Os demais experimentos, descritos nas Seções 8.2 a 8.6, empregam projeto fatorial 2^{5-1}_{V} fracionado, capturando os efeitos de cinco fatores e suas interações até segunda ordem sem confundimento.

6.2.2.2 Execução do Plano Experimental

Fase em que o pesquisador efetivamente aplica os testes, acompanha o andamento do plano e a execução de cada unidade experimental, certificando-se de sua conclusão e da

Tabela 1 – Projeto fatorial fracionado 2^{7-4}_{III} com replicações

Execução n.º	Bloco	Unid. Experimental	A	B	C	D	E	F	G
1	1	2	1	-1	-1	-1	-1	1	1
2	1	7	-1	1	1	-1	-1	1	-1
3	1	1	-1	-1	-1	1	1	1	-1
4	1	8	1	1	1	1	1	1	1
5	1	4	1	1	-1	1	-1	-1	-1
6	1	5	-1	-1	1	1	-1	-1	1
7	1	6	1	-1	1	-1	1	-1	-1
8	1	3	-1	1	-1	-1	1	-1	1
9	2	7	-1	1	1	-1	-1	1	-1
10	2	2	1	-1	-1	-1	-1	1	1
11	2	5	-1	-1	1	1	-1	-1	1
12	2	8	1	1	1	1	1	1	1
13	2	1	-1	-1	-1	1	1	1	-1
14	2	6	1	-1	1	-1	1	-1	-1
15	2	4	1	1	-1	1	-1	-1	-1
16	2	3	-1	1	-1	-1	1	-1	1
17	3	5	-1	-1	1	1	-1	-1	1
18	3	4	1	1	-1	1	-1	-1	-1
19	3	8	1	1	1	1	1	1	1
20	3	6	1	-1	1	-1	1	-1	-1
21	3	1	-1	-1	-1	1	1	1	-1
22	3	7	-1	1	1	-1	-1	1	-1
23	3	2	1	-1	-1	-1	-1	1	1
24	3	3	-1	1	-1	-1	1	-1	1

Fonte: O autor (2020)

não interferência de fatores externos em seu funcionamento. É primordial que o pesquisador tome nota de qualquer anomalia em prol de decidir a inclusão dos resultados na análise. É comum a utilização de notas escritas ou planilhas eletrônicas, não só para registrar valores como também para proceder com apontamentos acerca de erros, interrupções, acidentes ou quaisquer comportamentos irregulares.

Todos os experimentos foram executados na sequência randomizada definida em cada projeto experimental. Eventualmente, unidades experimentais atípicas apresentaram erros, interrupções ou demora excessiva para conclusão (quando comparadas com suas predecessoras similares). Muitos dos erros foram devidos à interrupção dos nós preemptivos do *cluster* por parte do provedor de serviços em nuvem. Mesmo sendo a plataforma *Spark* dotada de capacidade de recuperação de falhas quando da ocorrência de *RDDs* ou nós perdidos (Seção 2.2.2.3), houve casos de interrupção total de serviço. Para estes, o experimento foi refeito.

6.2.2.3 Coleta de Resultados

As métricas de desempenho definidas na Seção 6.2.1.1 devem ser cuidadosamente aferidas. Quando necessário, repetições de mensuração [5] devem ser conduzidas para evitar coleta equivocada do resultado. Cada mensuração deve ser inserida na exata ordem de execução definida no plano experimental (Seção 6.2.2.1) para utilização na análise de aderência às premissas de emprego de modelos lineares (Seção 5.3).

No contexto particular da pesquisa não se observou a necessidade por repetições de mensuração, uma vez que os valores de tempo e custo foram fornecidos com precisão pela infraestrutura computacional do provedor em nuvem. Assim, todos os resultados empregados nos projetos experimentais foram coletados e registrados com exatidão.

6.2.3 Análise de Resultados

De posse das observações experimentais, o pesquisador deve verificar as premissas para emprego de modelos lineares, tratar os resultados que eventualmente introduzam distorções, e proceder com a triagem de fatores relevantes.

6.2.3.1 Análise de Variabilidade dos Resíduos

Na Seção 5 são introduzidas as bases para análise de modelos lineares e as premissas de normalidade, independência e homogeneidade de variância dos resíduos. Não raro, torna-se necessário proceder com transformações de escala (Seção 5.4) para adequação do modelo aos processos de inferência estatística e predição.

Ocasionalmente são identificadas unidades experimentais que precisam ser novamente executadas, pois mesmo que não tenham sido identificados comportamentos visivelmente anômalos no processo de condução (Seção 6.2.2.2), a existência de pontos discrepantes com valores extremos pode distorcer o modelo. Tais resultados ocorrem pela já citada alta variabilidade existente em experimentos reais — em especial em ambiente computacional de *cluster* de máquinas distribuídas com preempção de nós; mesmo que se conduzam projetos bem planejados e coletas meticulosas.

[5] Não confundir os termos repetição de mensuração com replicação de unidades experimentais, cujas definições encontram-se no Anexo A - Notas Teóricas Complementares, Seção A.1 - Conceitos Experimentais.

6.2.3.2 Triagem e Ordenamento de Efeitos

Uma vez que o modelo se torne aderente, de posse do nível de significância estatística e da proporção de variação explicada para cada fator principal ou gerado (interação), o pesquisador deverá elencar aqueles que produzem impacto sobre o resultado com intervalos de confiança mensuráveis, além de quantificar o efeito de variações isoladas dos mesmos. Quando for o caso, importa verificar se há dispersão de efeitos entre os coeficientes parciais de regressão (Seção 5.5.1) devido a transformações outras além da logarítmica.

6.2.4 Geração de Modelos

A metodologia produz os seguintes resultados:

- Modelo completo;
- Triagem de fatores;
- *Ranking* de impacto;
- Modelo reduzido;
- Inferência estatística de efeitos.

O modelo completo é resultado de todos os fatores e interações do projeto — em sua respectiva resolução; e explica o mais alto percentual de variação sobre a resposta, excluso o percentual não explicado devido a erros e variáveis ocultas. A maior utilidade do modelo completo reside na sua capacidade preditora e na possibilidade de triar os fatores de maior impacto, com suas respectivas proporções de variação explicada, o que permite a derivação de um *ranking* dos fatores de impacto.

O modelo reduzido é artefato de maior utilidade prática por incluir apenas os fatores estatisticamente significativos, reduzindo o número de variáveis na equação e tornando sua aplicação mais fácil e direta. Essa redução normalmente se dá pelo princípio da dispersão dos efeitos onde um sistema ou processo é provavelmente mais influenciado por uma parcela dos efeitos principais e interações de baixa ordem, restando pouca influência das interações de alta ordem (RIDGE, 2007). Frequentemente perceber-se-á redução do coeficiente de determinação

do modelo em favor da maior utilidade prática para a tomada de decisão. Todavia, é comum que tal redução seja irrisória.

Por fim, a inferência estatística dos efeitos dos fatores principais e das interações é extraída a partir dos coeficientes parciais de regressão (Seção 5.2.2) e permite quantificar a mudança sobre a variável de resposta face a alterações nos níveis dos fatores estudados.

6.2.5 Tomada de Decisão

Todas as etapas descritas anteriormente permitem ao pesquisador elaborar uma fundamentação sólida para as decisões relativas aos aspectos do problema investigado. Na Seção 9 - Discussão o leitor poderá conferir o impacto das variáveis experimentais de *hardware* e volume de dados em experimentos reais com *clusters* distribuídos.

6.3 SÍNTESE

Esta seção introduziu uma metodologia sequencial exploratória, com particular atenção voltada para a aplicação de *DoE* sobre métricas de desempenho de tempo e custo em *clusters* distribuídos. O procedimento conduz o usuário de um estágio com nenhum conhecimento (ou conhecimento empírico) sobre o desempenho de algoritmos até o estágio em que os fatores relevantes são identificados e se derivam os modelos e equações que governam tal desempenho com intervalos de confiança mensuráveis e estatisticamente seguros. Os planos experimentais e as análises podem ser realizadas com qualquer *software* estatístico, tais como: R [6], *Statistical Package for the Social Sciences — SPSS* [7], *Statistical Analysis System — SAS* [8], *Minitab* [9], ou mesmo planilhas eletrônicas. Exemplos aplicados da metodologia poderão ser conferidos nos projetos experimentais apresentados nas Seções 7 - Projeto de Experimentos $2^k r$ Fatorial Completo e 8 - Projeto de Experimentos $2^{k-p} r$ Fatorial Fracionado.

[6] https://www.r-project.org
[7] https://www.ibm.com/analytics/spss-statistics-software
[8] https://www.sas.com
[9] https://www.minitab.com

7 PROJETO DE EXPERIMENTOS $2^k r$ FATORIAL COMPLETO

> “ *Run, Forrest, run!* ”
>
> — Jenny Curran, *Fictional Character*, Groom (1986)

Esta seção descreve um projeto experimental para triagem de fatores e criação de modelos preditivos de desempenho da técnica de aprendizagem de máquina supervisionada Florestas Aleatórias — doravante RF1; em ambiente de *cluster* distribuído. Detalhes acerca do algoritmo encontram-se na Seção 2.3.4. As bases teóricas sobre projetos de experimentos, análise de resultados e criação de modelos constam nas Seções 4 e 5, enquanto a metodologia empregada pode ser conferida na Seção 6.

7.1 PROJETO EXPERIMENTAL: FLORESTAS ALEATÓRIAS (RF1)

A tarefa computacional compreendeu o treinamento, teste e avaliação de aprendizagem de máquina da técnica Florestas Aleatórias para classificação binária. Os hiperparâmetros para treinamento foram definidos conforme Quadro 7, a seguir:

Quadro 7 – Hiperparâmetros para treinamento da técnica RF1

Hiperparâmetro	**Descrição**	**Valor**
Número de árvores	`hibench.rf.numTrees`	100
Estratégia de divisão [a]	`hibench.rf.featureSubsetStrategy`	`auto`
Medida de impureza	`hibench.rf.impurity`	`gini`
Profundidade máxima	`hibench.rf.maxDepth`	4
Número máximo de *bins*	`hibench.rf.maxBins`	32

[a] – `auto`, `all`, `sqrt`, `log2`, `onethird`. A opção `auto` delega a escolha ao algoritmo.

Fonte: O autor (2020)

7.1.1 Obtenção dos Dados

Uma das maiores barreiras para a execução de *benchmarks* para grandes volumes de dados consiste justamente na obtenção de grandes volumes de dados. Sobre esta questão, Han, Xiaoyi e Jiangtao (2014) afirmam que:

> "Tradicionalmente, embora alguns *benchmarks* utilizem conjuntos de dados reais e assim garantam veracidade, o volume e velocidade dos cenários reais não podem ser adaptados de forma flexível aos diferentes requisitos de *benchmark*. (...) em muitos cenários práticos, obter uma variedade de dados reais não é trivial, uma vez que muitos proprietários de dados não desejam compartilhá-los devido a questões confidenciais. Assim, em *benchmarks* de sistemas *Big Data* tem sido consensual o uso de geradores de dados sintéticos(HAN; XIAOYI; JIANGTAO, 2014)."

Outra barreira reside na transferência dos dados para o *cluster*, pois estes precisam estar disponíveis em rede local, para diminuição da latência de troca de dados entre os nós. Assim, os dados devem ser transferidos das diversas fontes de origem, provocando intenso tráfego de rede e incorrendo em gastos consideráveis de infraestrutura em nuvem.

Ming et al. (2014) destacam a importância de gerar dados sintéticos em grande escala (volume), em diferentes formatos (variedade) e a taxas controladas (velocidade), mantendo importantes características dos dados (veracidade). Por sua vez, Bahmani et al. (2012), Li et al. (2015) e Chen et al. (2014), desenvolveram estudos com bons resultados a partir da utilização de dados sintéticos. Uma comparação entre as ferramentas *Hibench, GridMix, PigMix, YCSB, Performance benchmark, TPC-DS, BigBench, LinkBench* e *CloudSuite* é apresentada por Han, Xiaoyi e Jiangtao (2014). Os autores analisaram as plataformas de *Big Data* as quais as suítes atendiam, as cargas de trabalho disponíveis, a escalabilidade em termos de volume, velocidade, variedade de tipos de dados, além de considerações sobre a veracidade destes.

Os dados utilizados no projeto experimental descrito nesta seção foram gerados diretamente na infraestrutura *Google Cloud*, em um *cluster* específico para tal tarefa. Foi escolhida a plataforma *Intel HiBench 7* (HUANG et al., 2010), devido a:

- sua capacidade de processamento de tarefas tanto em lote quanto em tempo real;
- existência de cargas de trabalho voltadas para aprendizagem de máquina;
- implementação específica voltada para o ecossistema *Hadoop* e *Spark*;
- capacidade de geração de dados sintéticos;
- disponibilidade de código aberto e extensibilidade para outras cargas de trabalho.

Produziu-se uma base balanceada com 50% de classes positivas e 50% de classes negativas. O processo resultou em 3.650.000 instâncias de 40 mil dimensões (números randômicos que seguem uma distribuição Gaussiana), totalizando 1.062 *Terabytes* (TB) de dados divididos em 1500 partições de 742.63 *Megabytes* (MB), cada.

7.1.2 Fatores e Níveis

Os fatores sob estudo foram: (N) número de máquinas ou nós, (C) número de núcleos de processamento por nó, (M) quantidade de memória RAM por nó e (D) quantidade de discos SSD acoplados em cada nó. Os níveis (–) / (+) escolhidos foram: 8 (–) ou 28 (+) nós, 12 (–) ou 32 (+) núcleos, 3 (–) ou 6 (+) vezes memória por núcleo, 1 (–) ou 8 (+) discos SSD. O Quadro 8 sumariza os fatores e seus níveis de variação:

Quadro 8 – Fatores e níveis, projeto 2^4 fatorial completo

Fator	**(–1)**	**(+1)**	**Descrição**
Nós (N)	8	28	Número total de máquinas do *cluster*
Núcleos (C)	12	32	Número de núcleos de processamento por nó
Memória RAM (M)	3x	6x	Proporção de memória RAM em cada nó
Discos SSD (D)	1	8	Quantidade de discos SSD por nó

Fonte: O autor (2020)

Devido à existência de limite mínimo de memória RAM por núcleo no ambiente de execução *Google Cloud*, certas combinações gerariam mais de dois níveis por fator. Como solução estabeleceu-se a memória (M) como uma quantidade proporcional de RAM em termos do número de núcleos. Assim, para M = 3x, leia-se "uma quantidade de memória RAM equivalente a três vezes o número de núcleos, em *Gigabytes* (GB)". Desta forma, o projeto passa a ter dois níveis, máximo e mínimo, e a variável pode ser tratada como independente.

7.1.3 Ambiente de Execução

Cada unidade experimental foi executada em *cluster* efêmero totalmente dedicado à execução da carga de trabalho, sem interferências externas ou concorrência de quaisquer outras tarefas. Os tempos de resposta foram medidos com precisão através dos registros de execução do *cluster* e o custo foi obtido através das funcionalidades de monetização da plataforma *Google Data Proc* com o auxílio da ferramenta *Google Big Query*.

7.1.4 Projeto de Experimentos

Escolheu-se o projeto 2^k fatorial completo randomizado com três replicações, com capacidade para análise de todas as interações até 4ª ordem (N:C:M:D) sem confusão de efeitos. Conforme a Equação 4.1 o estudo compreende $r \times 2^k = 3 \times 2^4 = 48$ experimentos. A Tabela 3 exibe os resultados experimentais para tempo em minutos e custos em moeda Real brasileiro (R$). Os custos para todos os experimentos foram computados sob a mesma cotação de dólar (US$), não havendo imprecisões relacionadas com variação cambial dentro do mesmo projeto experimental. As colunas '$\bar{\mathbf{t}}$' e '$\bar{\mathbf{c}}$' representam suas respectivas médias. A coluna '**Plano**' exibe a sequência randômica de execução de cada unidade experimental. Para compreensão, tome-se o experimento #1, a combinação de fatores foi replicada em três unidades experimentais, executadas na 1ª, 32ª e 46ª posições do projeto.

Tabela 3 – Projeto experimental $2^4 \times 3$ para tempo e custo de RF1

#	Plano [a]	N	C	M	D	t_1, t_2, t_3 (min.)	$\bar{t}$	c_1, c_2, c_3 ($)	$\bar{c}$
1	1-32-46	8 $^-$	12 $^-$	3 $^-$	1 $^-$	(20.13, 21.03, 20.25)	20.47	(7.87, 7.37, 7)	7.41
2	7-17-43	28 $^+$	12 $^-$	3 $^-$	1 $^-$	(6.5, 6.82, 6.55)	6.62	(7.49, 7.46, 7.47)	7.47
3	15-19-48	8 $^-$	32 $^+$	3 $^-$	1 $^-$	(13.37, 13.8, 8.92)	12.03	(12.33, 12.85, 9.37)	11.52
4	6-23-40	28 $^+$	32 $^+$	3 $^-$	1 $^-$	(4.2, 4.48, 4.32)	4.33	(13.18, 14.97, 14.02)	14.06
5	2-24-44	8 $^-$	12 $^-$	6 $^+$	1 $^-$	(20.32, 20.65, 19.5)	20.16	(8.42, 8.53, 8.22)	8.39
6	16-31-37	28 $^+$	12 $^-$	6 $^+$	1 $^-$	(6.43, 7.05, 9.77)	7.75	(8.22, 9.12, 11.24)	9.53
7	10-25-47	8 $^-$	32 $^+$	6 $^+$	1 $^-$	(8.58, 9.4, 9.07)	9.02	(11.24, 11.49, 11.01)	11.25
8	9-27-34	28 $^+$	32 $^+$	6 $^+$	1 $^-$	(3.88, 4.52, 3.97)	4.12	(19.76, 17.61, 16.18)	17.85
9	4-20-41	8 $^-$	12 $^-$	3 $^-$	8 $^+$	(19.68, 19.98, 26.08)	21.91	(10.38, 10.48, 13.96)	11.61
10	5-28-33	28 $^+$	12 $^-$	3 $^-$	8 $^+$	(6.33, 6.85, 6.8)	6.66	(12.67, 13.25, 13.23)	13.05
11	11-22-36	8 $^-$	32 $^+$	3 $^-$	8 $^+$	(11.48, 9.02, 9.02)	9.84	(14.66, 11.95, 12.39)	13
12	14-26-42	28 $^+$	32 $^+$	3 $^-$	8 $^+$	(3.88, 4.1, 3.55)	3.84	(18.81, 19.29, 18.65)	18.92
13	12-21-45	8 $^-$	12 $^-$	6 $^+$	8 $^+$	(19.52, 20.08, 19.32)	19.64	(11.47, 11.86, 11.53)	11.62
14	8-30-35	28 $^+$	12 $^-$	6 $^+$	8 $^+$	(7.77, 7.07, 6.65)	7.16	(16.45, 14.9, 15)	15.45
15	13-18-39	8 $^-$	32 $^+$	6 $^+$	8 $^+$	(8.87, 9.43, 8.92)	9.07	(13.87, 15.95, 13.98)	14.6
16	3-29-38	28 $^+$	32 $^+$	6 $^+$	8 $^+$	(3.68, 5.22, 4.35)	4.42	(21.6, 26.16, 22.82)	23.53

Fonte: O autor (2020), com resultados de 48 experimentos no ambiente *Spark — Google Data Proc*

7.1.5 Análise de Variabilidade de Resíduos e Modelo de Tempo

Padrões incomuns nos gráficos de resíduos indicam violação das suposições de NIID. A Figura 7a exibe os resíduos padronizados em relação aos níveis de cada fator. Observa-se notável diferença nas faixas de dispersão dos resíduos, sugerindo violação da homogeneidade de

variância. A Figura 7b reforça os problemas de homogeneidade, como pode ser visto no gráfico de resíduos *versus* valores ajustados do modelo. O gráfico de resíduos na sequência temporal de execução das unidades experimentais (Figura 7c) insinua que a premissa de independência dos erros é atendida. O gráfico quantil-quantil (Figura 7d) evidencia que os resíduos não se aproximam de uma distribuição Normal. A ausência de normalidade é confirmada pelo teste de Shapiro e Wilk (1965), com *p-value* $=$ *0.0001221*. As unidades experimentais nº4, 41 e 48 são evidenciadas como pontos discrepantes no gráfico da distância de *Cook*, na Figura 7e.

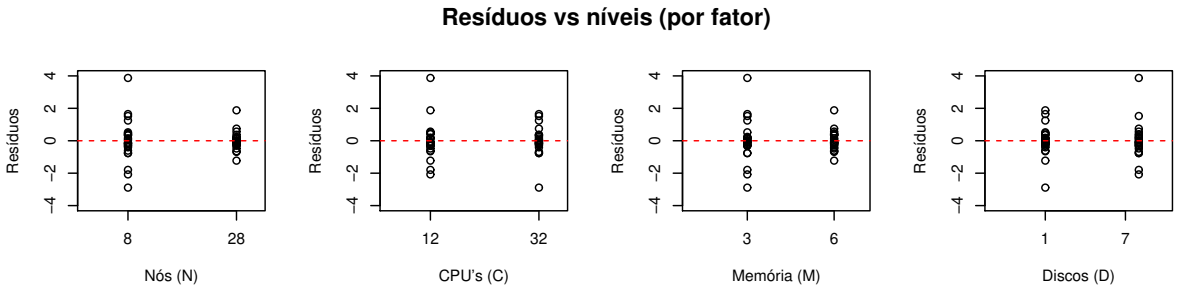
As violações foram corrigidas através de transformação de potência de Box e Cox (1964) para a escala logarítmica ($\lambda = -0.06 \approx 0$). Os efeitos sobre os resíduos do modelo podem ser observados na Figura 8. As distorções de homogeneidade de variância em relação aos níveis de cada fator foram minimizadas (Figura 8a). As Figuras 8b e 8c também evidenciam melhora na qualidade do ajuste, inclusive com diminuição da dispersão entre os resíduos, fato esperado quando da aplicação das transformações de potência. O gráfico quantil-quantil de normalidade dos resíduos [1] mostrou maior aproximação de uma distribuição Normal teórica, confirmada pelo teste de Shapiro e Wilk (1965) com *p-value* $= 0.04742 \approx 0.05$. Por fim, o teste de *Bonferroni* (FOX; WEISBERG, 2019) apontou as observações nº37 e 48 como pontos discrepantes, evidenciados no gráfico de *Cook* (Figura 8e). Todavia, estes apresentaram menor desvio padrão da média quando comparados com o modelo ajustado sobre os dados em escala original.

A Tabela 4 detalha o modelo linear que explica 93.6% da variação para o tempo de execução, $R^2 = 0.975, R^2_{adj} = 0.963$, *F-statistic* $=$ 83.1 em 15 e 32 graus de liberdade e $p-value < 2.2e^{-16}$. Nós e núcleos são os fatores mais relevantes(Figura 9b) com significância estatística, relevância prática, e direção negativa — ou seja: uma relação inversa com o tempo. Quão maior o fator, menor o tempo de execução. M e D não produzem impacto sobre o desempenho (*p-value* $>$ 0.05). As interações N:C e N:M, mesmo estatisticamente significativas explicam 0.79% e 0.63% da resposta. Desse modo, é possível assumi-las como sem significância prática e pode-se rejeitar suas influências no modelo.

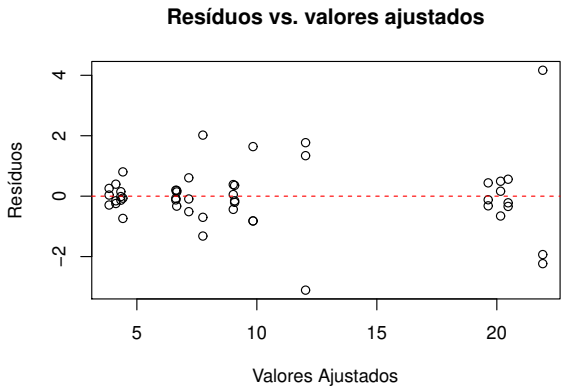
De acordo com a Seção 5.5.2 o modelo para o tempo $t*$ pode ser expresso pela Equação 7.1, removendo os termos insignificantes e convertendo os fatores N e C para $\overline{n} = \left(\frac{N-\frac{(28+8)}{2}}{\frac{28-8}{2}}\right) =$

[1] Não confundir normalidade dos resíduos com normalidade dos dados de resposta.

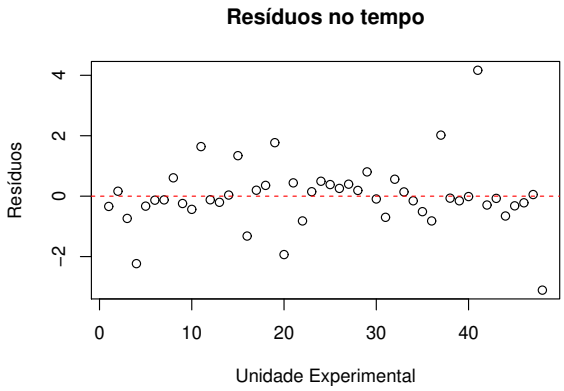
Figura 7 – Resíduos do modelo de tempo de RF1 em escala original



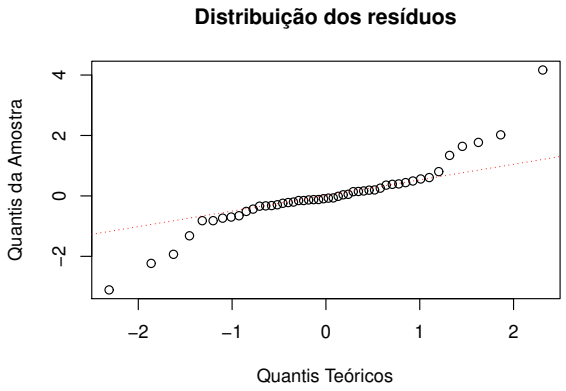
(a) Variância dos resíduos *vs.* níveis dos fatores N, C, M, e D.



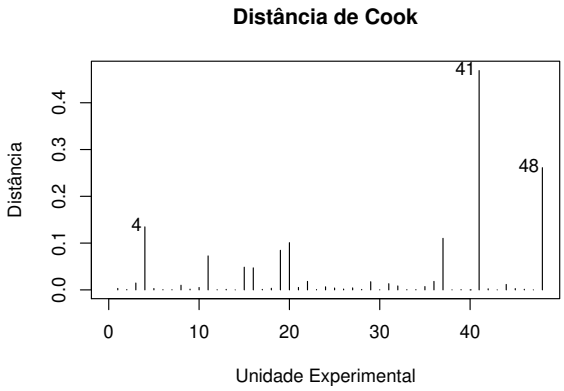
(b) Variância dos resíduos *vs.* valores ajustados.



(c) Resíduos em sequência temporal.



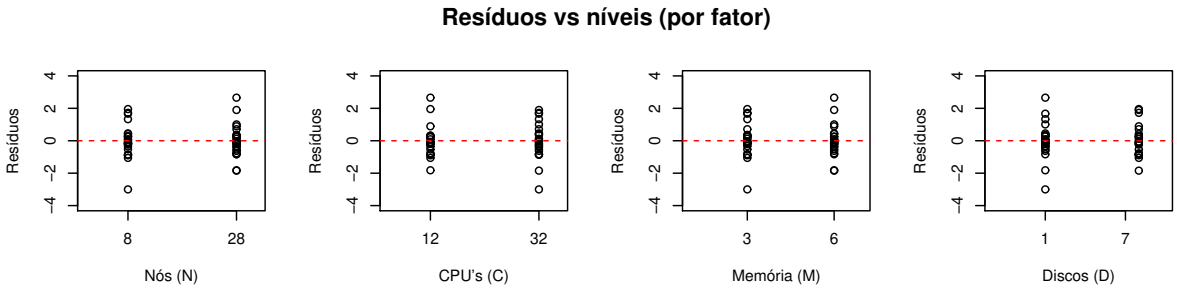
(d) Normalidade dos resíduos.



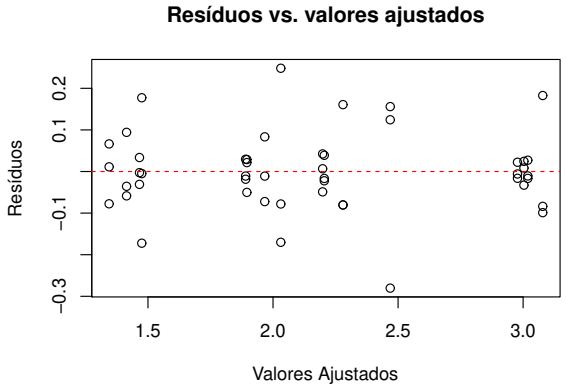
(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

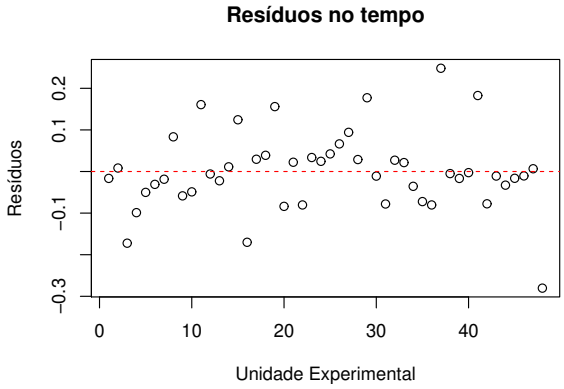
Figura 8 – Resíduos do modelo de tempo de RF1 em escala logarítmica



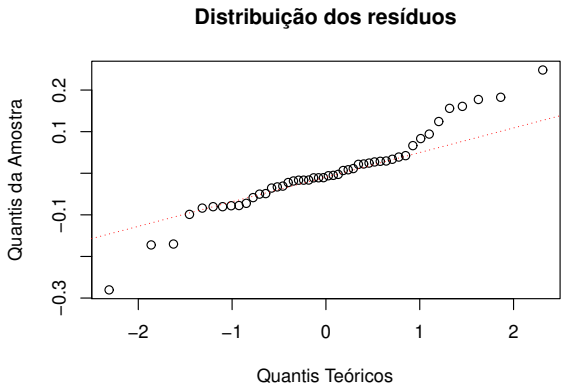
(a) Variância dos resíduos *vs.* níveis dos fatores N, C, M, e D.



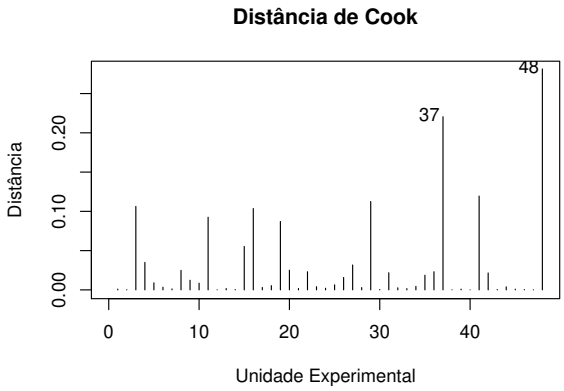
(b) Variância dos resíduos *vs.* valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos extremos.

Fonte: O autor (2020)

Tabela 4 – Modelo linear para tempo de execução de RF1 (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.169613	131.366	$< 2e^{-16}$ ***	- -
N	**-0.484109**	**-29.312**	$< \mathbf{2^{e-16}}$ ***	**67.20**
C	**-0.313117**	**-18.959**	$< \mathbf{2^{e-16}}$ ***	**28.11**
M	-0.010663	-0.646	0.5231	0.03
D	-0.016748	-1.014	0.3182	0.08
N:C	0.052649	3.188	0.0032 **	0.79
N:M	0.047056	2.849	0.0076 **	0.63
C:M	-0.022609	-1.369	0.1806	0.15
N:D	0.001792	0.109	0.9143	0.00
C:D	-0.013604	-0.824	0.4162	0.05
M:D	0.013924	0.843	0.4054	0.06
N:C:M	0.005963	0.361	0.7205	0.01
N:C:D	0.013417	0.812	0.4226	0.05
N:M:D	0.000190	0.012	0.9909	0.00
C:M:D	0.033280	2.015	0.0524	0.32
N:C:M:D	-0.001839	-0.111	0.9120	0.00

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

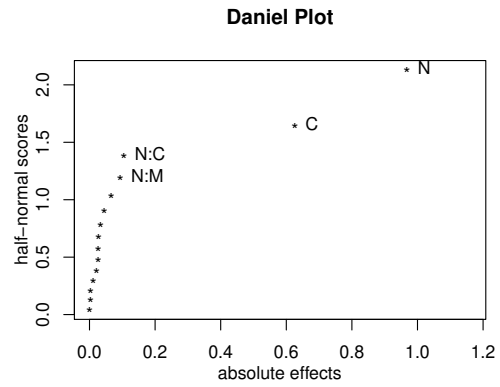
$\left(\frac{N-18}{10}\right)$ e $\overline{c} = \left(\frac{N-\frac{(12+32)}{2}}{\frac{32-12}{2}}\right) = \left(\frac{C-22}{10}\right)$:

$$t_*(N, C) = 2.169613 - 0.484109\,\overline{n} - 0.313117\,\overline{c} \tag{7.1}$$

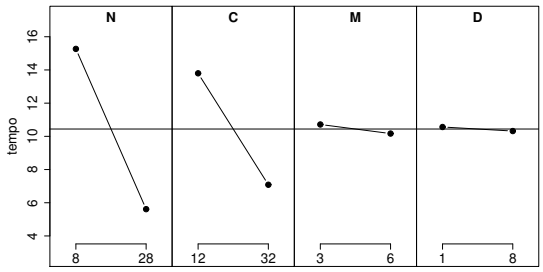
O efeito E_X de um hipotético fator X sobre a variável de resposta equivale ao dobro do coeficiente de regressão β_X. Entretanto, devido à natureza multiplicativa da transformação logarítmica, os efeitos precisam ser ajustados através da transformação inversa $e^{2 \times \beta_X} - 1$, onde $e \approx 2.7182$ é a base do logaritmo natural. Dessa forma, o efeito do número de nós sobre o tempo é $E_N = e^{2 \times -0.484109} - 1 = -0.6202$. Isso significa que, fixado o número de núcleos, se o número de nós aumenta de 8 para 28, em média o tempo de execução de RF1 diminuirá em 62.02%. De maneira análoga, $E_C = e^{2 \times -0.313117} - 1 = -0.4654$, ou seja, fixado o número de nós, se o número de núcleos por nó aumentar de 12 para 32, o tempo diminuirá em 46.54%.

A Equação 7.1 explica 95.32% da resposta ($R^2 = 0.9532$ e $R^2_{Adj} = 0.9511$) e pode ser utilizada para estimar o tempo de execução de RF1 para configurações de *cluster* de 8 a 28 nós e qualquer capacidade possível de 12 a 32 núcleos por nó.

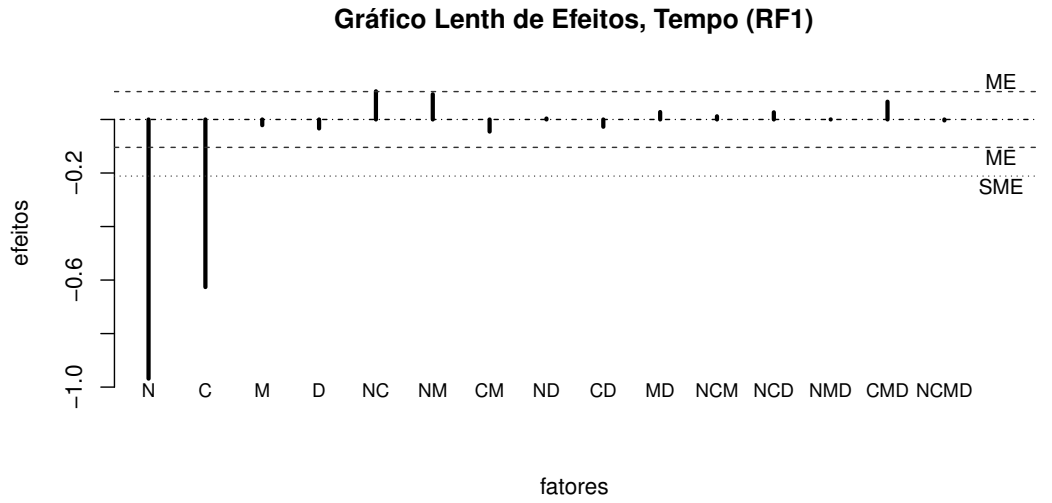
Figura 9 – Análise gráfica de modelo linear para tempo de RF1



(a) Ranking de efeitos significativos.



(b) Magnitude dos efeitos principais.



(c) Efeitos no tempo de execução de RF1.

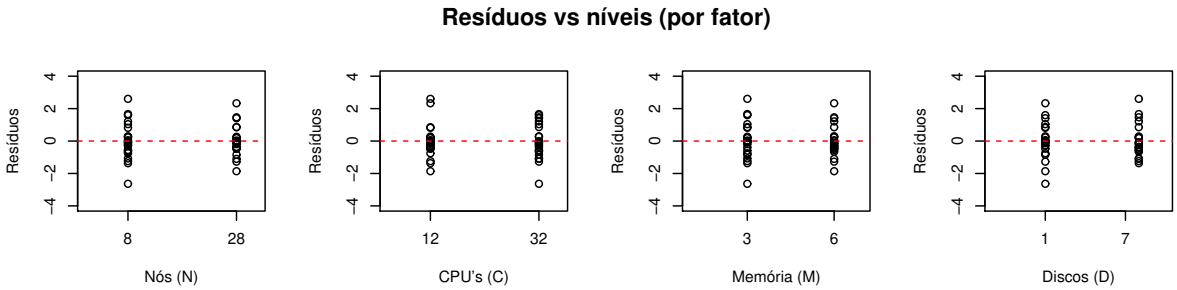
Fonte: O autor (2020)

7.1.6 Análise de Variabilidade de Resíduos e Modelo de Custo

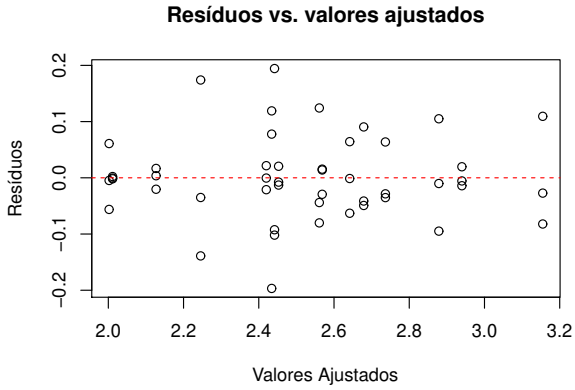
Por simplicidade, para esta análise e nos projetos experimentais 8.2 a 8.6 doravante, serão exibidos apenas os gráficos de resíduos para o modelo ajustado após transformações de potência. O leitor deve assumir que foi conduzida investigação similar à Seção 7.1.5.

Os resíduos do modelo ajustado sobre os dados originais apontaram violações de homogeneidade de variância em relação aos níveis do fator C. O gráfico de resíduos *versus* valores ajustados endossou a heterogeneidade de variância. O gráfico quantil-quantil não evidenciou problemas quanto à normalidade dos resíduos, que foi confirmada pelo teste de Shapiro e Wilk

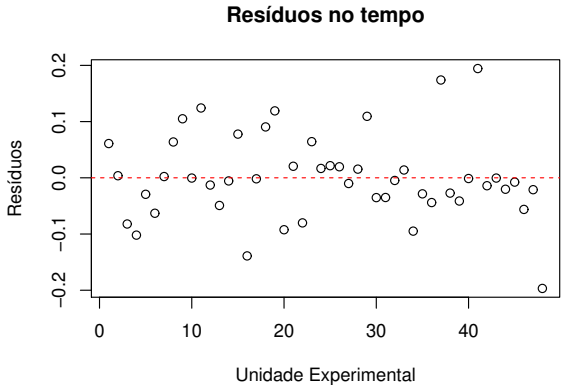
Figura 10 – Resíduos do modelo de custo de RF1 em escala logarítmica



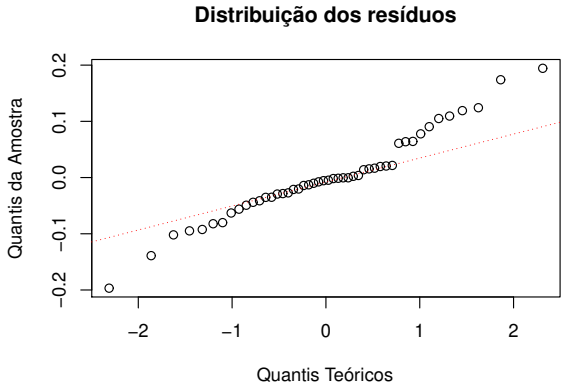
(a) Variância dos resíduos *vs.* níveis dos fatores N, C, M e D.



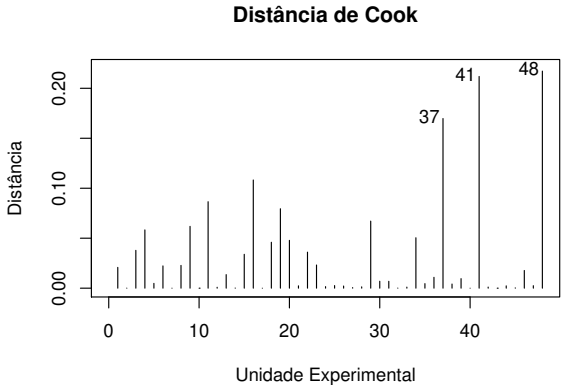
(b) Resíduos *vs.* valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

(1965) com p-value = 0.09864. Também não se identificaram problemas de independência de erros e pontos discrepantes. O cálculo $\lambda = -0.02 \approx 0$ recomendou transformação de potência de Box e Cox (1964) para a escala logarítmica.

Após transformação, os resíduos em face dos níveis *vs.* valores ajustados passaram a aderir às exigências NIID (Figuras 10a e 10b). A Figura 10c confirma a manutenção de independência

dos erros. A Figura 10d deixa dúvidas em relação à normalidade, que é confirmada pelo *p-value* $= 0.189543$ do teste de Shapiro e Wilk (1965). O teste de *Bonferroni* (FOX; WEISBERG, 2019) não revelou a existência de pontos discrepantes, embora as unidades experimentais 37, 41 e 48 mostrem até dois desvios padrões (2σ) no gráfico de *Cook* da Figura 10e.

Tabela 5 – Modelo linear para custo de execução de RF1 (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.5184	190.856	$< 2e^{-16}$ ***	- -
N	**0.1287**	**9.755**	$\mathbf{4.14e^{-11}}$ ***	**15.85**
C	**0.1952**	**14.796**	$\mathbf{7.31e^{-16}}$ ***	**36.45**
M	**0.0683**	**5.178**	$\mathbf{1.18e^{-05}}$ ***	**4.46**
D	**0.1734**	**13.138**	$\mathbf{1.94e^{-14}}$ ***	**28.74**
N:C	**0.0614**	**4.654**	$\mathbf{5.41e^{-05}}$ ***	**3.61**
N:M	**0.0384**	**2.912**	**0.0065** **	**1.41**
C:M	0.0011	0.084	0.93366	0.00
N:D	0.0295	2.234	0.03259 *	0.83
C:D	**-0.053**	**-4.043**	**0.00031** ***	**2.72**
M:D	-0.0044	-0.33	0.74324	0.02
N:C:M	0.0051	0.386	0.70186	0.02
N:C:D	-0.0059	-0.446	0.65878	0.03
N:M:D	-0.0067	-0.505	0.61736	0.04
C:M:D	0.0182	1.376	0.17823	0.32
N:C:M:D	-0.0126	-0.958	0.34529	0.15

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

O modelo linear da Tabela 5 explica 94.67% do custo de execução, com $R^2 = 0.9467$, $R^2_{adj} = 0.9217$, *F-statistic* $= 37.9$ com 15 e 32 graus de liberdade e *p-value* $< 4.145e^{-16}$. A Figura 11b apresenta todos os efeitos com direção positiva: maior o fator, maior o custo. Embora a conclusão seja intuitiva, o modelo revelou precedência entre os fatores, validando a capacidade da metodologia de capturar a dinâmica da execução algorítmica. O maior impacto se deve ao número de núcleos por nó, seguido pelo número de discos, número de nós e pela quantidade de memória por nó. Os fatores principais e as interações N:C, N:M, C:D e N:D são estatisticamente significativas. Todavia, a interação N:D contribui com apenas 0.83% sobre a resposta e sua influência pode ser rejeitada.

A Equação 7.2 estima o custo $c*$ de RF1, onde $\overline{n}$ e $\overline{c}$ possuem o mesmo significado expresso

na Equação 7.1, enquanto $\overline{m} = \left(\frac{M-4.5}{1.5}\right)$ e $\overline{d} = \left(\frac{D-4.5}{3.5}\right)$.

$$c_*(N,C,M,D) = 2.5184 + 0.1287\,\overline{n} + 0.1952\,\overline{c} + 0.0683\,\overline{m} + 0.1734\,\overline{d} + 0.0614\,\overline{nc} + 0.0384\,\overline{nm} - 0.053\,\overline{cd} \tag{7.2}$$

O efeito do número de nós $E_N = e^{2\times 0.1287} - 1 = 0.2936$. Ou seja, o custo de execução aumenta em média 29.36% como resultado da variação de 8 para 28 nós. Da mesma forma, $E_C = e^{2\times 0.1952} - 1 = 0.4776$, ou 47.76%. $E_M = e^{2\times 0.0683} - 1 = 0.1464 = 14.64\%$. Enquanto $E_D = e^{2\times 0.1734} - 1 = 0.4145$, ou um aumento de custo de 41.45%. Entretanto, devido à presença de interações, os efeitos principais não devem ser tratados como isolados. As dependências podem ser melhor compreendidas pela análise dos gráficos de interação. Na Figura 11c (N:C) observa-se que a variação de custo não é linear, existindo um efeito multiplicativo quando o número de nós é aumentado em conjunto com o número de núcleos. Perceba-se a diferença de magnitude sobre o custo para os níveis inferior e superior dos dois fatores. Efeito semelhante ocorre com a interação N:M, exibida na Figura 11d. O mesmo efeito multiplicativo não é percebido na Figura 11e, relativa à interação N:D.

A Equação 7.2 explica 93.25% ($R^2 = 0.9325$ e $R^2_{Adj} = 0.9207$) do custo de execução considerando este conjunto particular de dados, sua localidade nos nós do *cluster*, distribuição de valores e implementação específica. O modelo estima o custo para variações de *cluster* de 8 a 28 nós, 12 a 32 núcleos, fator de memória de 3x a 6x e 1 a 8 discos por nó.

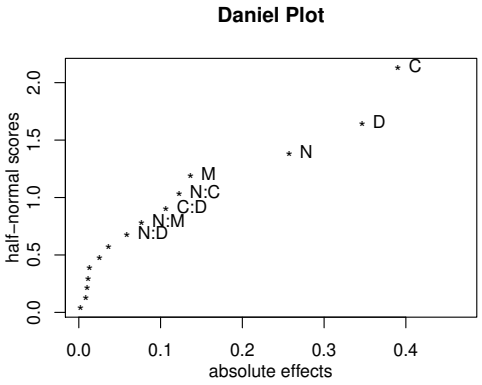
7.1.7 Discussão

Reunindo os dois modelos, como a memória não tem efeito no tempo e pouco efeito no custo, uma recomendação direta para o fornecimento de *hardware* seria (i) aumento no número de nós do *cluster* ao invés da quantidade de núcleos por nó, e (ii) nenhum aumento nos discos SSD, pois não produzem efeitos sobre o tempo, mas influenciam o custo (41%) em magnitude semelhante ao fator mais caro (47%). Em suma, mais nós, menos núcleos por nó, memória não afeta o tempo e pouco afeta os custos de execução, enquanto discos produzem apenas aumento de custo, incorrendo em desperdício.

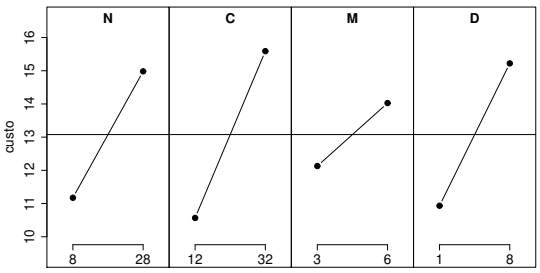
No intervalo do estudo, há 24.000 configurações de *hardware* possíveis [2]. Os modelos permitem estimar o custo em termos do menor tempo, o tempo em termos do menor custo ou

[2] Variação de nós (N) e discos (D) um a um, núcleos (C) dois a dois, e memória (M) a cada 0.25 GB.

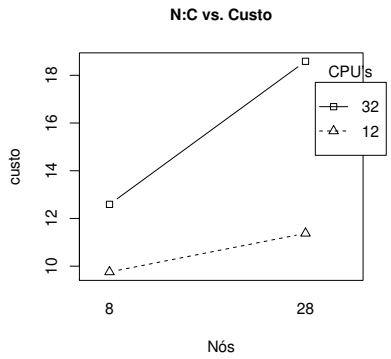
Figura 11 – Análise gráfica de modelo linear para custo de RF1



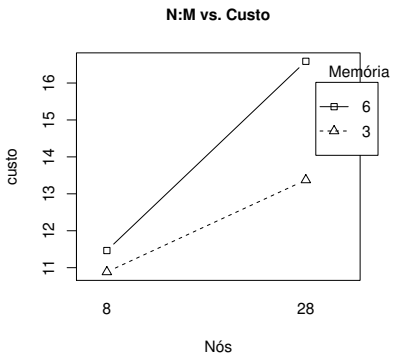
(a) Ranking de efeitos significativos.



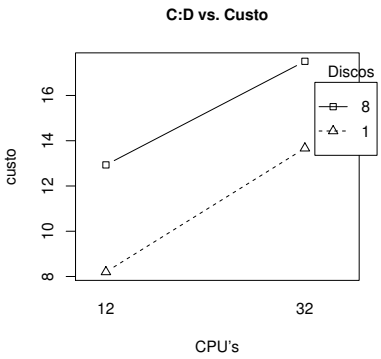
(b) Magnitude dos efeitos principais.



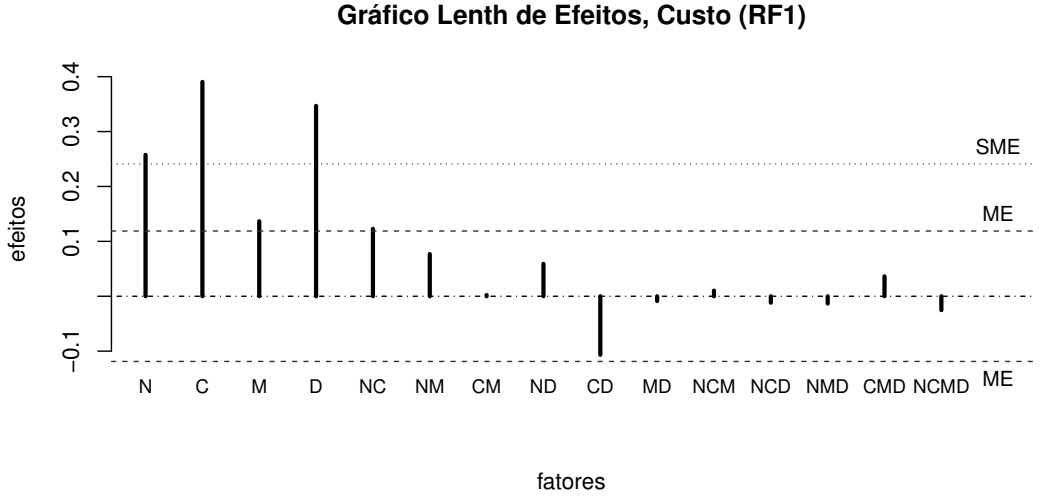
(c) Interação N:C.



(d) Interação N:M.



(e) Interação C:D.



(f) Efeitos do custo de execução de RF1.

Fonte: O autor (2020)

ainda estimar configurações para portfólios combinados estipulando limites superiores ou inferiores. Para fins de ilustração: o menor tempo de execução de RF1 é alcançado por um *cluster* $[N_{28}, C_{32}]$ [3] em 3.94 $\rightarrow$ (2.14; 7.26)$_{\text{IC}}$ minutos, ao custo de \$14.54 $\rightarrow$ (\$11.78; \$17.93)$_{\text{IC}}$. Da mesma forma, a configuração mais barata resulta do *cluster* $[N_8, C_{12}, M_3, D_1]$, equivalente a \$7.39 $\rightarrow$ (\$5.99; \$9.11)$_{\text{IC}}$, executando por 19.43 (10.55; 35.76) minutos. Finalmente, para executar o algoritmo em até 8 minutos, gastando não mais que \$10, a melhor configuração seria $[N_{27}, C_{20}, M_3, D_1]$, sendo executado em 6.03 $\rightarrow$ (3.28; 11.09)$_{\text{IC}}$ minutos, custando \$9.95 $\rightarrow$ (\$8.06; \$12.27)$_{\text{IC}}$.

[3] Memória (M) e disco (D) não interferem no resultado, conforme Tabela 4 e Equação 7.1.

8 PROJETO DE EXPERIMENTOS 2^{k-p}_r FATORIAL FRACIONADO

> “ *Divide et Impera.* ”
>
> — Philip II, *King of Macedon, B.C. 4c.*, Paragi (2016)

Esta seção detalha cinco projetos experimentais de triagem de fatores de *hardware* e volume de dados, e a criação de modelos preditivos de desempenho de técnicas de aprendizagem de máquina supervisionada para classificação binária de documentos de um *Corpus* da Língua Portuguesa. O referencial teórico sobre as técnicas de aprendizagem de máquina pode ser encontrado nas Seções 2.3.1 a 2.3.5. As bases teóricas para compreensão do processo experimental constam nas Seções 4 e 5, enquanto a metodologia empregada pode ser conferida na Seção 6. Um resumo dos dados de entrada encontra-se na Seção 8.1.1 e sua obtenção e pré-processamento são detalhados no Apêndice A - *Corpus* da Língua Portuguesa.

8.1 PREÂMBULO

Embora um projeto de experimentos fatorial completo seja capaz de investigar os fatores principais e capturar todas as suas interações até a mais alta ordem, seu custo se tornaria inviável e uma organização mais coerente na forma de projeto fatorial fracionado foi utilizada. O contexto descrito a seguir se aplica aos projetos experimentais das Seções 8.2 a 8.6.

8.1.1 Algoritmos e Dados

As técnicas empregadas foram: Regressão Logística (RL), Máquinas de Vetores de Suporte (SVM), *Naïve Bayes* (NB), Florestas Aleatórias (RF2) e *Perceptron* de Múltiplas Camadas (MLP). Todas foram submetidas ao mesmo conjunto de dados e fatores, variando-se apenas a randomização do plano experimental. Os dados de entrada consistiram de um *Corpus* real (PT7 WEB) produzido a partir da mineração de páginas *Web* de sete países de Língua Portuguesa [1], a partir do repositório *Common Crawl*. Os dados em formato *HTML* bruto totalizaram 249 GB de dados, que resultaram em 36.49 GB após processo de extração e transformação em vetores de características apropriado ao processamento das técnicas de aprendizagem de máquina. Um total de 16.346.693 de páginas foram capturadas, processadas e rotuladas como instâncias da

[1] Angola, Brasil, Portugal, Cabo Verde, Guiné-Bissau, Macau e Moçambique.

Língua Portuguesa do Brasil (classe positiva) e Língua Portuguesa dos demais países (classe negativa). O conjunto de entrada foi balanceado com 55.2% de páginas em português não brasileiro e 44.8% de páginas em português brasileiro.

8.1.2 Fatores e Níveis

O Quadro 9 sumariza os cinco fatores sob investigação, seus respectivos níveis –1 e +1 e a conversão de escala de acordo com a Equação 5.6 da Seção 5.5.2.

Quadro 9 – Fatores e níveis, projeto 2^{5-1}_{V} fatorial fracionado

Fator	(–1)	(+1)	Fórmula de Conversão	Descrição
Nós (N)	8	28	$\overline{n} = \left(\frac{N-\frac{(8+28)}{2}}{\frac{28-8}{2}}\right) = \frac{N-18}{10}$	Número total de máquinas do *cluster*
Núcleos (C)	12	32	$\overline{c} = \left(\frac{C-\frac{(12+32)}{2}}{\frac{32-12}{2}}\right) = \frac{C-22}{10}$	Número de núcleos de processamento em cada nó
Memória (M)	3x	6x	$\overline{m}\left(\frac{M-\frac{(3+6)}{2}}{\frac{6-3}{2}}\right) = \frac{M-4.5}{1.5}$	Proporção de RAM por nó, face ao número de núcleos
Discos (D)	1	7	$\overline{d} = \left(\frac{D-\frac{(1+7)}{2}}{\frac{7-1}{2}}\right) = \frac{D-4}{3}$	Quantidade de discos SSD acoplados em cada nó
Volume (V)	3.4%	100%	$\overline{v} = \left(\frac{V-\frac{(3.4+100)}{2}}{\frac{100-3.4}{2}}\right) = \frac{V-51.7}{48.3}$	Percentual de volume dos dados de entrada

Fonte: O autor (2020)

As mesmas observações relativas ao fator Memória, descritas na Seção 7.1.2, foram consideradas na execução dos demais projetos experimentais. O valor para o nível –1 do fator Volume foi obtido após divisão do conjunto original do *Corpus* em 5% do total de páginas resultando em 3.4% do total em Gigabytes (GB).

8.1.3 Ambiente de Execução

Todos os experimentos doravante foram conduzidos em *clusters Spark* hospedados no ambiente *Google Cloud Data Proc* [2]. A tarefa computacional compreendeu o treinamento, teste e avaliação de métricas de aprendizagem de máquina para classificação binária de textos da

[2] https://console.cloud.google.com/dataproc/

Língua Portuguesa. Cada unidade experimental foi executada em *cluster* efêmero totalmente dedicado à execução da carga de trabalho. Os tempos de execução foram coletados com precisão a partir dos *logs* do *cluster* e o custo foi obtido através das funcionalidades de monetização do *Google Big Query*.

8.1.4 Projeto de Experimentos

Escolheu-se o projeto randomizado 2^{5-1}_{V} fatorial fracionado com três replicações, capaz de analisar todas as interações até 2ª ordem, sem confundimento de efeitos (Seção 4.2.3.2). De acordo com a Equação 4.1, um projeto fatorial completo com k = 5 fatores e r = 3 replicações resultaria em $2^k \times r = 2^5 \times 3 = 96$ experimentos. Ao negligenciar interações de ordem mais alta em razão de minimizar o esforço experimental, o projeto experimental fracionado resulta em $2^{(k-p)} \times r = 2^{(5-1)} \times 3 = 48$ experimentos.

O tipo e a resolução de projeto para os experimentos das Seções 8.2 a 8.6 estão citados neste preâmbulo por simplificação da apresentação textual. Os detalhes de cada projeto experimental se encontram nas Tabelas 7, 10, 13, 16 e 19 e obedecem à mesma dinâmica de nomenclatura definida para a Tabela 3 da Seção 7.1.4. Os custos para todos os experimentos foram computados sob a mesma cotação de dólar (US$), não havendo imprecisões relacionadas com variação cambial dentro do mesmo projeto experimental.

8.2 PROJETO EXPERIMENTAL: REGRESSÃO LOGÍSTICA (LR)

O Quadro 10 exibe os hiperparâmetros utilizados no treinamento, enquanto a Tabela 7 detalha o projeto experimental empregado no estudo.

Quadro 10 – Hiperparâmetros para treinamento da técnica LR

Hiperparâmetro	**Descrição**	**Valor**
Distribuição	`lr.setFamily`	`binomial`
Parâmetro *aggregation depth*	`lr.setAggregationDepth`	2
Parâmetro *elastic net*	`lr.setElasticNetParam`	0
Regularização	`lr.setRegParam`	0
Máximo de iterações	`lr.setMaxIter`	10
Tolerância de convergência	`lr.setTol`	10^{-6}

Fonte: O autor (2020)

Tabela 7 – Projeto experimental $2_V^{5-1} \times 3$ para tempo e custo de LR

#	Plano	N	C	M	D	V	t_1, t_2, t_3 (min.)	$\bar{t}$	c_1, c_2, c_3 ($)	$\bar{c}$
1	15-28-34	8 $^-$	12 $^-$	3 $^-$	1 $^-$	100 $^+$	(3.2, 3.02, 2.97)	3.06	(1.44, 1.37, 1.43)	1.41
2	5-22-43	28 $^+$	12 $^-$	3 $^-$	1 $^-$	3.4 $^-$	(1.5, 1.53, 1.68)	1.57	(2.3, 4.13, 2.42)	2.95
3	12-31-35	8 $^-$	32 $^+$	3 $^-$	1 $^-$	3.4 $^-$	(1.93, 1.62, 1.5)	1.68	(1.66, 2.6, 2.28)	2.18
4	16-24-47	28 $^+$	32 $^+$	3 $^-$	1 $^-$	100 $^+$	(2.33, 2.28, 2.18)	2.26	(5.82, 7.53, 7.76)	7.04
5	9-21-42	8 $^-$	12 $^-$	6 $^+$	1 $^-$	3.4 $^-$	(1.53, 1.53, 1.48)	1.51	(1.25, 1.25, 1.06)	1.19
6	10-23-33	28 $^+$	12 $^-$	6 $^+$	1 $^-$	100 $^+$	(1.95, 2.03, 2.08)	2.02	(6.16, 3.03, 3.19)	4.13
7	2-19-37	8 $^-$	32 $^+$	6 $^+$	1 $^-$	100 $^+$	(2.23, 2.33, 2.3)	2.29	(3.73, 3.71, 3.79)	3.74
8	8-17-39	28 $^+$	32 $^+$	6 $^+$	1 $^-$	3.4 $^-$	(1.35, 1.9, 1.47)	1.57	(8, 3.81, 6.73)	6.18
9	11-32-46	8 $^-$	12 $^-$	3 $^-$	7 $^+$	3.4 $^-$	(1.45, 1.58, 1.48)	1.5	(1.81, 1.99, 1.83)	1.88
10	13-18-38	28 $^+$	12 $^-$	3 $^-$	7 $^+$	100 $^+$	(2.27, 2.13, 2.15)	2.18	(5.91, 6.16, 5.49)	5.85
11	6-25-48	8 $^-$	32 $^+$	3 $^-$	7 $^+$	100 $^+$	(2.75, 3.27, 2.58)	2.87	(5.31, 5.08, 5.01)	5.13
12	3-26-41	28 $^+$	32 $^+$	3 $^-$	7 $^+$	3.4 $^-$	(1.52, 1.83, 1.52)	1.62	(9.95, 7.32, 9.95)	9.07
13	4-20-40	8 $^-$	12 $^-$	6 $^+$	7 $^+$	100 $^+$	(3.2, 2.98, 2.77)	2.98	(2.73, 2.7, 2.82)	2.75
14	7-27-36	28 $^+$	12 $^-$	6 $^+$	7 $^+$	3.4 $^-$	(1.57, 1.62, 1.63)	1.61	(5.25, 5.45, 5.55)	5.42
15	14-30-45	8 $^-$	32 $^+$	6 $^+$	7 $^+$	3.4 $^-$	(1.48, 1.52, 1.65)	1.55	(4.68, 4.75, 4.71)	4.71
16	1-29-44	28 $^+$	32 $^+$	6 $^+$	7 $^+$	100 $^+$	(2.47, 2.32, 2.6)	2.46	(13.29, 13.41, 13.22)	13.31

Fonte: O autor (2020), com resultados de 48 experimentos no ambiente *Spark — Google Data Proc*

8.2.1 Análise de Variabilidade de Resíduos e Modelo de Tempo

Os resíduos evidenciaram independência de erros e normalidade, com teste de Shapiro e Wilk (1965) resultando em *p-value* $= 0.05056$. Porém, perceberam-se violações na homogeneidade de variância por níveis de fator, especialmente para C; além de distorções no gráfico de

resíduos *versus* valores ajustados. Com $\lambda = 0.22$, formalmente sugere-se uma transformação de potência de Box e Cox (1964) para escala raiz cúbica $y^* = \sqrt[3]{y}$. Todavia, como o valor posicionou-se entre as aproximações logarítmica e raiz cúbica $(0 < \lambda < 0.3)$, é facultada a escolha por aquela que permita melhor interpretação dos dados. A comparação entre os dois modelos mostrou diferença irrisória de 0.0027 sobre o coeficiente de determinação R^2. Portanto, em favor da conveniência interpretativa do modelo, optou-se pela transformação logarítmica $y_* = \log y$. A Figura 12 exibe os resíduos para o modelo ajustado sobre os dados transformados.

A Tabela 8 detalha o modelo linear para o tempo que explica 93.95% da variação na resposta de acordo com o coeficiente de determinação $R^2 = 0.9395$ e $R^2_{Adj} = 0.9112$, *F-statistic* $= 33.16$ em 15 e 32 graus de liberdade e *p-value* $= 2.979e^{-15}$. Os fatores significativos foram N, V e as interações de segunda ordem N:C, N:V e M:D. Os fatores C, M e D não apresentaram significância estatística no contexto investigado. A ordem relativa de significância estatística entre os fatores pode ser conferida no gráfico *half-normal* de efeitos da Figura 13a que, juntamente com a proporção de variação explicada (% V.E.), contribui para a escolha dos fatores que devem compor o modelo reduzido. Observe-se que a interação D:V não deve ser considerada como tendo significância prática devido à contribuição de apenas 0.82% sobre o modelo.

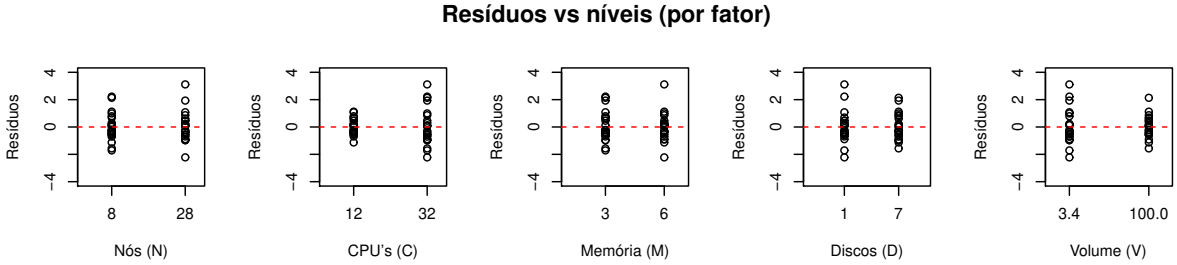
O tempo t_* pode ser estimado pela Equação 8.1, onde $\overline{n}, \overline{c}, \overline{m}, \overline{d}$ e $\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2.

$$\begin{aligned} t_*(N, C, M, D, V) = & 0.681609 - 0.050741\overline{n} + 0.229063\overline{v} + 0.027313\overline{nc} \\ & - 0.059908\overline{nv} + 0.043812\overline{md} \end{aligned} \tag{8.1}$$

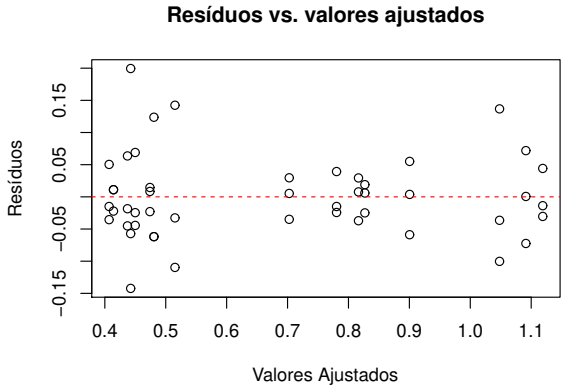
O efeito do número de nós é $E_N = e^{2 \times -0.050741} - 1 \approx -0.0965$. Ou seja, se apenas o número de nós variar de 8 para 28, o tempo de execução de LR diminuirá em 9.65%. Com efeito inverso, $E_V = e^{2 \times 0.229063} - 1 \approx 0.5811$, e implica que a variação isolada de volume entre 3.4% e 100% do total aumentará o tempo de execução em 58.11%, em média.

O impacto dos fatores principais pode ser facilmente compreendido através do gráfico da Figura 13b no qual percebe-se que a variação de memória (M) e discos (D) afeta de maneira irrisória o tempo de execução. Ademais, a variação no fator núcleos (C) evidencia impacto nulo sobre o resultado. Em 13c identifica-se um resultado não intuitivo devido à interação N:C — a redução de tempo é maior quando se aumenta o *cluster* através de nós com 12

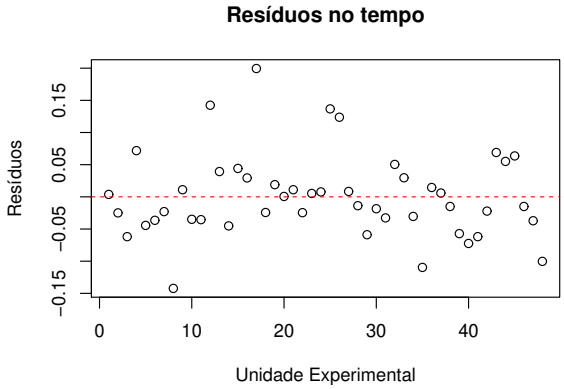
Figura 12 – Resíduos do modelo de tempo de LR em escala logarítmica



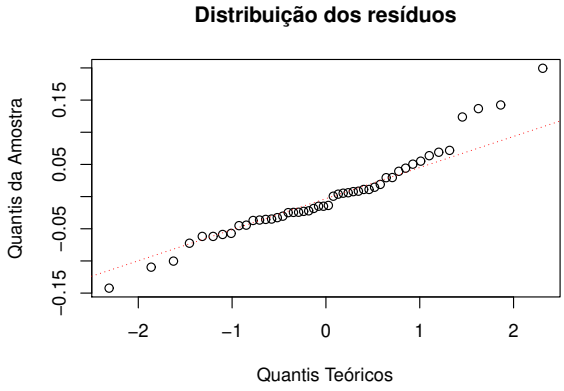
(a) Resíduos *vs.* níveis dos fatores N, C, M, D e V.



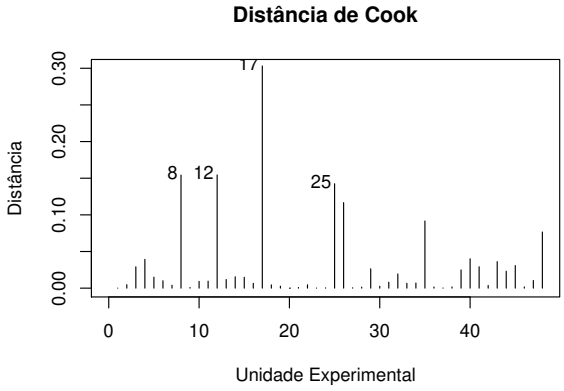
(b) Variância dos resíduos *vs.* valores ajustados.



(c) Resíduos em sequência temporal.



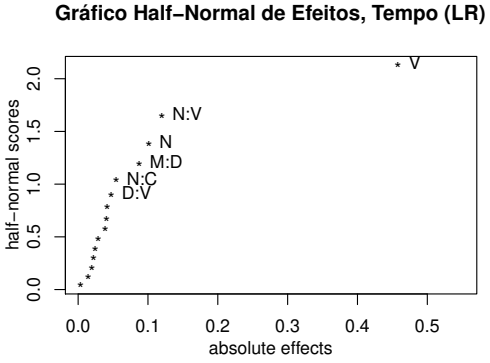
(d) Normalidade dos resíduos.



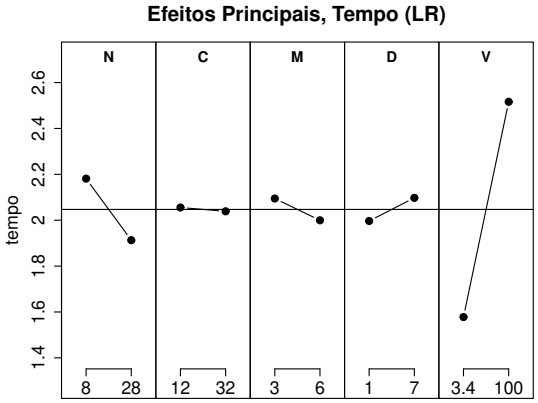
(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

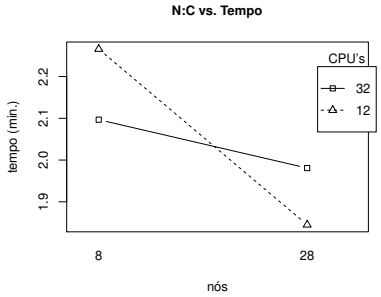
Figura 13 – Análise gráfica de modelo linear para tempo de LR



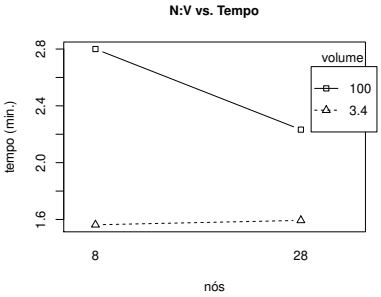
(a) Ranking de efeitos significativos.



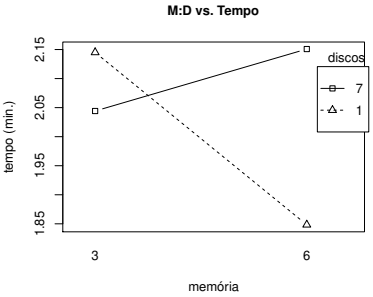
(b) Magnitude dos efeitos principais.



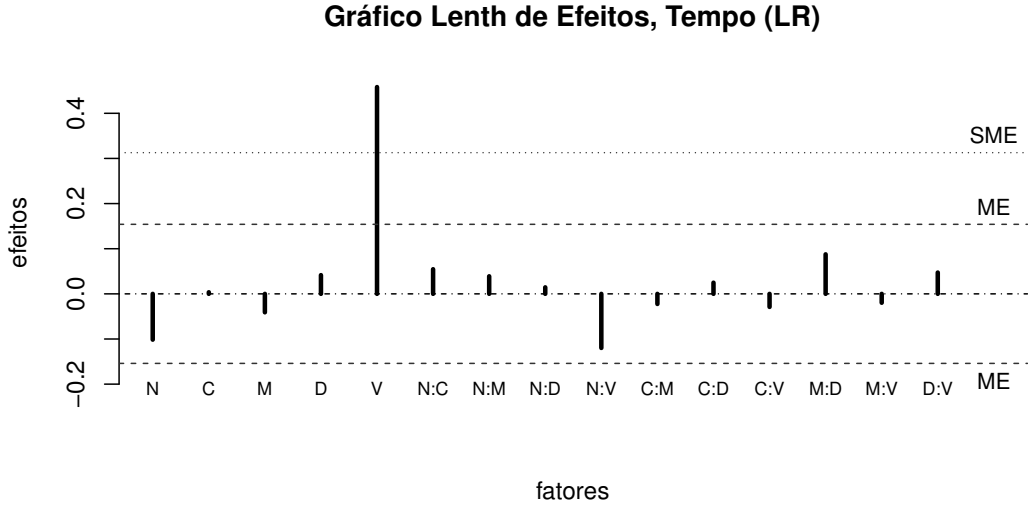
(c) Interação N:C.



(d) Interação N:V.



(e) Interação M:D.



(f) Efeitos do tempo de execução de LR.

Fonte: O autor (2020)

Tabela 8 – Modelo linear para tempo de execução de LR (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	0.681609	60.213	$< 2e^{-16}$***	- -
N	**-0.050741**	**-4.482**	**$8.88e^{-05}$ *** **	**3.80 %**
C	0.001788	0.158	0.875507	0.00 %
M	-0.020457	-1.807	0.080142	0.62 %
D	0.020772	1.835	0.075817	0.64 %
V	**0.229063**	**20.235**	**$< 2e^{-16}$ *** **	**77.36 %**
N:C	**0.027313**	**2.413**	**0.021735 ***	**1.10 %**
N:M	0.019492	1.722	0.094739	0.56 %
N:D	0.007234	0.639	0.527362	0.08 %
N:V	**-0.059908**	**-5.292**	**$8.49e^{-06}$ *** **	**5.29 %**
C:M	-0.011204	-0.990	0.329720	0.19 %
C:D	0.012393	1.095	0.281779	0.23 %
C:V	-0.014491	-1.280	0.209714	0.31 %
M:D	**0.043812**	**3.870**	**0.000503 *** **	**2.83 %**
M:V	-0.009862	-0.871	0.390147	0.14 %
D:V	0.023620	2.087	0.044976 *	0.82 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

núcleos, em detrimento de nós com 32 núcleos. A interação N:V (Figura 13d) sugere que não há ganho no tempo de processamento ao aumentar o número de nós do *cluster* para volumes de dados de entrada no nível 3.4%, fazendo sentido apenas para volumes maiores. Por fim, outro resultado não intuitivo observa-se na interação M:D (Figura 13e): aumentar memória só reduz o tempo quando há poucos discos; enquanto o aumento de memória em nós com sete discos degrada o desempenho. Finalmente, o gráfico de *Lenth* detalha os efeitos para todos os fatores e interações de segunda ordem do modelo.

A Equação 8.1 explica 91.63% da variação na resposta ($R^2 = 0.9163$ e $R^2_{Adj} = 0.8991$) e estima o tempo de execução de LR para configurações de *cluster* entre 8 e 28 nós, 12 a 32 núcleos por nó, com fator de memória entre 3x e 6x, 1 a 8 discos e percentuais a partir de 3.4% do volume de entrada de dados.

8.2.2 Análise de Variabilidade de Resíduos e Modelo de Custo

Os resíduos mostraram heterogeneidade de variância para os níveis de fatores, especialmente para N, M e D. Apresentaram ainda viés à esquerda e à direita no gráfico de quantis. O cálculo de $\lambda = -0.18$ sugeriu transformação de potência de Box e Cox (1964) para a escala logarítmica. A Figura 14 exibe os resíduos para o modelo ajustado sobre os dados transformados, para os quais passou-se a observar homogeneidade de variância (Figuras 14a e 14b), independência de erros (Figura 14c), e distribuição de quantis próximos a uma distribuição Normal (Figura 14d).

A Tabela 9 detalha o modelo linear para o custo que explica 94.87% da variação na resposta de acordo com o coeficiente de determinação $R^2 = 0.9487$ e $R^2_{Adj} = 0.9247$, *F-statistic* = 39.49 em 15 e 32 graus de liberdade e *p-value* = $2.254e^{-16}$. Apenas os fatores principais apresentaram significância estatística, podendo o modelo pode ser analisado sem efeitos de interações. O custo c_* é estimado pela Equação reduzida 8.2, onde $\overline{n}, \overline{c}, \overline{m}, \overline{d}$ e $\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2.

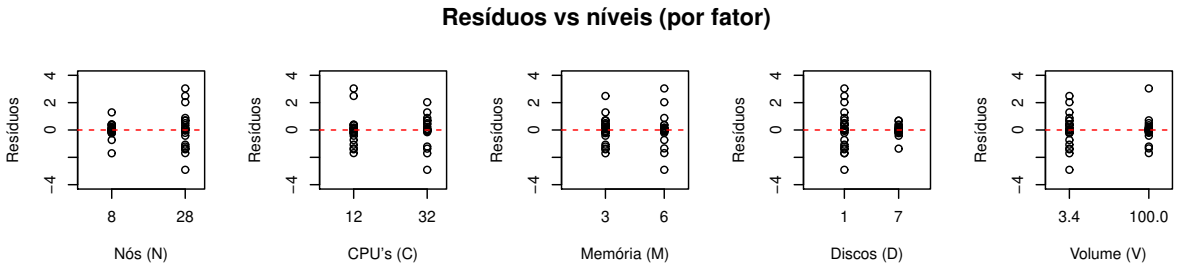
Tabela 9 – Modelo linear para custo de execução de LR (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	1.361578	51.334	< 2e-16 ***	- -
N	**0.433848**	**16.357**	**< 2e^{-16} *** **	**42.86 %**
C	**0.359551**	**13.556**	**8.27e^{-15} *** **	**29.43 %**
M	**0.071441**	**2.693**	**0.0112 * **	**1.16 %**
D	**0.271477**	**10.235**	**1.28e^{-11} *** **	**16.78 %**
V	**0.134913**	**5.087**	**1.55e^{-05} *** **	**4.14 %**
N:C	-0.029700	-1.120	0.2711	0.20 %
N:M	-0.013310	-0.502	0.6192	0.04 %
N:D	-0.007254	-0.273	0.7862	0.01 %
N:V	-0.015487	-0.584	0.5634	0.05 %
C:M	0.015706	0.592	0.5579	0.06 %
C:D	-0.000276	-0.010	0.9918	0.00 %
C:V	0.015630	0.589	0.5598	0.06 %
M:D	0.005320	0.201	0.8423	0.01 %
M:V	0.002525	0.095	0.9247	0.00 %
D:V	-0.017716	-0.668	0.5090	0.07 %

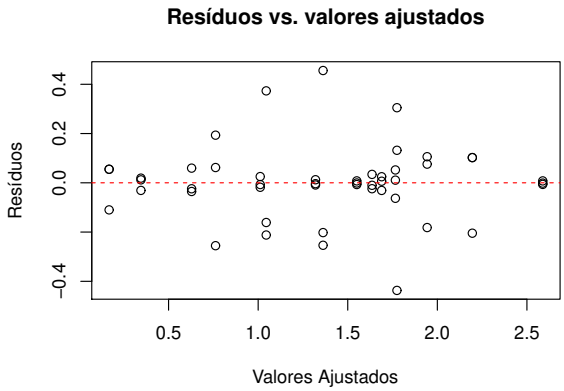
a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

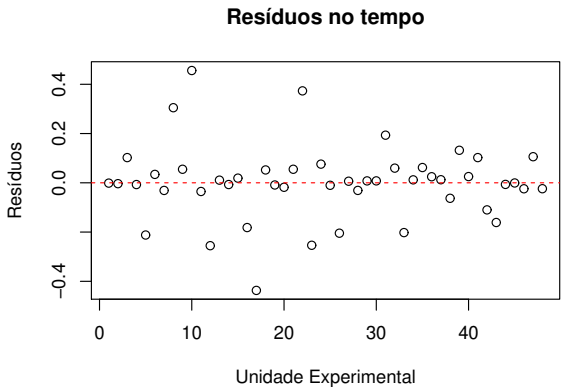
Figura 14 – Resíduos do modelo de custo de LR em escala logarítmica



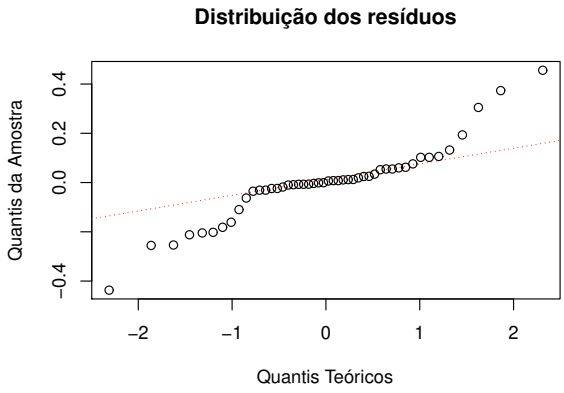
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



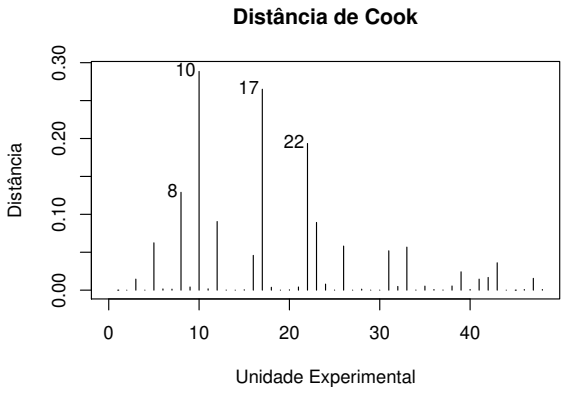
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

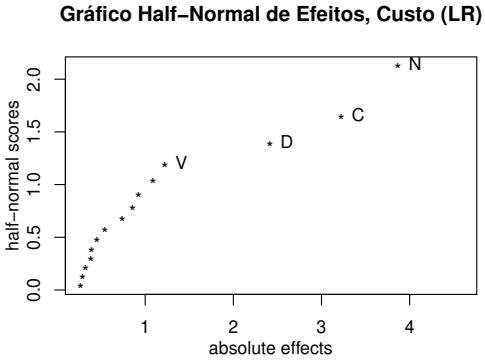
$$c_*(N,C,M,D,V) = 1.361578 + 0.433848\overline{n} + 0.359551\overline{c} + 0.071441\overline{m} + 0.271477\overline{d} + 0.134913\overline{v} \tag{8.2}$$

O efeito do número de nós sobre o custo é $E_N = e^{2 \times 0.433848} - 1 \approx 1.3815$. Isso significa que, fixados os demais fatores, se o número de nós aumenta de 8 para 28, o custo de execução de LR aumentará cerca de 138.15%, em média. Da mesma forma, $E_C = e^{2 \times 0.359551} - 1 \approx 1.0526$ ou efeito de variação de 105.26%. $E_M = e^{2 \times 0.071441} - 1 \approx 0.1536 = 15.36\%$. Discos produzem um impacto isolado $E_D = e^{2 \times 0.271477} - 1 \approx 0.7211 = 72.11\%$. Por fim, a variação no volume de dados de entrada de 3.4% para 100% causa um impacto $E_V = e^{2 \times 0.134913} - 1 \approx 0.3097$ ou 30.97%.

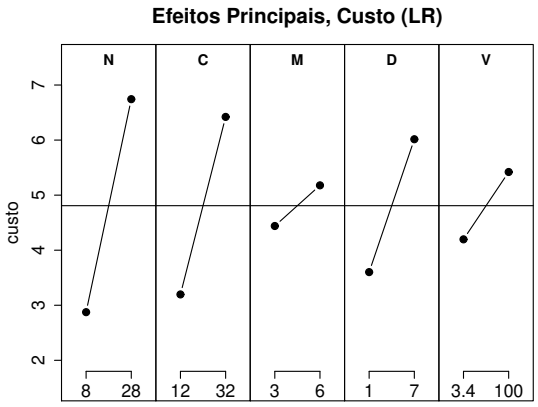
A Figura 15b mostra que todos os fatores principais influenciam o custo de LR, com destaque para N e C. As interações não são estatisticamente significativas. O gráfico de *Lenth* (Figura 15c) detalha todos os efeitos, onde percebe-se o impacto nulo das interações de segunda ordem, ou seja, o custo de execução se explica pelos fatores principais isoladamente, com participação minoritária dos fatores memória M e V.

O modelo reduzido da Equação 8.2 explica 94.38% da variação no custo ($R^2 = 0.9438$ e $R^2_{Adj} = 0.9371$) e pode ser utilizado para estimar o custo de execução de LR para configurações de *cluster* de 8 a 28 nós, 12 a 32 núcleos por nó, com fator de memória entre 3x e 6x, 1 a 7 discos e qualquer percentual a partir de 3.4% do volume de dados de entrada.

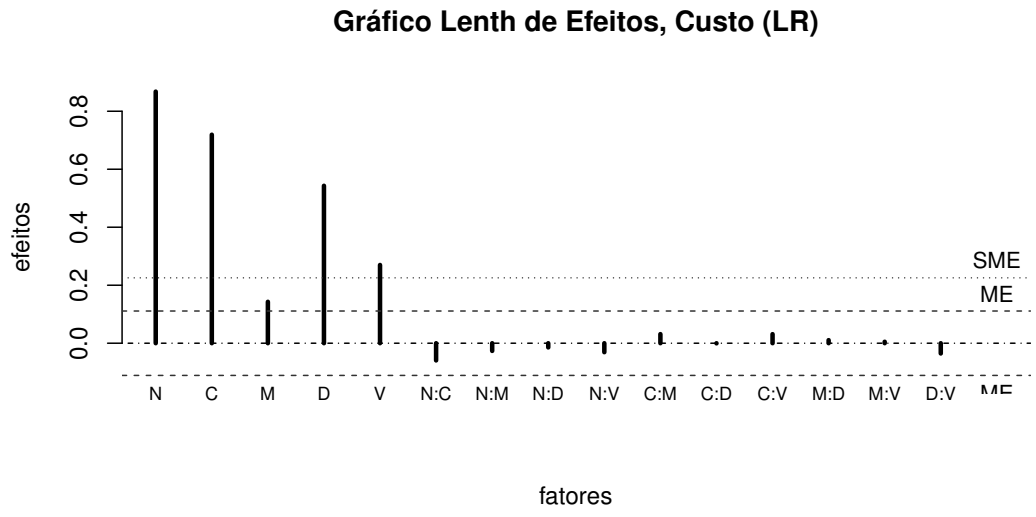
Figura 15 – Análise gráfica de modelo linear para custo de LR



(a) Ranking de efeitos significativos.



(b) Magnitude dos efeitos principais.



(c) Efeitos do custo de execução de LR.

Fonte: O autor (2020)

8.3 PROJETO EXPERIMENTAL: MÁQUINAS DE VETORES DE SUPORTE (SVM)

O Quadro 11 exibe os hiperparâmetros utilizados no treinamento, enquanto a Tabela 10 detalha o projeto experimental empregado no estudo.

Quadro 11 – Hiperparâmetros para treinamento da técnica SVM

Hiperparâmetro	**Descrição**	**Valor**
Parâmetro *aggregation depth*	`svm.setAggregationDepth`	2
Regularização	`svm.setRegParam`	0
Máximo de iterações	`svm.setMaxIter`	25
Tolerância de convergência	`svm.setTol`	10^{-6}

Fonte: O autor (2020)

Tabela 10 – Projeto experimental $2_V^{5-1} \times 3$ para tempo e custo de SVM

#	Plano	N	C	M	D	V	t_1, t_2, t_3 (min.)	$\bar{t}$	c_1, c_2, c_3 ($)	$\bar{c}$
1	5-21-44	8 $^-$	12 $^-$	3 $^-$	1 $^-$	100 $^+$	(20.33, 20.75, 20.45)	20.51	(8.33, 8.54, 8.39)	8.42
2	14-25-42	28 $^+$	12 $^-$	3 $^-$	1 $^-$	3.4 $^-$	(10.78, 8.37, 8.73)	9.29	(11.28, 9.35, 9.72)	10.12
3	8-29-48	8 $^-$	32 $^+$	3 $^-$	1 $^-$	3.4 $^-$	(9.22, 8.82, 9.23)	9.09	(10.44, 10.09, 8.25)	9.59
4	15-31-35	28 $^+$	32 $^+$	3 $^-$	1 $^-$	100 $^+$	(15.8, 16.02, 15.82)	15.88	(40.28, 40.23, 40.31)	40.27
5	13-19-34	8 $^-$	12 $^-$	6 $^+$	1 $^-$	3.4 $^-$	(8.4, 8.07, 8.92)	8.46	(4.51, 4.40, 4.70)	4.54
6	7-23-39	28 $^+$	12 $^-$	6 $^+$	1 $^-$	100 $^+$	(11.53, 12.18, 11.6)	11.77	(14.11, 15.16, 14.12)	14.46
6	4-32-41	8 $^-$	32 $^+$	6 $^+$	1 $^-$	100 $^+$	(20.13, 18.37, 14.27)	17.59	(24.29, 22.6, 18.04)	21.64
8	9-24-36	28 $^+$	32 $^+$	6 $^+$	1 $^-$	3.4 $^-$	(6.4, 6.27, 6.03)	6.23	(21.37, 22.21, 21.34)	21.64
9	1-27-43	8 $^-$	12 $^-$	3 $^-$	7 $^+$	3.4 $^-$	(7.95, 9.73, 10.55)	9.41	(5.71, 6.61, 7.02)	6.45
10	6-28-33	28 $^+$	12 $^-$	3 $^-$	7 $^+$	100 $^+$	(12, 16.8, 11.87)	13.56	(20.72, 26.4, 20.39)	22.5
11	16-20-47	8 $^-$	32 $^+$	3 $^-$	7 $^+$	100 $^+$	(19.72, 20.93, 14.87)	18.51	(24.44, 26.48, 20.04)	23.65
12	11-17-40	28 $^+$	32 $^+$	3 $^-$	7 $^+$	3.4 $^-$	(5.63, 6.52, 5.6)	5.92	(23.88, 25.2, 23.92)	24.33
13	12-30-37	8 $^-$	12 $^-$	6 $^+$	7 $^+$	100 $^+$	(19.95, 22.2, 20.4)	20.85	(13.49, 14.62, 13.71)	13.94
14	3-26-38	28 $^+$	12 $^-$	6 $^+$	7 $^+$	3.4 $^-$	(10.18, 9.28, 8.03)	9.16	(17.31, 18.46, 16.58)	17.45
15	10-18-46	8 $^-$	32 $^+$	6 $^+$	7 $^+$	3.4 $^-$	(7.95, 9.28, 9.02)	8.75	(13.75, 15.63, 15.8)	15.06
16	2-22-45	28 $^+$	32 $^+$	6 $^+$	7 $^+$	100 $^+$	(16.12, 19.37, 14.13)	16.54	(58.89, 67.86, 53.11)	59.95

Fonte: O autor (2020), com resultados de 48 experimentos no ambiente *Spark — Google Data Proc*

8.3.1 Análise de Variabilidade de Resíduos e Modelo de Tempo

A análise gráfica dos resíduos indicou heterogeneidade de variância, especialmente em relação aos fatores C e V, além de desvio padrão de 2σ na distância de *Cook* para as unidades experimentais nº 28, 41 e 47. O teste de Shapiro e Wilk (1965) evidenciou ausência de normalidade com *p-value* $= 0.021622$. O cálculo $\lambda = -0.50$ sugeriu transformação de potência de Box e Cox (1964) para escala raiz quadrada recíproca $y^* = -\frac{1}{\sqrt{y}}$.

A Figura 16 exibe os gráficos de resíduos para o modelo ajustado sobre os dados transformados. Em 16a os resíduos se espalham com homogeneidade entre os níveis. O gráfico da Figura 16b também não evidencia problemas nos resíduos em face dos valores ajustados. Na Figura 16c os resíduos na sequência do tempo sugerem aleatoriedade e independência. Estes ainda passaram a se aproximar de uma distribuição Normal, o que pode ser verificado através do gráfico quantil-quantil da Figura 16d e do teste de Shapiro e Wilk (1965) com *p-value* $= 0.690998$. Apenas as unidades experimentais nº 1 e 28 apresentaram-se como pontos discrepantes (Figura 16e), todavia com distância de *Cook* diminuída em relação aos dados originais, resultando em menor distorção do modelo.

A Tabela 11 detalha o modelo que explica 95.28% da variação para o tempo de execução de acordo com o coeficiente de determinação $R^2 = 0.9528$ e $R^2_* = 0.9307$, *F-statistic* = 43.08 em 15 e 32 graus de liberdade e *p-value* $< 2.2e^{-16}$. Os fatores significativos foram N, C, V, e as interações de segunda ordem C:V e M:D. A quantidade de memória e o número de discos SSD não produzem impacto sobre o desempenho, já que seus níveis de significância são maiores que 0.05 com proporção de variação explicada abaixo de 1%.

Tabela 11 – Modelo linear para tempo de execução de SVM (escala raiz quadrada recíproca)

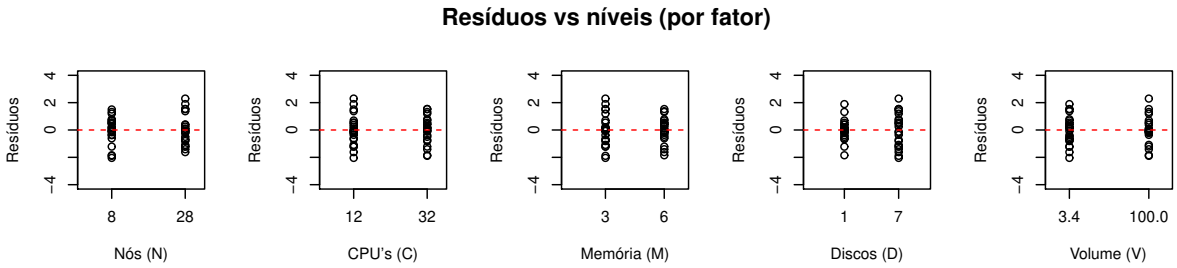
	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	-0.299658	-128.431	$< 2e^{-16}$ ***	- -
N	**-0.017571**	**-7.531**	**$1.42e^{-08}$ *** **	**8.36 %**
C	**-0.007334**	**-3.143**	**0.00359 ****	**1.46 %**
M	-0.001994	-0.854	0.39923	0.11 %
D	0.001529	0.655	0.51696	0.06 %
V	**0.052296**	**22.414**	**$< 2e^{-16}$*** **	**74.08%**
N:C	-0.003161	-1.355	0.18492	0.27 %
N:M	0.001438	0.616	0.54215	0.06 %
N:D	-0.000588	-0.252	0.80268	0.01 %
N:V	-0.001125	-0.482	0.63288	0.03 %
C:M	0.002235	0.958	0.34531	0.14 %
C:D	-0.002617	-1.122	0.27035	0.19 %
C:V	**0.011432**	**4.900**	**$2.66e^{-05}$*** **	**3.54%**
M:D	**0.015930**	**6.827**	**$1.01e^{-07}$*** **	**6.87%**
M:V	-0.000262	-0.112	0.91145	0.00 %
D:V	0.002013	0.863	0.39463	0.11 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

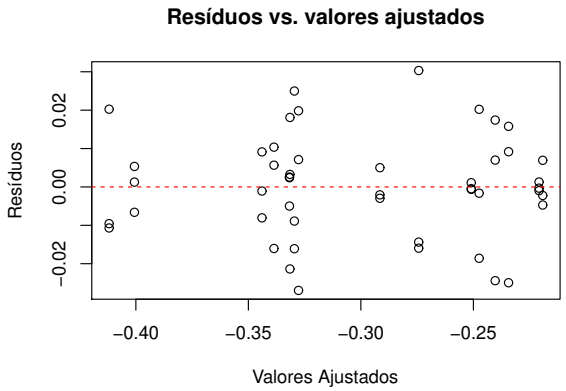
b – V.E.: proporção de variação explicada pelo fator.

O modelo reduzido é expresso pela Equação 8.3, onde $\overline{n}, \overline{c}, \overline{m}, \overline{d}$ e $\overline{v}$ estão em conformidade

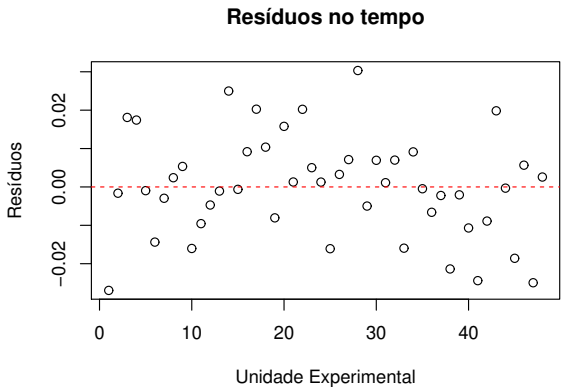
Figura 16 – Resíduos do modelo de tempo de SVM em escala raiz quadrada recíproca



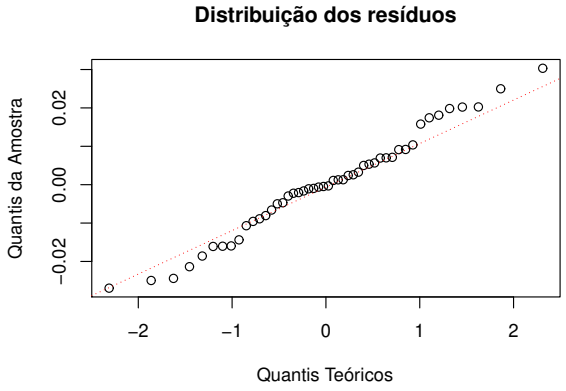
(a) Resíduos *vs.* níveis dos fatores N, C, M, D e V.



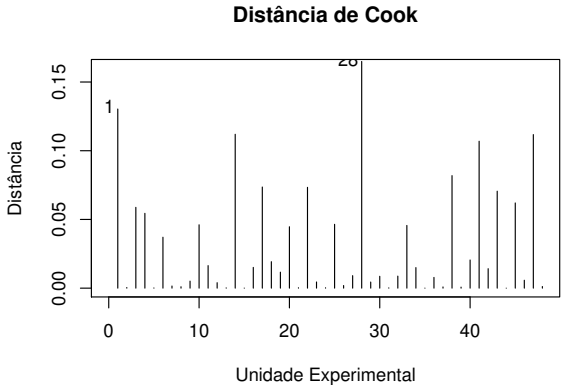
(b) Variância dos resíduos *vs.* valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

com as conversões do Quadro 9, Seção 8.1.2.

$$t_*(N,C,M,D,V) = -0.299658 - 0.017571\,\overline{n} - 0.007334\,\overline{c} + 0.052296\,\overline{v} + 0.011432\,\overline{cv} + 0.015930\,\overline{md} \tag{8.3}$$

Os coeficientes parciais de regressão encontram-se em escala raiz quadrada recíproca, tornando não trivial a interpretação dos efeitos (Seção 5.5.1). Portanto, não se aplicam os gráficos de efeitos principais, interações, *half-normal* e *Lenth*. Entretanto, o modelo continua possuindo aplicabilidade prática para predição. A operação $t = \frac{1}{(t_*)^2}$ deve ser aplicada sobre a resposta para obtenção do valor em escala original.

O modelo reduzido da Equação 8.3 explica 94.48% da variação no custo ($R^2 = 0.9448$ e $R^2_{Adj} = 0.9351$) e pode ser utilizado para estimar o tempo de execução de SVM para configuração de *cluster* entre 8 e 28 nós, 12 a 32 núcleos por nó, com fator de memória entre 3x e 6x, 1 a 7 discos e qualquer percentual a partir de 3.4% do volume de dados.

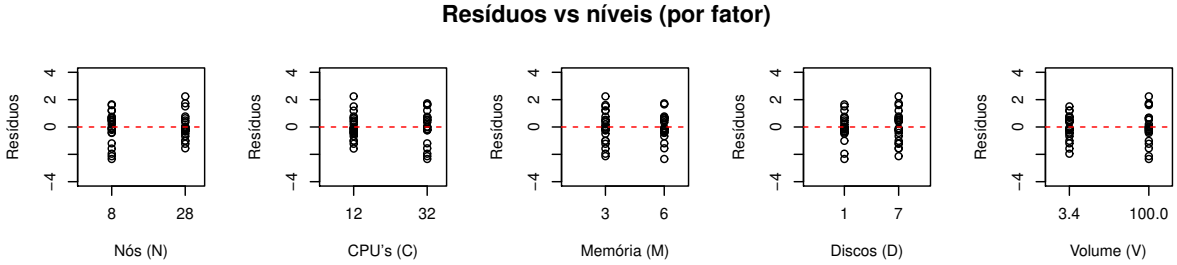
8.3.2 Análise de Variabilidade de Resíduos e Modelo de Custo

A análise dos resíduos evidenciou heterogeneidade de variância e o valor $\lambda = -0.18$ sugeriu transformação de potência de Box e Cox (1964) para a escala logarítmica. A Figura 17 exibe os resíduos para o modelo ajustado sobre os dados transformados que passaram a apresentar homogeneidade de variância (Figuras 17a e 17b) e independência de erros (Figura 17c). A Figura 17d sugere normalidade dos resíduos, confirmada pelo teste de Shapiro e Wilk (1965) com *p-value* $= 0.462390$. Por fim, apenas três pontos foram evidenciados como discrepantes com desvio $< 2\sigma$ no gráfico de *Cook* da Figura 17e.

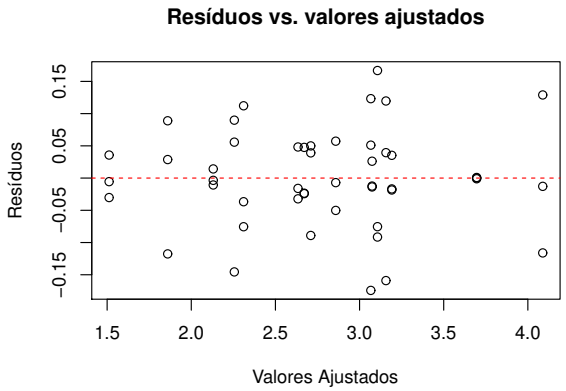
A Tabela 12 detalha o modelo linear para o custo que explica 98.66% da variação na resposta de acordo com os coeficientes de determinação $R^2 = 0.9866$ e $R^2_* = 0.9804$, *F-statistic* $= 157.4$ em 15 e 32 graus de liberdade e *p-value* $< 2.2e^{-16}$. Os fatores significativos foram N, C, D, V, e a interação M:D. Embora significativo, o fator M isoladamente não deve ser considerado como tendo significância prática, devido à contribuição de apenas 0.78% sobre o modelo. Raciocínio análogo se aplica às interações N:V, C:D e C:V.

O modelo reduzido para o custo c_* pode ser expresso pela Equação 8.4, onde $\overline{n}, \overline{c}, \overline{m}, \overline{d}$ e

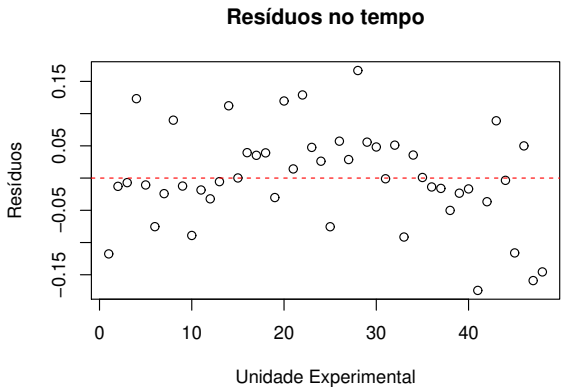
Figura 17 – Resíduos do modelo de custo de SVM em escala logarítmica



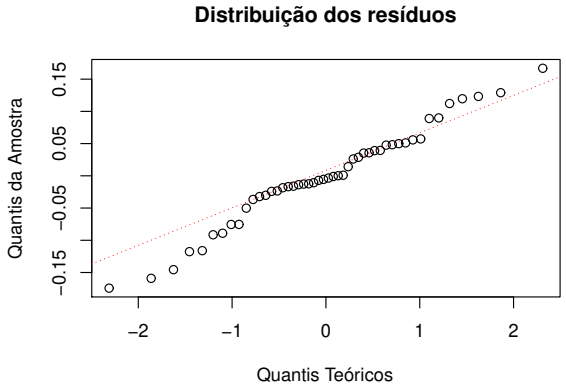
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



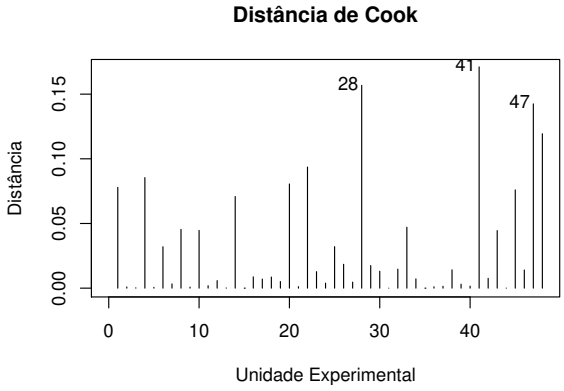
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

Tabela 12 – Modelo linear para custo de execução de SVM em escala logarítmica

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.770174	210.376	$<$ 2e-16 ***	–
N	**0.354434**	**26.917**	**$<$ 2e-16*** **	**30.27 %**
C	**0.384779**	**29.221**	**$<$ 2e-16*** **	**35.67 %**
M	0.056726	4.308	0.000146 ***	0.78 %
D	**0.180547**	**13.711**	**6.05e^{-15}*** **	**7.85 %**
V	**0.298588**	**22.676**	**$<$ 2e^{-16}*** **	**21.48 %**
N:C	0.003136	0.238	0.813301	0.00 %
N:M	-0.008266	-0.628	0.534644	0.02 %
N:D	0.006069	0.461	0.648009	0.01 %
N:V	-0.032763	-2.488	0.018241 *	0.26 %
C:M	0.023275	1.768	0.086664	0.13 %
C:D	-0.048777	-3.704	0.000798 ***	0.57 %
C:V	0.048418	3.677	0.000860 ***	0.56 %
M:D	**0.065326**	**4.961**	**2.23e^{-05}*** **	**1.03 %**
M:V	-0.010347	-0.786	0.437793	0.03 %
D:V	-0.002825	-0.215	0.831489	0.00 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

$\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2.

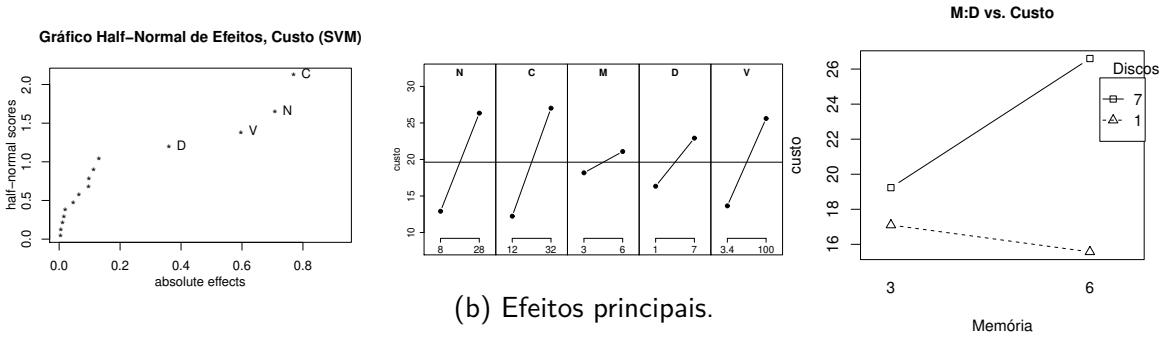
$$c_*(N,C,M,D,V) = 2.770174 + 0.354434\,\overline{n} + 0.384779\,\overline{c} + 0.180547d + 0.298588\overline{v} + 0.065326\,\overline{md} \tag{8.4}$$

O efeito do fator N sobre o tempo é $E_N = e^{2\times 0.354434} - 1 \approx 1.0317$, ou seja, fixados todos os fatores, se o número de nós variar de 8 para 28, o custo de execução do algoritmo SVM aumentará em 103.17%. De maneira análoga, $E_C = e^{2\times 0.384779} - 1 \approx 1.1588 = +115.88\%$, $E_D = e^{2\times 0.180547} - 1 \approx 0.4348 = 43.5\%$ e $E_V = e^{2\times 0.298588} - 1 \approx 0.8169 = 81.69\%$.

O efeito de D não é absoluto, uma vez que existe a interação M:D [3] e pode ser melhor compreendido pela análise da Figura 18c. O **aumento** de memória **diminui** sensivelmente o custo quando o sistema computacional tem apenas um disco em cada nó. Contrariamente, o aumento de memória nos níveis máximos de disco incorrem em expressivo custo. Embora o

[3] Mesmo não sendo um fator significativo em isolado, M produz um efeito sobre a resposta que só surge em combinação com os efeitos de D.

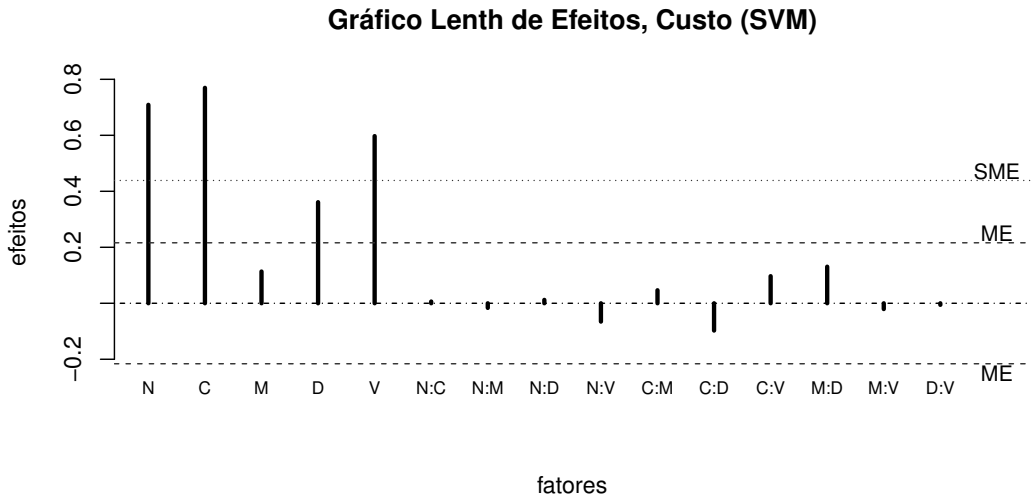
Figura 18 – Análise gráfica de modelo linear para custo de SVM



(a) Ranking de efeitos.

(b) Efeitos principais.

(c) Efeitos da interação N:V.



(d) Efeitos do custo de execução de SVM.

Fonte: O autor (2020)

último fato seja intuitivo, o primeiro não o é, sou seja: aumentar memória nos níveis mínimos de configuração produz redução de custo.

A Equação 8.4 explica 97.08% da variação na resposta ($R^2 = 0.9708$ e $R^2_{Adj} = 0.9665$) e estima o custo de execução do algoritmo SVM para qualquer variação de configuração de *cluster* nas condições investigadas.

8.4 PROJETO EXPERIMENTAL: *NAÏVE BAYES* (NB)

O Quadro 12 exibe os hiperparâmetros utilizados no treinamento, enquanto a Tabela 13 detalha o projeto experimental empregado no estudo.

Quadro 12 – Hiperparâmetros para treinamento da técnica NB

Hiperparâmetro	**Descrição**	**Valor**
Tipo	`nb.setModelType`	`multinomial`
Suavização	`nb.setSmoothing`	1.0

Fonte: O autor (2020)

Tabela 13 – Projeto experimental $2_V^{5-1} \times 3$ para tempo e custo de NB

#	Plano	N	C	M	D	V	t_1, t_2, t_3 (min.)	$\bar{t}$	c_1, c_2, c_3 ($)	$\bar{c}$
1	1-29-43	8 $^-$	12 $^-$	3 $^-$	1 $^-$	100 $^+$	(4.9, 4.5, 4.53)	4.64	(2.52, 2.33, 2.36)	2.4
2	5-28-47	28 $^+$	12 $^-$	3 $^-$	1 $^-$	3.4 $^-$	(1.92, 1.85, 2.15)	1.97	(3.37, 3.26, 4.24)	3.62
3	8-18-44	8 $^-$	32 $^+$	3 $^-$	1 $^-$	3.4 $^-$	(2.03, 2.3, 2.12)	2.15	(3.63, 3.81, 3.7)	3.71
4	16-32-45	28 $^+$	32 $^+$	3 $^-$	1 $^-$	100 $^+$	(3.25, 3.62, 3.37)	3.41	(11.31, 12.27, 11.56)	11.71
5	12-30-38	8 $^-$	12 $^-$	6 $^+$	1 $^-$	3.4 $^-$	(3.17, 3.17, 3.15)	3.16	(2.17, 2.2, 2.15)	2.17
6	13-26-36	28 $^+$	12 $^-$	6 $^+$	1 $^-$	100 $^+$	(3.12, 3.03, 3.57)	3.24	(5.2, 5.17, 5.56)	5.31
7	14-22-41	8 $^-$	32 $^+$	6 $^+$	1 $^-$	100 $^+$	(3.77, 3.77, 3.58)	3.71	(6.37, 6.22, 6.09)	6.23
8	10-31-48	28 $^+$	32 $^+$	6 $^+$	1 $^-$	3.4 $^-$	(1.78, 1.77, 1.85)	1.80	(9.44, 10.19, 10.7)	10.11
9	3-20-34	8 $^-$	12 $^-$	3 $^-$	7 $^+$	3.4 $^-$	(3.38, 3.32, 3.3)	3.33	(3.34, 3.43, 3.95)	3.57
10	9-25-35	28 $^+$	12 $^-$	3 $^-$	7 $^+$	100 $^+$	(3, 3.07, 3.1)	3.06	(8.42, 8.42, 8.41)	8.42
11	6-24-40	8 $^-$	32 $^+$	3 $^-$	7 $^+$	100 $^+$	(3.75, 4.1, 3.83)	3.89	(7.65, 7.69, 7.36)	7.57
12	15-19-37	28 $^+$	32 $^+$	3 $^-$	7 $^+$	3.4 $^-$	(1.77, 1.77, 1.75)	1.76	(13.17, 13.58, 13.1)	13.28
13	7-23-42	8 $^-$	12 $^-$	6 $^+$	7 $^+$	100 $^+$	(4.1, 4.67, 4.33)	4.37	(4.23, 4.64, 4.34)	4.4
14	11-27-33	28 $^+$	12 $^-$	6 $^+$	7 $^+$	3.4 $^-$	(1.78, 2.05, 1.9)	1.91	(7.34, 7.64, 7.56)	7.51
15	4-21-46	8 $^-$	32 $^+$	6 $^+$	7 $^+$	3.4 $^-$	(2.15, 2.12, 2.15)	2.14	(7.04, 6.79, 6.87)	6.9
16	2-17-39	28 $^+$	32 $^+$	6 $^+$	7 $^+$	100 $^+$	(3.13, 3.52, 3.57)	3.41	(19.92, 20.29, 20.25)	20.15

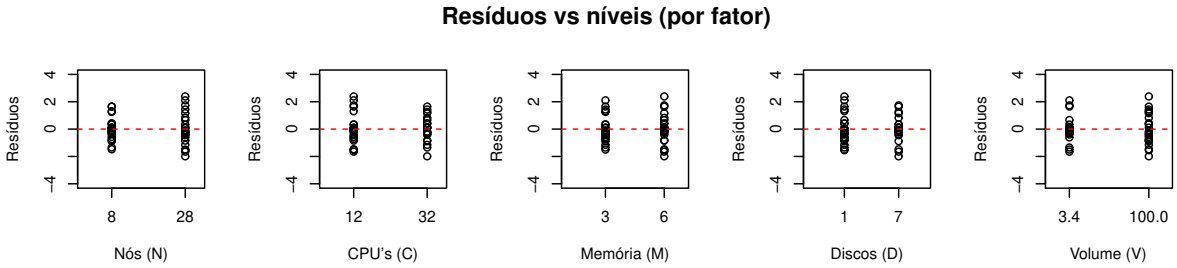
Fonte: O autor (2020), com resultados de 48 experimentos no ambiente *Spark — Google Data Proc*

8.4.1 Análise de Variabilidade de Resíduos e Modelo de Tempo

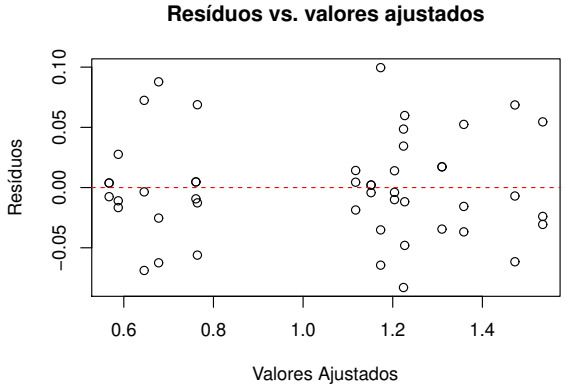
Os gráficos de resíduos do modelo sobre os dados originais evidenciaram heterogeneidade de variância entre os níveis dos fatores, especialmente para C, M e V. Também percebeu-se distorção no gráfico de resíduos *vs.* valores ajustados. Não foram observados problemas de aleatoriedade de erros ou efeito severo de pontos discrepantes. O teste de Shapiro e Wilk (1965) evidenciou normalidade dos resíduos tanto em escala original (*p-value* $= 0.177393$) quanto em escala transformada (*p-value* $= 0.222483$). Devido ao valor $\lambda = -0.10$ optou-se

por uma transformação de potência de Box e Cox (1964) para a escala logarítmica $y^* = \log y$. Notou-se então diminuição significativa da heterogeneidade de variância entre os níveis de fatores (Figura 19a) e correção da distorção dos resíduos *vs.* valores ajustados (Figura 19b). Não houve impacto na aleatoriedade de erros, na normalidade dos resíduos ou na influência de pontos discrepantes, conforme Figuras 19c, 19d e 19e, respectivamente.

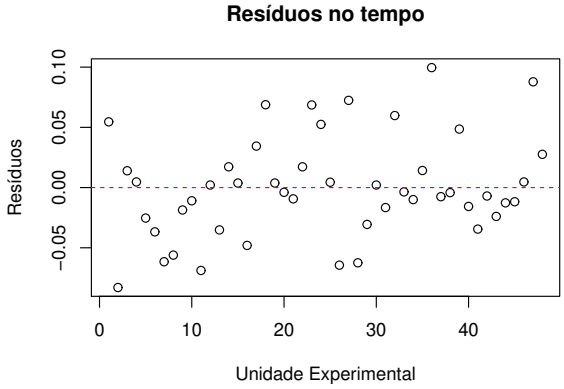
Figura 19 – Resíduos do modelo de tempo de NB em escala logarítmica



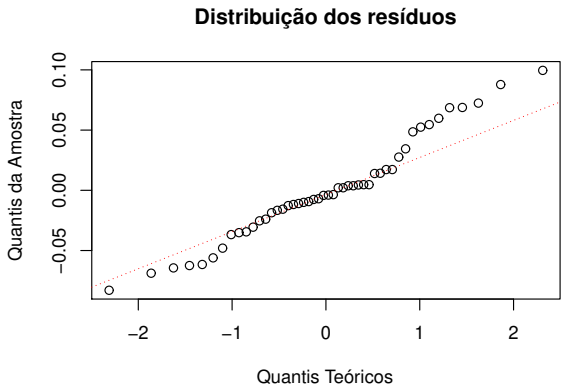
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



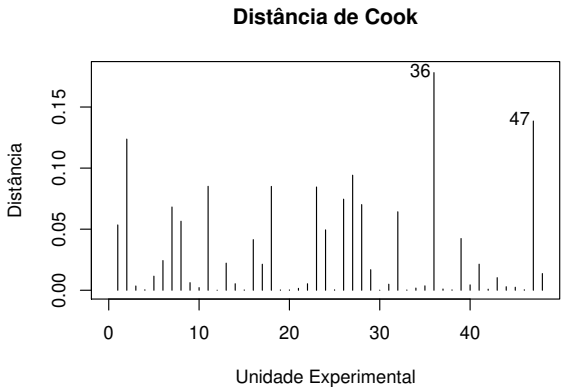
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

O modelo linear da Tabela 19 explica 98.3% da variação sobre o tempo de NB $R^2 = 0.983$ e $R^2_{Adj} = 0.9830$, *F-statistic* = 123.1 em 15 e 32 graus de liberdade e *p-value* $< 2.2e^{-16}$. Os fatores estatisticamente significativos com %V.E $> 1\%$ são N, C, V e as interações N:C e C:V. O *ranking* dos efeitos pode ser conferido no gráfico *half-normal* da Figura 20a. Os fatores M e D não mostraram relevância para o modelo, enquanto a interação N:V, embora estatisticamente significativa, contribui com apenas 0.84% sobre o resultado, podendo ser desconsiderada para propósitos práticos.

Tabela 14 – Modelo linear para tempo de execução de NB (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	1.048429	142.212	$< 2e^{-16}$ - -	- -
N	**-0.146079**	**-19.815**	**$< 2^{-16}$ *****	**20.90 %**
C	**-0.073584**	**-9.981**	**2.37^{-11} *****	**5.30 %**
M	-0.007823	-1.061	0.296573	0.06 %
D	-0.004718	-0.640	0.526750	0.02 %
V	**0.253638**	**34.404**	**$< 2e^{-16}$ *****	**63.00 %**
N:C	**0.072605**	**9.848**	**3.29^{-11} *****	**5.16 %**
N:M	0.012982	1.761	0.087804	0.17 %
N:D	-0.009162	-1.243	0.222972	0.08 %
N:V	0.029241	3.966	0.000385 ***	0.84 %
C:M	0.003536	0.480	0.634730	0.01 %
C:D	0.007505	1.018	0.316281	0.06 %
C:V	**0.051285**	**6.956**	**7.03^{-08} *****	**2.58 %**
M:D	-0.010186	-1.382	0.176665	0.10 %
M:V	0.000609	0.083	0.934648	0.00 %
D:V	-0.004260	-0.578	0.567397	0.02 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

A Equação 8.5 expressa o modelo reduzido para o tempo t_*, onde $\overline{n}, \overline{c}$ e $\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2.

$$t_*(N,C,V) = 1.048429 - 0.146079\overline{n} - 0.073584\overline{c} + 0.253638\overline{v} + 0.072605\overline{nc} + 0.051285\overline{cv} \tag{8.5}$$

A variação do fator N de 8 para 28 nós, mantendo-se fixos C e V, produz efeito $E_N = e^{2 \times -0.146079} - 1 = -0.2533$, ou seja, um impacto de redução médio de 25% no tempo total, se

mantidos fixos todos os demais fatores significativos. Similarmente, $E_C = e^{2 \times -0.073584} - 1 = -0.1368$ ou -13.68%, enquanto $E_V = e^{2 \times 0.253638} - 1 = +0.6607$, ou aproximadamente 66% de degradação sobre o tempo.

A Figura 20b apresenta o efeito dos fatores principais sobre o tempo de execução, enquanto o gráfico de *Lenth* detalha todos os efeitos que influenciam o tempo para NB no contexto investigado. Memória (M) e disco (D) não exercem influência sobre o tempo. Deve-se observar que, embora o volume exerça influência positiva (aumento) no tempo de execução, seu impacto não se compara às técnicas MLP (Seção 8.5) e RF2 (Seção 8.6).

O modelo reduzido expresso pela Equação 8.5 explica 96.94% da variação sobre a resposta ($R^2 = 0.9694$ e $R^2_{Adj} = 0.9658$) e pode ser utilizado para estimar o tempo de execução de NB para qualquer configuração de *cluster* nas condições investigadas.

8.4.2 Análise de Variabilidade de Resíduos e Modelo de Custo

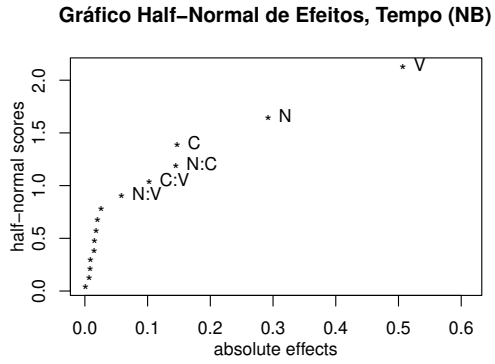
A análise gráfica de resíduos do modelo ajustado sobre os dados originais evidenciou heterogeneidade de variância, principalmente para os fatores N e D. Todavia, não foram percebidos problemas de homogeneidade de variância ou de independência de erros nos demais gráficos. O teste de Shapiro e Wilk (1965) resultou em *p-value* $= 0.047289$. Com valor $\lambda = 0.67 \approx 0.5$ optou-se por transformação de potência de Box e Cox (1964) para escala raiz quadrada: $y^* = \sqrt{y}$. O resultado da transformação sobre os resíduos do modelo pode ser conferido nos gráficos da Figura 21.

O modelo da Tabela 15 explica 99.64% da variação sobre o custo de NB ($R^2 = 0.9964$ e $R^2_{Adj} = 0.9948$), *F-statistic* = 598.4 em 15 e 32 graus de liberdade e *p-value* $< 2.2e^{-16}$. A análise combinada da significância estatística com a proporção de variação explicada (%V.E.) do modelo indica que todos os fatores principais e a interação N:C são relevantes. As demais interações não foram consideradas para o modelo reduzido.

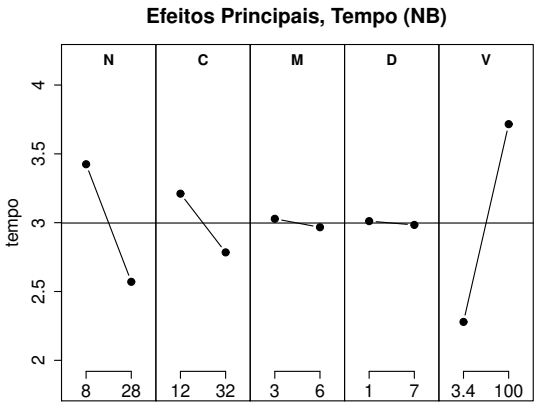
O custo c_* de execução de NB é dado pela Equação 8.6, onde $\overline{n}, \overline{c}$ e $\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2. A transformação inversa da raiz quadrada ($custo = c_*^2$) deve ser aplicada para obtenção do valor monetário na escala original.

$$\begin{aligned} c_*(N, C, M, D, V) = & 2.586954 + 0.485612\overline{n} + 0.479762\overline{c} + 0.088909\overline{m} \\ & + 0.305569\overline{d} + 0.164327\overline{v} + 0.131238\overline{nc} \end{aligned} \tag{8.6}$$

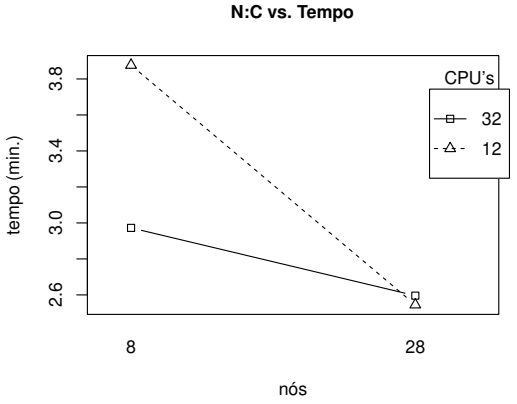
Figura 20 – Análise gráfica de modelo linear para tempo de NB



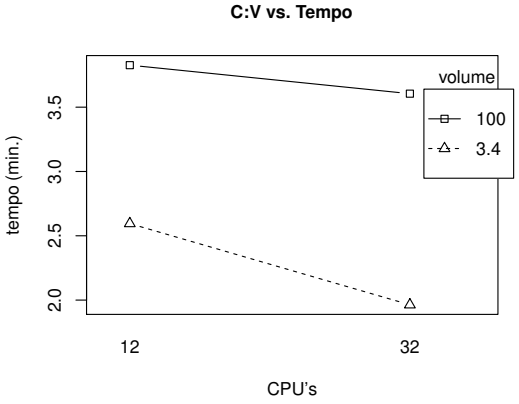
(a) Ranking de efeitos significativos.



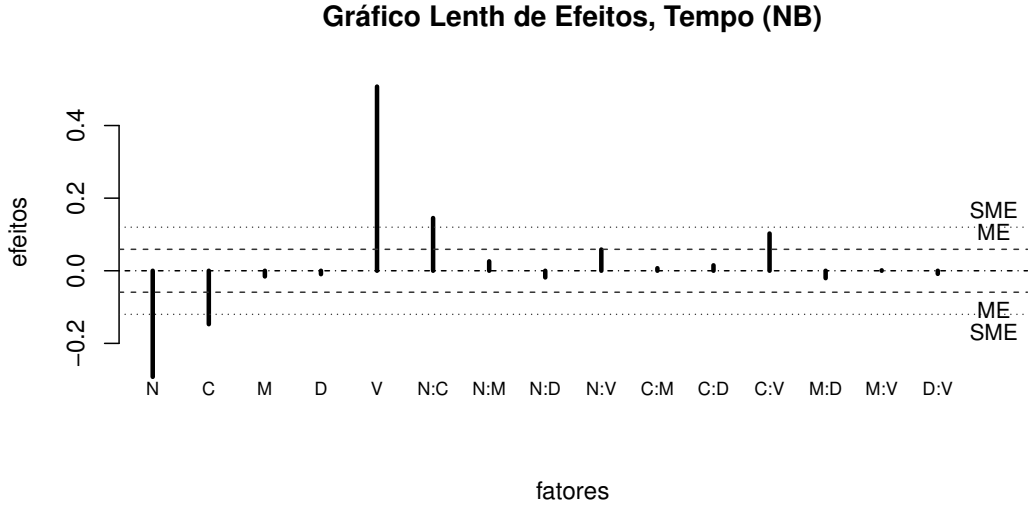
(b) Magnitude dos efeitos principais.



(c) Interação N:V.



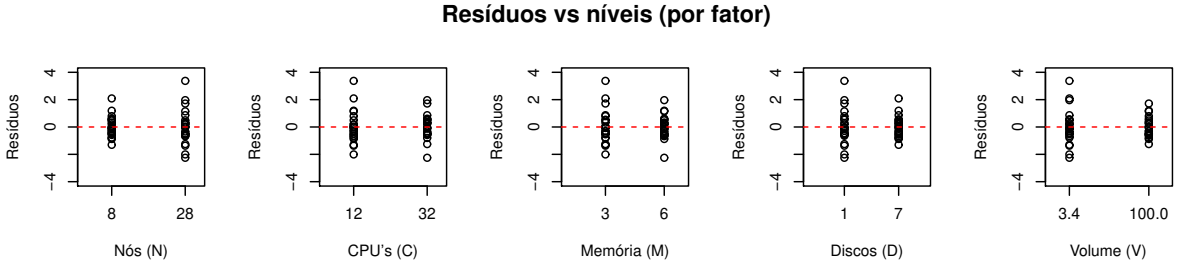
(d) Interação M:D.



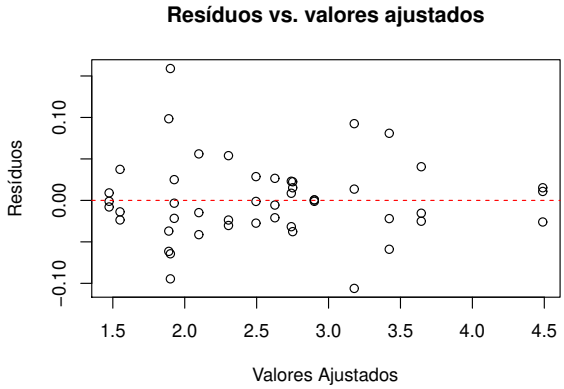
(e) Efeitos do tempo de execução de NB.

Fonte: O autor (2020)

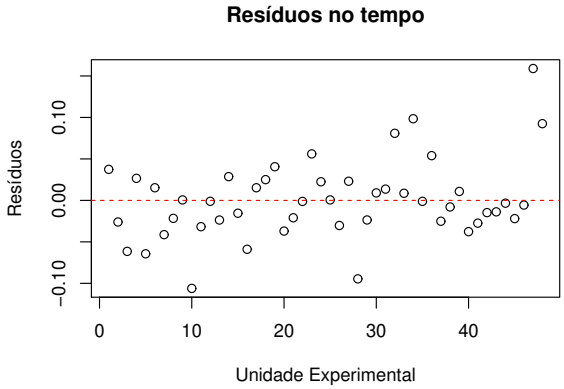
Figura 21 – Resíduos do modelo de custo de NB em escala raiz quadrada



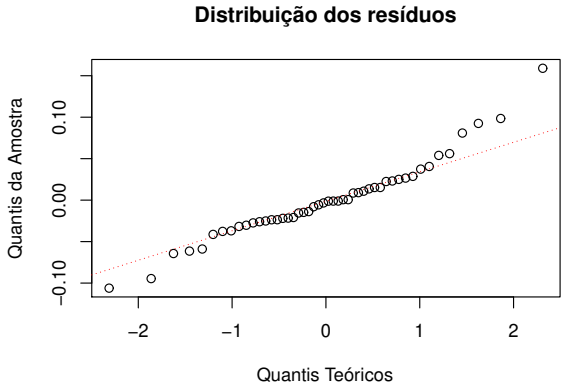
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



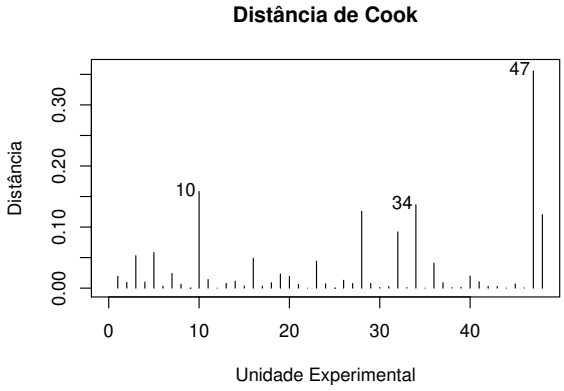
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

Tabela 15 – Modelo linear para custo de execução de NB (escala raiz quadrada)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.586954	310.490	$< 2e^{-16}$ ***	- -
N	**0.485612**	**58.284**	**$< 2e^{-16}$ *** **	**37.71 %**
C	**0.479762**	**57.582**	**$< 2e^{-16}$ *** **	**36.81 %**
M	**0.088909**	**10.671**	**$4.52e^{-12}$ *** **	**1.26 %**
D	**0.305569**	**36.675**	**$< 2e^{-16}$ *** **	**14.93 %**
V	**0.164327**	**19.723**	**$< 2e^{-16}$ *** **	**4.32 %**
N:C	**0.131238**	**15.751**	**$< 2e^{-16}$ *** **	**2.75** %
N:M	0.016714	2.006	0.0534	0.04 %
N:D	0.065820	7.900	$5.15e^{-09}$ ***	0.69 %
N:V	0.042194	5.064	$1.65e^{-05}$ ***	0.28 %
C:M	0.041804	5.017	$1.89e^{-05}$ ***	0.28 %
C:D	0.005480	0.658	0.5154	0.00 %
C:V	0.058213	6.987	$6.45e^{-08}$***	0.54 %
M:D	0.007287	0.875	0.3883	0.01 %
M:V	0.006427	0.771	0.4461	0.01 %
D:V	0.002899	0.348	0.7302	0.00 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

Os efeitos dos coeficientes parciais de regressão não são diretamente interpretáveis devido à dispersão provocada pela transformação raiz quadrada (Seção 5.5.1). O modelo permanece aplicável para predição, mas não é prático para inferência estatística e representação gráfica. Isto posto não se aplicam os gráficos de efeitos principais e interações, gráfico *half-normal* ou mesmo o gráfico de *Lenth*. Muito embora seja difícil estabelecer uma interpretação genérica para o efeito do fator, ainda é possível uma interpretação direta sobre cada instância predita.

Para efeitos de compreensão, tome-se uma variação isolada do fator D e considere-se a notação H_w = [N, C, M, **D**, V] como o arranjo *hardware*/volume de um *cluster* w, sendo H_1 = [8, 12, 3, **1**, 3.4] e H_2 = [8, 12, 3, **7**, 3.4]. Com base no modelo reduzido da Equação 8.6, $c_{*,H_1} = 1.194$, ou R$ 1.43 [4]. O efeito quantitativo sobre um valor predito específico pode ser calculado pela derivada parcial sobre E_D, $2 \times \beta_D \times (c_{*,H_1})^2 = 2 \times 0.305569 \times (1.194)^2 = 0.871$. Uma vez que o plano experimental fatorial está organizado como uma variação de duas unidades do nível –1 para +1, a mudança do arranjo H_1 para H_2 incorreria em um aumento de $2 \times 0.871 = 1.74$. Dessa forma, $c_{*,H_2} = c_{*,H_1} + 1.74 = 1.43 + 1.74 = 3.17$, em média;

[4] O resultado 1.43 deriva da potenciação de dois $(1.194)^2$, transformação inversa da raiz quadrada.

valor que satisfaz o cálculo predito pelo modelo simplificado $c_{*,H_2} = (1.805)^2 =$ \$3.36 com intervalo de confiança de (\$ 2.64; \$ 3.94).

O modelo reduzido expresso pela Equação 8.6 explica 97.78% da variação sobre a resposta ($R^2 = 0.9778$ e $R^2_{Adj} = 0.9746$) e pode ser utilizado para estimar o custo de execução do algoritmo NB para qualquer configuração de *cluster* nas condições investigadas.

8.5 PROJETO EXPERIMENTAL: FLORESTAS ALEATÓRIAS (RF2)

Esta Tese apresenta dois projetos experimentais distintos que empregam Florestas Aleatórias: RF1 sobre dados sintéticos, descritos na Seção 7.1; e RF2 sobre dados reais, descritos nesta seção. O Quadro 13 exibe os hiperparâmetros utilizados no treinamento, enquanto a Tabela 16 detalha o projeto experimental empregado no estudo.

Quadro 13 – Hiperparâmetros para treinamento da técnica RF2

Hiperparâmetro	**Descrição**	**Valor**
Número de árvores	`rf.numTrees`	20
Estratégia de divisão [a]	`rf.featureSubsetStrategy`	`auto`
Medida de impureza	`rf.impurity`	`gini`
Profundidade máxima	`rf.maxDepth`	5
Número máximo de *bins*	`rf.maxBins`	32

[a] – `auto, all, sqrt, log2, onethird`. A opção `auto` delega a escolha ao algoritmo.

Fonte: O autor (2020)

Tabela 16 – Projeto experimental $2_V^{5-1} \times 3$ para tempo e custo de RF2

#	Plano	N	C	M	D	V	t_1, t_2, t_3 (min.)	$\bar{t}$	c_1, c_2, c_3 ($)	$\bar{c}$
1	12-24-35	8 $^-$	12 $^-$	3 $^-$	1 $^-$	100 $^+$	(81.18, 80.68, 71.73)	77.86	(24.95, 24.82, 22.1)	23.96
2	8-28-42	28 $^+$	12 $^-$	3 $^-$	1 $^-$	3.4 $^-$	(2.50, 2.30, 2.50)	2.43	(3.13, 2.96, 3.16)	3.08
3	6-30-46	8 $^-$	32 $^+$	3 $^-$	1 $^-$	3.4 $^-$	(3.37, 3.20, 3.22)	3.26	(3.89, 3.92, 3.78)	3.86
4	9-31-48	28 $^+$	32 $^+$	3 $^-$	1 $^-$	100 $^+$	(20.27, 18.7, 18.65)	19.21	(39.69, 36.86, 36.64)	37.73
5	1-29-40	8 $^-$	12 $^-$	6 $^+$	1 $^-$	3.4 $^-$	(4.77, 4.78, 6.05)	5.20	(2.31, 2.32, 2.76)	2.46
6	13-25-37	28 $^+$	12 $^-$	6 $^+$	1 $^-$	100 $^+$	(27.22, 24.63, 26.75)	26.20	(24.77, 22.4, 24.15)	23.77
7	5-26-45	8 $^-$	32 $^+$	6 $^+$	1 $^-$	100 $^+$	(40.02, 32.08, 36.30)	36.13	(37.49, 30.11, 33.94)	33.85
8	10-18-43	28 $^+$	32 $^+$	6 $^+$	1 $^-$	3.4 $^-$	(1.67, 1.62, 1.67)	1.65	(7.26, 7.33, 7.69)	7.43
9	7-17-39	8 $^-$	12 $^-$	3 $^-$	7 $^+$	3.4 $^-$	(5.18, 5.25, 5.10)	5.18	(3.4, 3.35, 3.38)	3.38
10	3-22-33	28 $^+$	12 $^-$	3 $^-$	7 $^+$	100 $^+$	(28.00, 32.77, 27.50)	29.42	(33.54, 38.13, 32.82)	34.83
11	14-19-41	8 $^-$	32 $^+$	3 $^-$	7 $^+$	100 $^+$	(37.37, 38.07, 32.57)	36.00	(34.41, 35.14, 30.47)	33.34
12	2-32-36	28 $^+$	32 $^+$	3 $^-$	7 $^+$	3.4 $^-$	(1.87, 1.68, 1.80)	1.78	(11.06, 10.45, 11.17)	10.89
13	4-27-44	8 $^-$	12 $^-$	6 $^+$	7 $^+$	100 $^+$	(71.7, 75.27, 75.92)	74.3	(34.27, 35.93, 36.24)	35.48
14	11-20-38	28 $^+$	12 $^-$	6 $^+$	7 $^+$	3.4 $^-$	(2.00, 1.95, 1.95)	1.97	(6.76, 5.96, 5.94)	6.22
15	16-23-47	8 $^-$	32 $^+$	6 $^+$	7 $^+$	3.4 $^-$	(3.00, 2.98, 3.02)	3.00	(6.43, 6.16, 6.26)	6.28
16	15-21-34	28 $^+$	32 $^+$	6 $^+$	7 $^+$	100 $^+$	(19.78, 19.35, 22.07)	20.4	(55.62, 54.77, 61.17)	57.19

Fonte: O autor (2020), com resultados de 48 experimentos no ambiente *Spark — Google Data Proc*

8.5.1 Análise de Variabilidade de Resíduos e Modelo de Tempo

Os resíduos para todos os níveis de fatores apresentaram heterogeneidade de variância, em especial N e V. De maneira análoga, os resíduos *versus* valores ajustados do modelo evidenciaram violações à homogeneidade. Os quantis dos resíduos se distanciaram da distribuição Normal, com teste de Shapiro e Wilk (1965) resultando em *p-value* $= 0.000181$. Ainda, houve pontos discrepantes com desvios superiores a 4σ no gráfico de distância de *Cook*. O valor $\lambda = -0.18$ sugeriu transformação de potência de Box e Cox (1964) para escala logarítmica: $y^* = \log y$. A Figura 22 exibe os gráficos de resíduos para o modelo ajustado sobre os dados transformados. Observou-se que a transformação diminuiu as distorções de heterogeneidade de variância por nível de cada fator (Figura 22a) e, em especial, melhorou significativamente a distribuição dos quantis dos resíduos, conforme Figura 22d e teste de Shapiro e Wilk (1965) com *p-value* $= 0.616051$.

A Tabela 17 detalha o modelo linear para o tempo que explica 99.84% da variação sobre a resposta ($R^2 = 0.9984$ e $R^2_{Adj} = 0.9976$), *F-statistic* = 1319 em 15 e 32 graus de liberdade e *p-value* $<$ *2.2e-16*. A ordem relativa de significância estatística entre os fatores pode ser conferida no gráfico *half-normal* de efeitos da Figura 23a. Os fatores significativos foram N, C e V. Mesmo apresentando significância estatística, o fator M e as interações N:C e C:V não devem ser consideradas para a construção do modelo reduzido, devido à contribuição de apenas 0.05%, 0.39% e 0.1%, respectivamente.

O modelo reduzido para o tempo t_* pode ser expresso pela Equação 8.7, onde $\overline{n}, \overline{c}, \overline{m}, \overline{d}$ e $\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2.

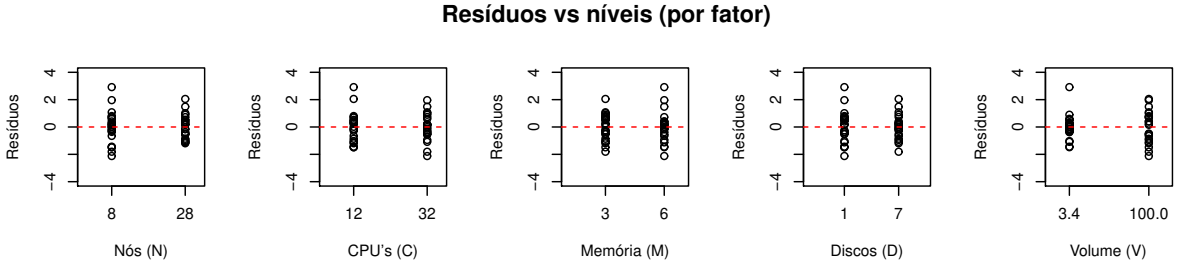
$$t_*(N, C, V) = 2.290607 - 0.383509\overline{n} - 0.228855\overline{c} + 1.264055\overline{v} \tag{8.7}$$

A variação isolada do fator N de 8 para 28 nós, mantendo-se fixos C e V, produz efeito $E_N = e^{2 \times -0.383509} - 1 \approx -0.5356$, ou seja, execução 53% mais rápido. De forma similar, $E_C = e^{2 \times -0.228855} - 1 \approx -0.3672 = -36.72\%$, enquanto a variação de volume produz o expressivo efeito $E_V = e^{2 \times 1.264055} - 1 \approx 1.1153 = +1153\%$ sobre o tempo.

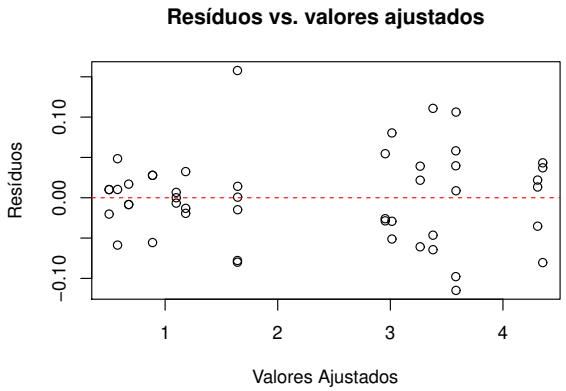
A Figura 23b apresenta o efeito dos fatores principais, enquanto o gráfico de *Lenth* detalha todos os efeitos que influenciam o tempo para RF2 no contexto investigado.

A Equação 8.7 explica 99.18% da variação sobre o tempo ($R^2 = 0.9918$ e $R^2_{Adj} = 0.9912$) e pode ser utilizada para estimar o tempo de execução do algoritmo RF2 dentro dos limites

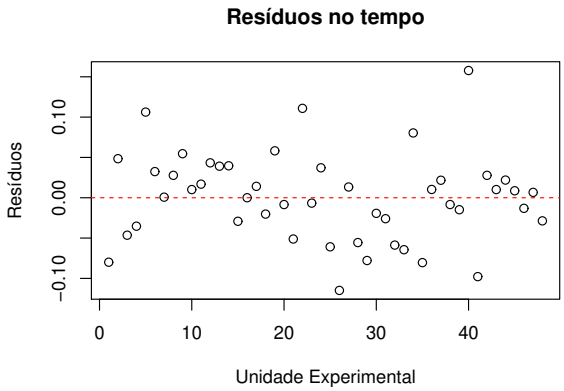
Figura 22 – Resíduos do modelo de tempo de RF2 em escala logarítmica



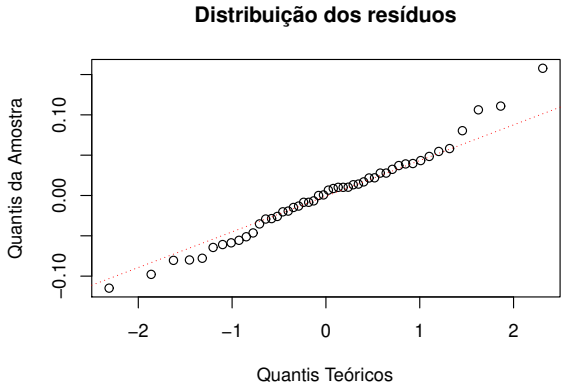
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



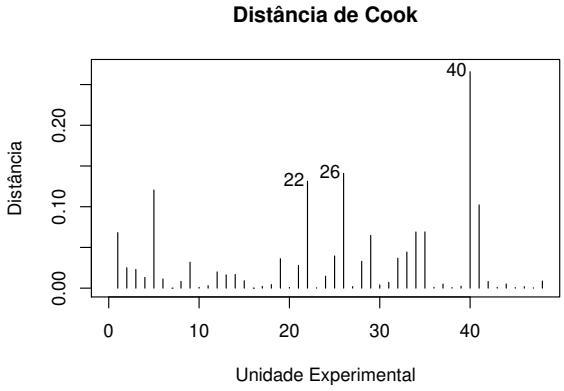
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

Tabela 17 – Modelo linear para tempo de execução de RF2 (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.290607	239.527	$< 2e^{-16}$	- -
N	**-0.383509**	**-40.103**	**$< 2e^{-16}$ *****	**8.12 %**
C	**-0.228855**	**-23.931**	**$< 2e^{-16}$ *****	**2.89 %**
M	-0.029431	-3.078	0.00426 **	0.05 %
D	-0.005865	-0.613	0.54404	0.00 %
V	**1.264055**	**132.181**	**$< 2e^{-16}$ *****	**88.17 %**
N:C	0.083898	8.773	$5.03e^{-10}$ ***	0.39 %
N:M	-0.013257	-1.386	0.17523	0.01 %
N:D	0.010335	1.081	0.28791	0.01 %
N:V	-0.018198	-1.903	0.06607	0.02 %
C:M	0.017252	1.804	0.08065	0.02 %
C:D	0.011911	1.246	0.22197	0.01 %
C:V	-0.042623	-4.457	$9.55e^{-05}$ ***	0.10 %
M:D	0.018808	1.967	0.05794	0.02 %
M:V	0.017162	1.795	0.08217	0.02 %
D:V	0.021571	2.256	0.03106	0.03 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

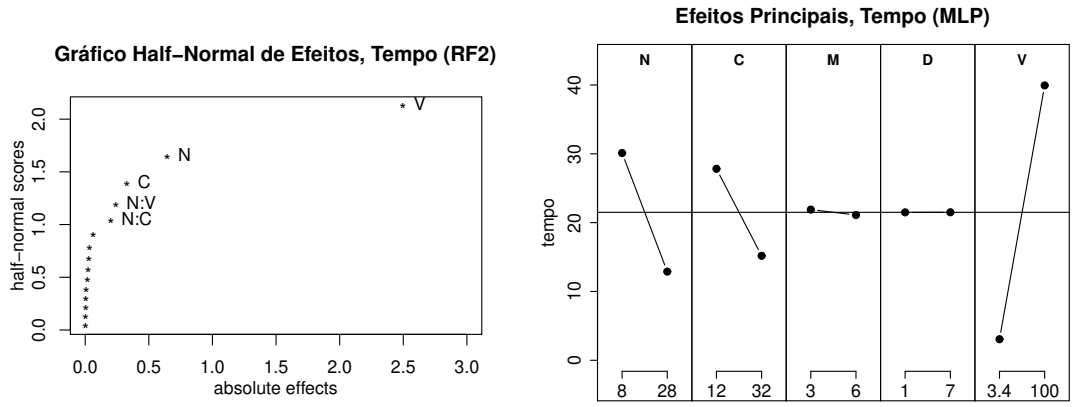
de variação de *hardware* e volume de entrada aqui apresentados.

8.5.2 Análise de Variabilidade de Resíduos e Modelo de Custo

Os resíduos apontaram problemas de homogeneidade de variância em face dos níveis de fatores e dos valores ajustados. Os quantis dos resíduos desviaram da distribuição Normal com *p-value* $= 0.002774$ dado pelo teste de Shapiro e Wilk (1965). A unidade experimental n° 34 evidenciou distância de *Cook* com desvio acima de 2σ. O valor $\lambda = -0.10$ sugeriu transformação de potência de Box e Cox (1964) para escala logarítmica: $y^* = \log y$. Após transformação, os resíduos evidenciaram aderência às premissas de homogeneidade (Figuras 24a e 24b), independência de erros (Figura 24c) e normalidade de resíduos (Figura 24d) com *p-value* $= 0.890561$ dado pelo teste de Shapiro e Wilk (1965). A Figura 24e sugere também a redução da distorção dos pontos discrepantes sobre o modelo.

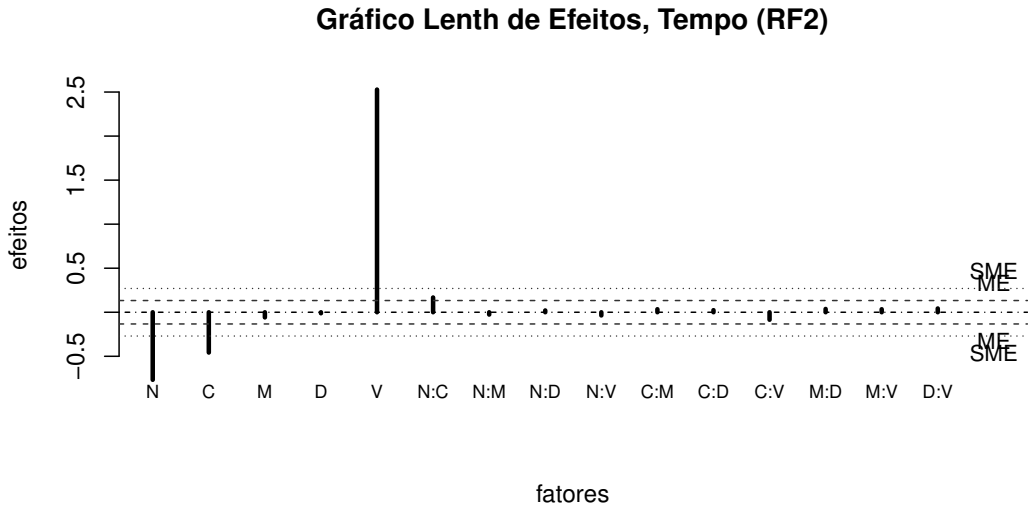
A Tabela 18 detalha o modelo linear para o custo que explica 99.78% da variação na resposta ($R^2 = 0.9978$ e $R^2_{Adj} = 0.9967$), *F-statistic* $=$ 949.6 em 15 e 32 graus de liberdade

Figura 23 – Análise gráfica de modelo linear para tempo de RF2



(a) Ranking de efeitos significativos.

(b) Magnitude dos efeitos principais.



(c) Efeitos do tempo de execução de RF2.

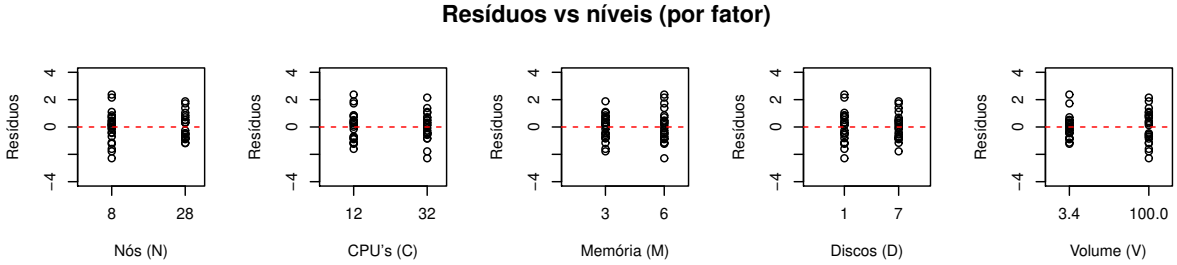
Fonte: O autor (2020)

e *p-value* $<$ *2.2e*$^{-16}$. Todos os fatores e interações, exceto N:M, C:M e M:V apresentaram significância estatística. Todavia, apenas N, C, D e V devem ser considerados para o modelo reduzido devido à proporção de variação explicada e consequente relevância prática. O *ranking* dos efeitos pode ser conferido no gráfico *half-normal* da Figura 25a.

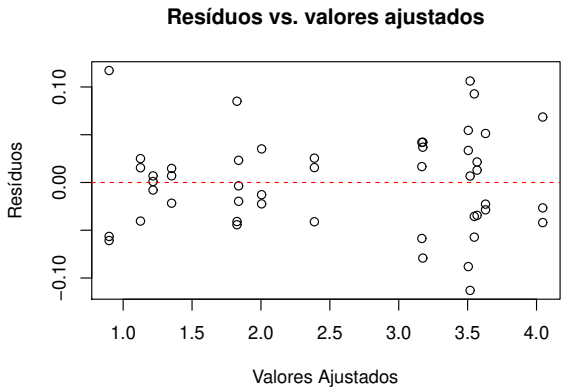
$$c_*(N,C,D,V) = 2.550306 + 0.166549\overline{n} + 0.234593\overline{c} + 0.191585\overline{d} + 0.969299\overline{v} \tag{8.8}$$

O efeito do fator N sobre o custo é $E_N = e^{2 \times 0.166549} - 1 = +0.3953$, o que significa que uma variação entre os número mínimo e máximo de nós do *cluster*, mantendo fixos todos

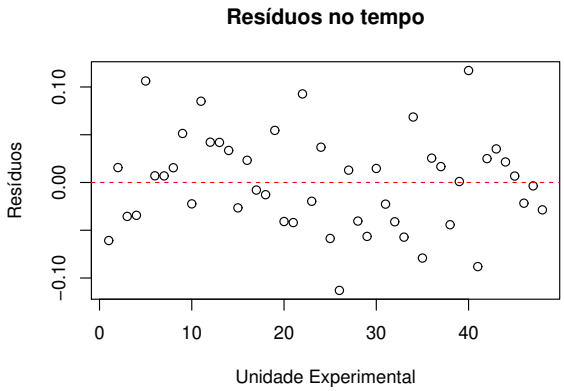
Figura 24 – Resíduos do modelo de custo de RF2 em escala logarítmica



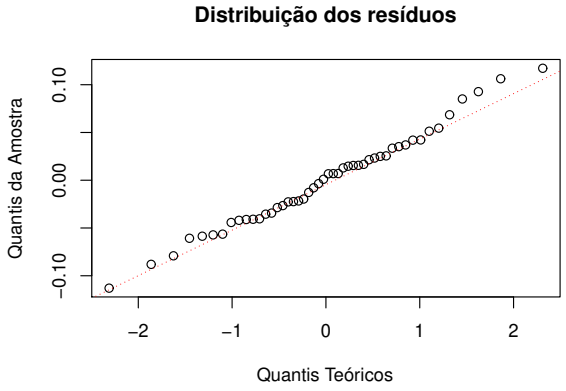
(a) Resíduos *vs.* níveis dos fatores N, C, M, D e V.



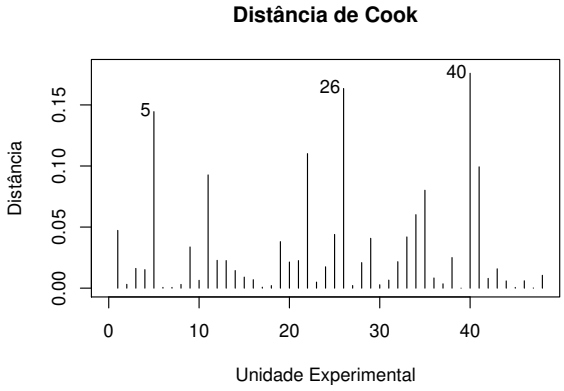
(b) Variância dos resíduos *vs.* valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

Tabela 18 – Modelo linear para custo de execução de RF2 (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.550306	292.012	$< 2e^{-16}$ ***	- -
N	**0.166549**	**19.070**	**$< 2e^{-16}$ *** **	**2.55 %**
C	**0.234593**	**26.861**	**$< 2e^{-16}$ *** **	**5.05 %**
M	0.057911	6.631	$1.77e^{-07}$ ***	0.31 %
D	**0.191585**	**21.937**	**$< 2e^{-16}$ *** **	**3.37 %**
V	**0.969299**	**110.985**	**$< 2e^{-16}$ *** **	**86.29 %**
N:C	0.065399	7.488	$1.59e^{-08}$ ***	0.39 %
N:M	-0.013899	-1.591	0.121342	0.02 %
N:D	0.043319	4.960	$2.23e^{-05}$ ***	0.17 %
N:V	-0.088448	-10.127	$1.66e^{-11}$ ***	0.72 %
C:M	0.008555	0.980	0.334657	0.01 %
C:D	-0.032635	-3.737	0.000729 ***	0.10 %
C:V	-0.079809	-9.138	$1.96e^{-10}$ ***	0.58 %
M:D	0.019564	2.240	0.032161 *	0.04 %
M:V	-0.002695	-0.309	0.759607	0.00 %
D:V	-0.044477	-5.093	$1.52e^{-05}$ ***	0.18 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

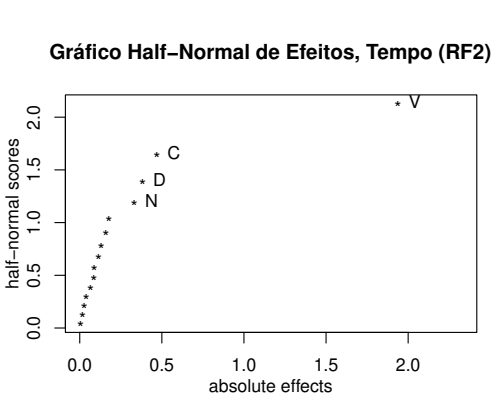
b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

os demais fatores, resultaria em um incremento de custo de 39.53%, em média. Igualmente, $E_C = e^{2 \times 0.234593} - 1 = +0.5987$ ou $\approx 60\%$; $E_D = e^{2 \times 0.191585} - 1 = +0.4669$ ou $\approx 47\%$; enquanto a variação de volume produz o relevante efeito $E_V = e^{2 \times 0.969299} - 1 = +5.949\%$, ou $\approx 595\%$ de aumento no custo de execução.

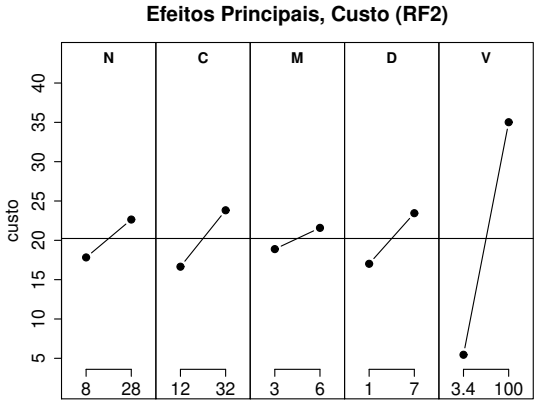
A magnitude dos efeitos dos fatores principais sobre o custo estão expressos na Figura 25b, onde se percebe a dominância de influência do fator V. Já o gráfico de *Lenth* detalha todos os efeitos que influenciam o custo para RF2 no contexto investigado.

A Equação 8.8 explica 97.26% da variação sobre o custo ($R^2 = 0.9726$ e $R^2_{Adj} = 0.9700$) e pode ser utilizada para estimar o tempo de execução do algoritmo RF2 dentro dos limites de variação de *hardware* e volume de entrada aqui apresentados.

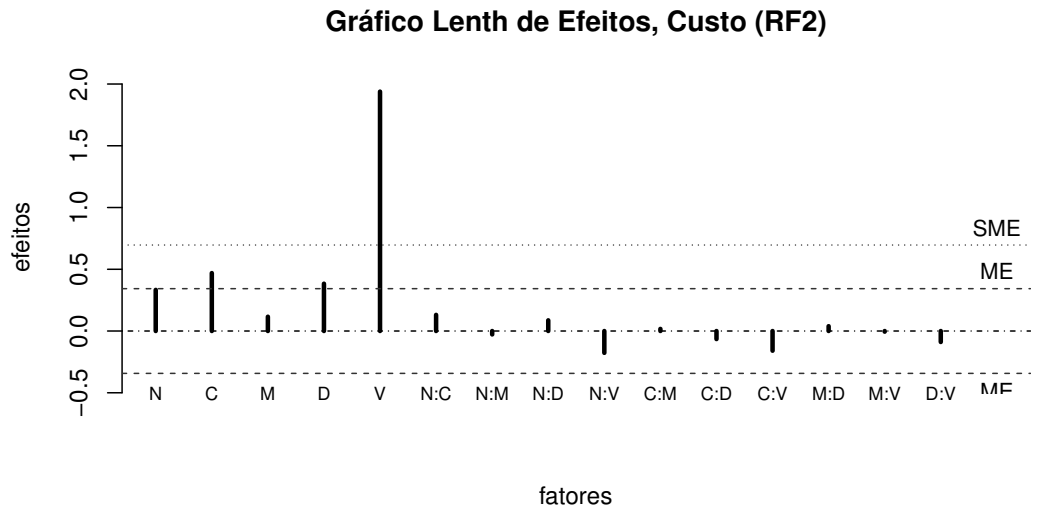
Figura 25 – Análise gráfica de modelo linear para custo de RF2



(a) Ranking de efeitos significativos.



(b) Magnitude dos efeitos principais.



(c) Efeitos do custo de execução de RF2.

Fonte: O autor (2020)

8.6 PROJETO EXPERIMENTAL: *PERCEPTRON* DE MÚLTIPLAS CAMADAS (MLP)

O Quadro 14 exibe os hiperparâmetros utilizados no treinamento, enquanto a Tabela 19 detalha o projeto experimental empregado no estudo.

Quadro 14 – Hiperparâmetros para treinamento da técnica MLP

Hiperparâmetro	**Descrição**	**Valor**
Número de camadas	- -	4
Número de neurônios	Camada de entrada	16384
Número de neurônios	Camada intermediária 1	30
Número de neurônios	Camada intermediária 2	5
Número de neurônios	Camada de saída	2
Algoritmo de otimização	`mlp.setSolver`	`l-bfgs` [a]
Máximo de iterações	`mlp.setMaxIter`	10
Taxa de aprendizagem	`mlp.stepSize`	0.03
Tolerância de convergência	`mlp.setTol`	10^{-6}

[a] – *Limited-memory BFGS* (LIU; NOCEDAL, 1989).

Fonte: O autor (2020)

Tabela 19 – Projeto experimental $2_V^{5-1} \times 3$ para tempo e custo de MLP

#	Plano	N	C	M	D	V	t_1, t_2, t_3 (min.)	$\bar{t}$	c_1, c_2, c_3 ($)	$\bar{c}$
1	8-25-35	8 $^-$	12 $^-$	3 $^-$	1 $^-$	100 $^+$	(64, 47.05, 53.27)	54.77	(19.26, 14.67, 16.58)	16.84
2	7-17-43	28 $^+$	12 $^-$	3 $^-$	1 $^-$	3.4 $^-$	(1.9, 1.65, 1.53)	1.69	(2.73, 2.65, 2.38)	2.59
3	5-18-45	8 $^-$	32 $^+$	3 $^-$	1 $^-$	3.4 $^-$	(1.78, 2.57, 1.72)	2.02	(2.76, 3.19, 2.57)	2.84
4	14-20-40	28 $^+$	32 $^+$	3 $^-$	1 $^-$	100 $^+$	(14.4, 14, 14.87)	14.42	(29.42, 28.59, 29.89)	29.3
5	1-21-36	8 $^-$	12 $^-$	6 $^+$	1 $^-$	3.4 $^-$	(3.68, 2.87, 2.8)	3.12	(1.87, 1.7, 1.66)	1.74
6	12-26-39	28 $^+$	12 $^-$	6 $^+$	1 $^-$	100 $^+$	(15.45, 20.25, 16.07)	17.26	(14.65, 18.61, 15.11)	16.12
7	13-24-34	8 $^-$	32 $^+$	6 $^+$	1 $^-$	100 $^+$	(29.77, 31.08, 25.35)	28.73	(27.53, 28.54, 24.11)	26.73
8	4-22-42	28 $^+$	32 $^+$	6 $^+$	1 $^-$	3.4 $^-$	(1.48, 1.6, 1.55)	1.54	(6.82, 8.24, 8.18)	7.75
9	3-23-46	8 $^-$	12 $^-$	3 $^-$	7 $^+$	3.4 $^-$	(3.15, 3.37, 2.88)	3.13	(2.49, 2.53, 2.47)	2.5
10	6-30-41	28 $^+$	12 $^-$	3 $^-$	7 $^+$	100 $^+$	(18.28, 17.88, 15.98)	17.38	(23.07, 22.73, 21.31)	22.37
11	15-29-48	8 $^-$	32 $^+$	3 $^-$	7 $^+$	100 $^+$	(26.12, 30.57, 27.55)	28.08	(25.01, 28.65, 27.62)	27.09
12	16-28-37	28 $^+$	32 $^+$	3 $^-$	7 $^+$	3.4 $^-$	(2.05, 1.5, 1.43)	1.66	(10.67, 11.31, 11.2)	11.06
13	11-19-47	8 $^-$	12 $^-$	6 $^+$	7 $^+$	100 $^+$	(47.52, 46.48, 49.27)	47.76	(23.39, 22.97, 24.25)	23.54
14	9-31-33	28 $^+$	12 $^-$	6 $^+$	7 $^+$	3.4 $^-$	(1.7, 1.75, 1.6)	1.68	(5.52, 5.67, 7.05)	6.08
15	2-32-44	8 $^-$	32 $^+$	6 $^+$	7 $^+$	3.4 $^-$	(2.07, 1.82, 1.77)	1.89	(5.07, 5.51, 7.37)	5.98
16	10-27-38	28 $^+$	32 $^+$	6 $^+$	7 $^+$	100 $^+$	(15.98, 13.15, 12.85)	13.99	(45.96, 39.64, 41.94)	42.51

Fonte: O autor (2020), com resultados de 48 experimentos no ambiente *Spark — Google Data Proc*

8.6.1 Análise de Variabilidade de Resíduos e Modelo de Tempo

Os resíduos evidenciaram heterogeneidade de variância por níveis de cada fator e distorções no gráfico de resíduos *versus* valores ajustados. O teste de Shapiro e Wilk (1965) acusou ausência de normalidade com *p-value* $= 8.6682 \times 10^{-8}$ e influência de pontos discrepantes com distâncias de *Cook* superiores a 4σ e 6σ, para as unidades experimentais nº 8 e 25, respectivamente. O valor de $\lambda = 0.061$ sugeriu transformação de potência de Box e Cox (1964) para escala logarítmica: $y^* = \log y$. A Figura 26 exibe os gráficos de resíduos para o modelo ajustado sobre os dados transformados. Observa-se homogeneidade de variância, diminuição do desvio de pontos discrepantes e significativa melhora na distribuição dos resíduos, cuja normalidade é sugerida pela Figura 26d e pelo teste de Shapiro e Wilk (1965) com *p-value* $= 0.044361 \approx 0.05$.

O modelo linear da Tabela 20 explica 99.46% da variação na resposta ($R^2 = 0.9946$ e $R^2_{Adj} = 0.9921$), *F-statistic* = 392.5 em 15 e 32 graus de liberdade e *p-value* $< 2.2e^{-16}$. Tanto o nível de significância estatística do modelo quanto o gráfico *half-normal* da Figura 27a evidenciam N, C, V, e as interações N:C e N:V como significativos. Todavia, as interações de segunda ordem não possuem importância prática (%V.E. $< 1\%$) e não foram consideradas na construção do modelo simplificado.

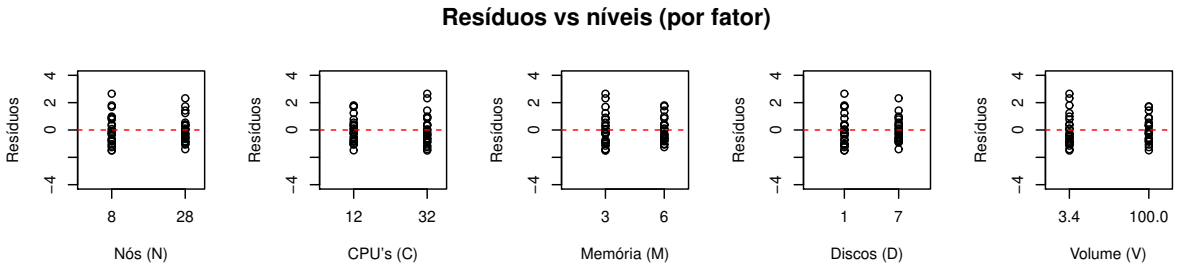
O tempo t_* pode ser estimado pela Equação 8.10, onde $\overline{n}, \overline{c}, \overline{m}, \overline{d}$ e $\overline{v}$ estão em conformidade com o Quadro 9, Seção 8.1.2.

$$t_*(N, C, V) = 1.943987 - 0.323166\,\overline{n} - 0.164413\,\overline{c} + 1.249130\overline{v} \tag{8.9}$$

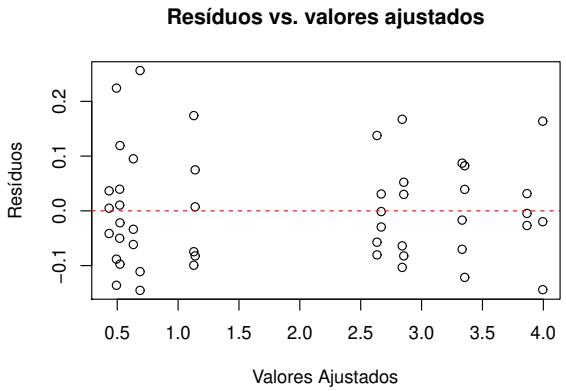
O efeito do fator N sobre o tempo é $E_N = e^{2 \times -0.323166} - 1 \approx -0.4760$, indicando que o aumento do tamanho *cluster* de 8 para 28 nós produz, em média, 48% de variação no tempo. Com similar raciocínio, $E_C = e^{2 \times -0.164413} - 1 \approx 0.2802 = -28.02\%$, enquanto a variação de volume produz o relevante efeito $E_V = e^{2 \times 1.249130} - 1 \approx 11.1613 = +1116.13\%$.

A Figura 27b apresenta o efeito dos fatores principais sobre o tempo de execução, enquanto o gráfico de *Lenth* detalha todos os efeitos que influenciam o tempo para MLP no contexto investigado. Memória (M) e disco (D) não exercem influência sobre o tempo. A performance é seriamente degradada em função do volume de entrada, conforme modelo linear, à taxa de 1116.13%, seguido por um efeito inverso (diminuição de tempo) em relação à variação de nós (N) e núcleos (C).

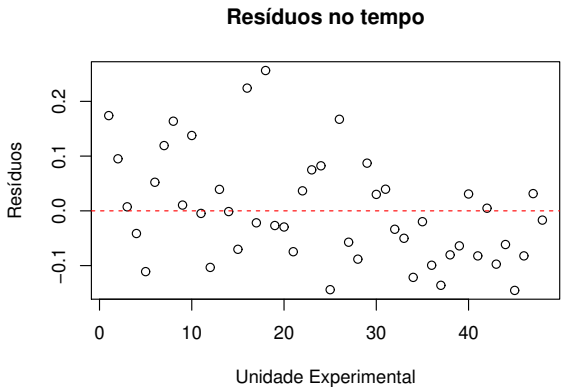
Figura 26 – Resíduos do modelo de tempo de MLP em escala logarítmica



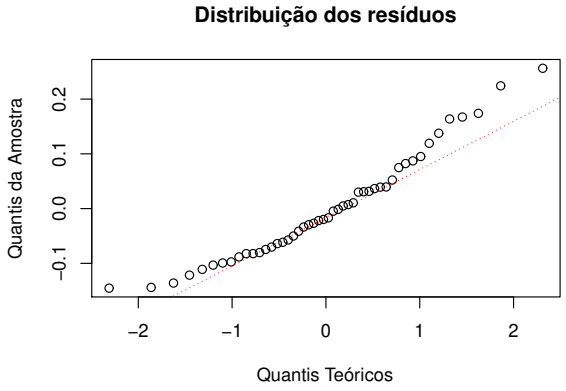
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



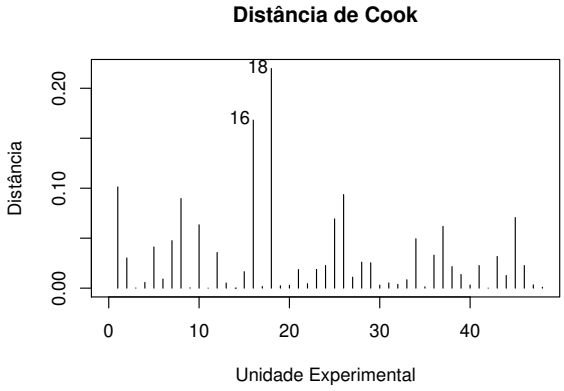
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

Tabela 20 – Modelo linear para tempo de execução de MLP (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	1.943987	113.790	$< 2e^{-16}$ ***	- -
N	**-0.323166**	**-18.916**	**$< 2e^{-16}$ *** **	**6.04 %**
C	**-0.164413**	**-9.624**	**$5.75e^{-11}$ *** **	**1.56 %**
M	-0.017783	-1.041	0.3057	0.02 %
D	-0.009952	-0.583	0.5643	0.01 %
V	**1.249130**	**73.117**	**$< 2e^{-16}$ *** **	**90.31 %**
N:C	0.100926	5.908	$1.42e^{-06}$ ***	0.59 %
N:M	0.003992	0.234	0.8167	0.00 %
N:D	0.014394	0.843	0.4057	0.01 %
N:V	-0.120761	-7.069	$5.13e^{-08}$ ***	0.84 %
C:M	0.001676	0.098	0.9225	0.00 %
C:D	0.003558	0.208	0.8363	0.00 %
C:V	-0.031346	-1.835	0.0758	0.06 %
M:D	-0.003235	-0.189	0.8510	0.00 %
M:V	-0.001655	-0.097	0.9234	0.00 %
D:V	-0.011653	-0.682	0.5001	0.01 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

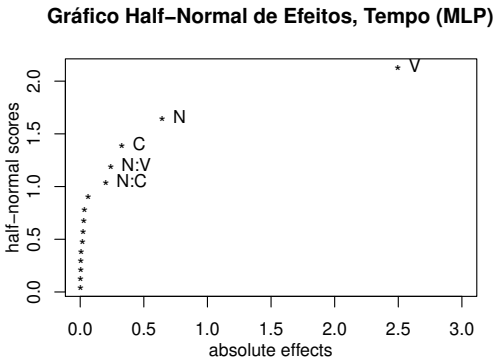
A Equação 8.9 explica 97.91% da variação na resposta ($R^2 = 0.9792$, $R^2_{Adj} = 0.9778$) e pode ser utilizada para estimar o tempo de execução de MLP para configurações de *cluster* nas condições investigadas.

8.6.2 Análise de Variabilidade de Resíduos e Modelo de Custo

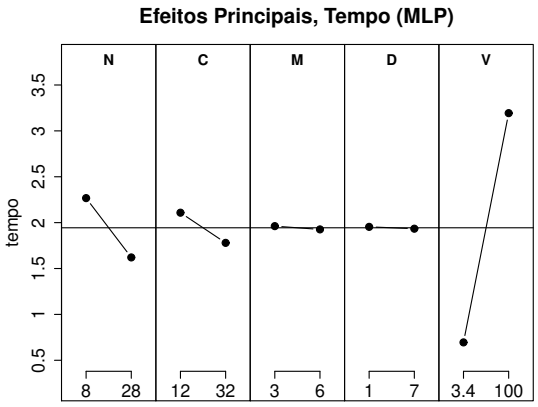
A análise gráfica evidenciou heterogeneidade de variância dos resíduos em face dos níveis de cada fator, especialmente para o fator V. O cálculo da distância de *Cook* detectou as unidades experimentais nº 10, 27 e 34 como pontos discrepantes. Todavia, o teste de Shapiro e Wilk (1965) apontou normalidade dos resíduos com *p-value* $= 0.067057$. O cálculo $\lambda = 0.10$ sugeriu transformação de potência de Box e Cox (1964) para a escala logarítmica.

Os resíduos do modelo ajustado sobre os dados transformados passaram a apresentar homogeneidade de variância (Figuras 28a e 28b) e independência de erros (Figura 28c). A Figura 28d sugere manutenção da normalidade dos resíduos, confirmada pelo teste de Shapiro e Wilk (1965) com *p-value* $= 0.495216$. Houve ainda redução do número de pontos discrepantes,

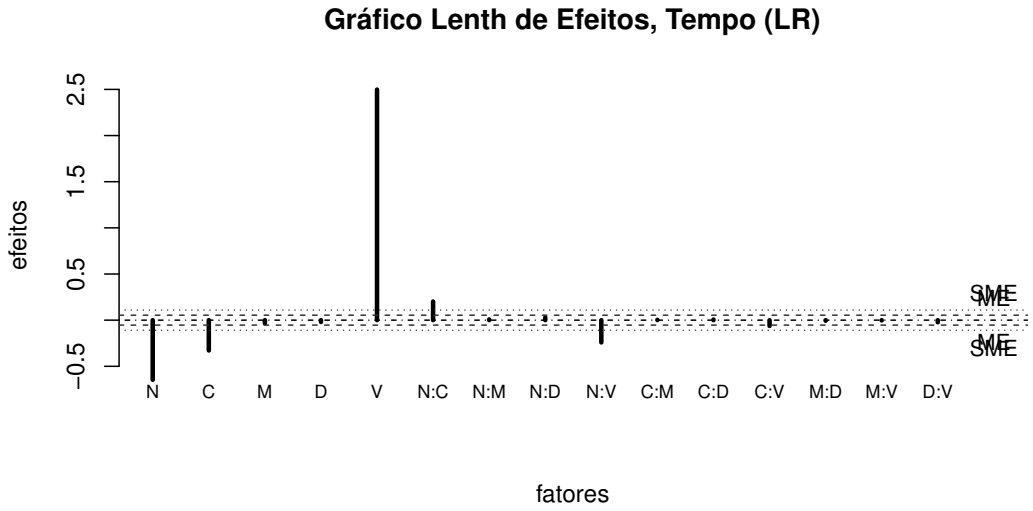
Figura 27 – Análise gráfica de modelo linear para tempo de MLP



(a) Ranking de efeitos significativos.



(b) Magnitude dos efeitos principais.



(c) Efeitos do tempo de execução de MLP.

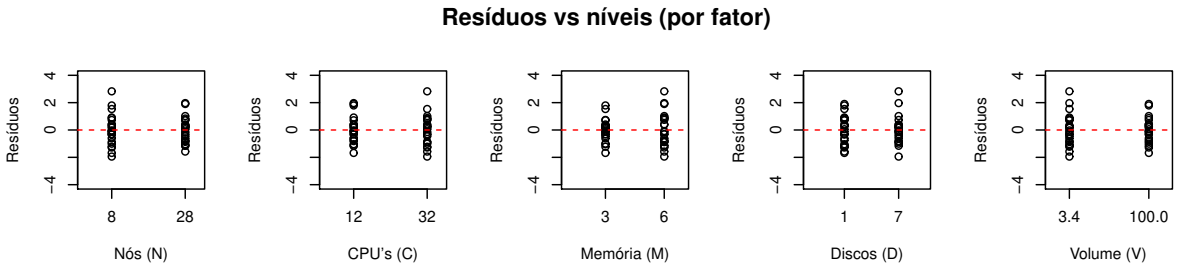
Fonte: O autor (2020)

restando apenas a unidade experimental n° 44 com desvio superior a 2σ em relação à distância de *Cook* (Figura 28e).

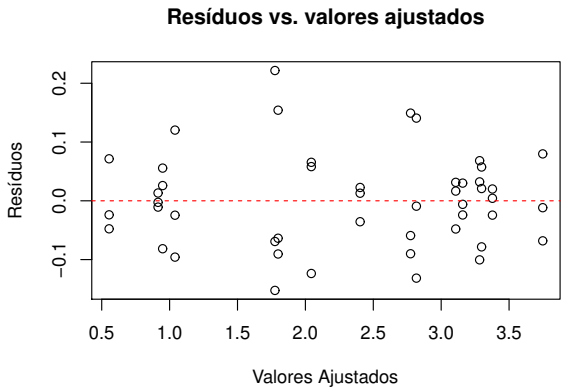
O modelo linear da Tabela 21 explica 99.39% da variação sobre o custo com coeficientes de determinação $R^2 = 0.9939$ e $R^2_{Adj} = 0.9911$, *F-statistic* = 348.9 em 15 e 32 graus de liberdade e *p-value* $< 2.2e^{-16}$. Os fatores significativos foram N, C, D, V, e a interação N:V. Embora apresentem significância estatística, o fatores M e as interações N:C, N:D, C:V, M:V e D:V não devem ser considerados devido ao valor de %V.E. < 1, explicando em conjunto apenas 2.3% do resultado.

O modelo reduzido para o custo c_* pode ser expresso pela Equação 8.10, onde $\overline{n}, \overline{c}, \overline{d}$ e $\overline{v}$

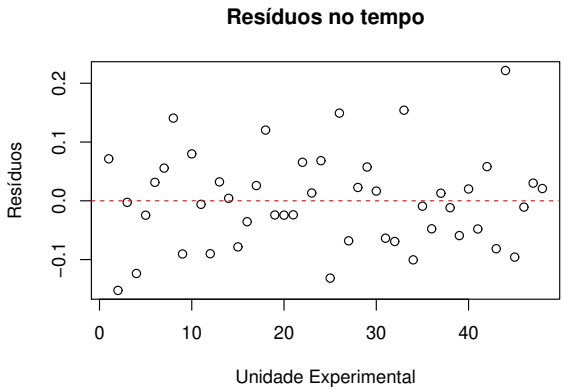
Figura 28 – Resíduos do modelo de custo de MLP em escala logarítmica



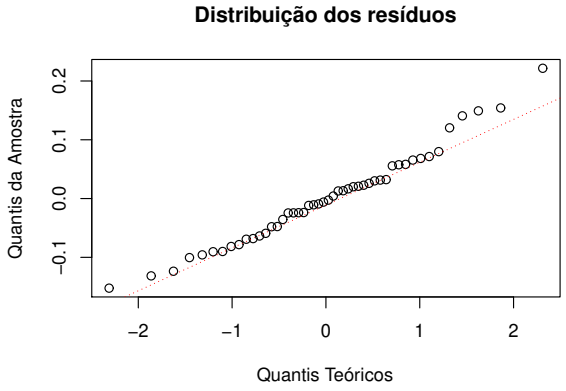
(a) Resíduos vs. níveis dos fatores N, C, M, D e V.



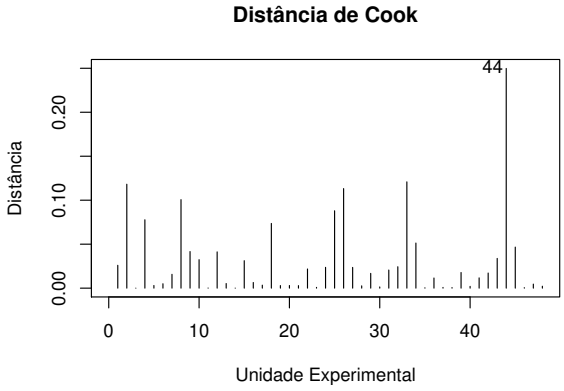
(b) Variância dos resíduos vs. valores ajustados.



(c) Resíduos em sequência temporal.



(d) Normalidade dos resíduos.



(e) Verificação de pontos discrepantes.

Fonte: O autor (2020)

Tabela 21 – Modelo linear para custo de execução de MLP (escala logarítmica)

	Coef.'s β	**t-value**	**p-value**[a]	**% V.E.** [b]
Intercepto	2.31514	167.070	$< 2e^{-16}$ ***	
N	**0.20999**	**15.154**	**$3.73e^{-16}$ *** **	**4.36 %**
C	**0.30587**	**22.073**	**$< 2e^{-16}$ *** **	**9.25 %**
M	0.07691	5.550	$4.01e^{-06}$ ***	0.59 %
D	**0.21031**	**15.177**	**$3.58e^{-16}$ *** **	**4.37 %**
V	**0.88028**	**63.525**	**$< 2e^{-16}$ *** **	**76.64 %**
N:C	0.06197	4.472	$9.15e^{-05}$ ***	0.38 %
N:M	-0.01084	-0.782	0.4400	0.01 %
N:D	0.02880	2.078	0.0458 *	0.08 %
N:V	**-0.15366**	**-11.089**	**$1.71e^{-12}$ *** **	**2.34 %**
C:M	0.01465	1.057	0.2983	0.02 %
C:D	-0.02522	-1.820	0.0781	0.06 %
C:V	-0.07477	-5.396	$6.28e^{-06}$ ***	0.55 %
M:D	0.01786	1.289	0.2067	0.03 %
M:V	-0.03138	-2.264	0.0305 *	0.10 %
D:V	-0.07797	-5.627	$3.21e^{-06}$ ***	0.60 %

a – $^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

b – % V.E. - proporção de variação explicada; não deve ser confundida com o efeito do fator, interpretado como percentual devido à transformação *log*.

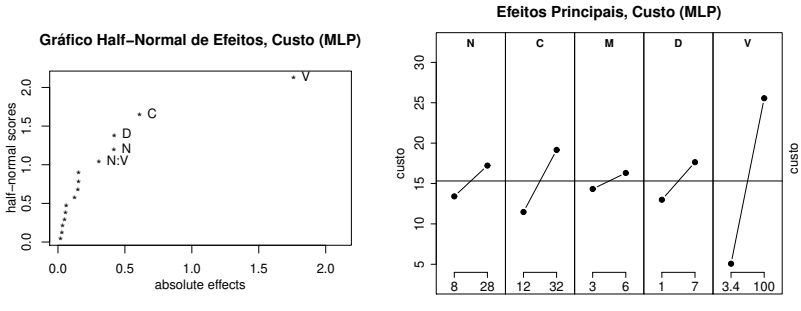
estão em conformidade com o Quadro 9, Seção 8.1.2.

$$c_*(N,C,D,V) = 2.315143 + 0.209988\,\overline{n} + 0.305872\,\overline{c} + 0.210307d + 0.880279\overline{v} - 0.153660\,\overline{nv} \tag{8.10}$$

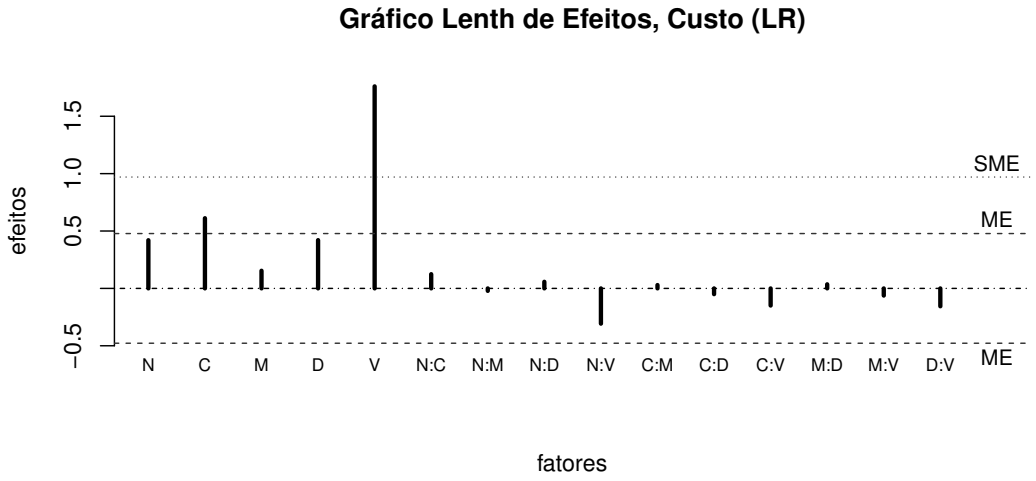
O efeito $E_N = e^{2\times 0.209988} - 1 = +0.5219$ indica um impacto médio de 52.19% no custo de execução do algoritmo MLP quando ocorre variação isolada de 8 a 28 nós no *cluster*. Analogamente, $E_C = e^{2\times 0.305872} - 1 = +0.8436$, $E_D = e^{2\times 0.210307} - 1 = +0.5229$ e $E_V = e^{2\times 0.880279} - 1 = +4.8157$.

A Figura 29b exibe a magnitude dos efeitos dos fatores principais, enquanto o gráfico de *Lenth* detalha a forte influência de V sobre o custo, reforçando a quantificação de E_V à taxa de 481.57%. A interação N:V deve ser analisada pelo coeficiente $\beta_{N:C}$ da Tabela 21 em conjunto com a Figura 29c. Os dois fatores combinados produzem diminuição no custo total ($E_{N:V} = -0.2645$), provavelmente devido ao ganho de tempo quando do aumento do tamanho do *cluster*.

Figura 29 – Análise gráfica de modelo linear para custo de MLP



(a) Ranking de efeitos. (b) Efeitos principais. (c) Efeitos da interação N:V.



(d) Efeitos do custo de execução de MLP.

Fonte: O autor (2020)

A Equação 8.10 explica 96.97% da resposta ($R^2 = 0.9697$ e $R^2_{Adj} = 0.9661$) e estima o custo de execução de MLP para qualquer variação de configuração de *cluster* nas condições investigadas.

9 DISCUSSÃO

> *We have no idea about the 'real' nature of things, and we're not particularly interested in what's 'true'. The function of modeling is to arrive at descriptions which are useful.*
>
> — Richard Bandler and John T. Grinder, *1979, Neurolinguists*, (BANDLER; GRINDER, 1979)

Esta seção discute os resultados dos projetos experimentais descritos nas Seções 7.1 e 8.2 a 8.6. São apresentados os modelos em formato consolidado em quatro diferentes dimensões de análise, além de uma caracterização das técnicas de aprendizagem de máquina em termos de seu comportamento predominante.

9.1 MODELOS CONSOLIDADOS

Na Tabela 22 encontram-se os dados consolidados para o projeto 2^4 fatorial completo relativo à técnica Floresta Aleatória (RF1) sobre dados sintéticos. Na Tabela 23 constam os resultados para os projetos 2^{5-1} fatoriais fracionados relativos às técnicas RF2, LR, SVM, NB e MLP executadas sobre o *Corpus Web*.

Tabela 22 – Modelo RF1 (dados sintéticos)

		Tempo (7.1)		**Custo** (7.2)	
		R^2	R^2_{adj}	R^2	R^2_{adj}
Modelo	MC	97.5	96.3	94.7	92.2
	MS	95.3	95.1	93.2	92.1
	Fatores	VE_x	E_x	VE_x	E_x
Contribuições e Efeitos	N	67.2	-62.02	15.85	29.36
	C	28.11	-46.54	36.45	47.77
	M			4.46	14.64
	D			28.74	41.45
	N:C			3.61	12.97
	N:M			1.41	7.89
	C:D			2.72	-10.05

Fonte: O autor (2020)

Todos os valores encontram-se expressos em quantidades percentuais. As tabelas enumeram o coeficiente de determinação simples (R^2) e ajustado (R^2_{adj}) para os modelos completo

(*MC*), com todos os fatores e interações; e simplificado (*MS*), apenas com os fatores e interações estatisticamente significativos e $\%VE > 1$. A contribuição individual de cada fator X sobre o modelo (VE_X) e a magnitude positiva ou negativa do efeito deste sobre o desempenho (E_X) também são especificadas. Por fim, a notação *(d)* representa os efeitos que não são facilmente interpretáveis devido à dispersão causada por transformações não logarítmicas (Seção 5.5.1), como é o caso do desempenho em termos de tempo para a técnica SVM (raiz quadrada recíproca) ou custo para a técnica NB (raiz quadrada).

Tabela 23 – Modelos LR, SVM, NB, RF2 e MLP (*Corpus* da Língua Portuguesa)

	LR				**SVM**				**NB**				**RF2**				**MLP**			
y	Tempo (8.1)		Custo (8.2)		Tempo (8.3)		Custo (8.4)		Tempo (8.5)		Custo (8.6)		Tempo (8.7)		Custo (8.8)		Tempo (8.9)		Custo (8.10)	
	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}	R^2	R^2_{adj}
MC	93.9	91.1	94.9	92.5	95.3	93.1	98.7	98.0	98.3	97.5	99.6	99.5	99.8	99.8	99.8	99.7	99.5	99.2	99.4	99.1
MS	91.6	89.9	94.4	93.7	94.5	93.5	97.1	96.7	96.9	96.6	97.8	97.5	99.2	99.1	97.3	97.0	97.9	97.8	96.9	96.6
Fator	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x	VE_x	E_x
N	3.80	-9.65	42.86	138.14	8.36	(d)	30.27	103.17	20.90	-25.33	37.71	(d)	8.12	-53.56	2.55	39.53	6.04	-47.60	4.36	52.19
C			29.43	105.26	1.46	(d)	35.67	115.88	5.30	-13.68	36.81	(d)	2.89	-36.72	5.05	59.87	1.56	-28.02	9.25	84.36
M			1.16	15.36							1.26	(d)								
D			16.78	72.11			7.85	43.5			14.93	(d)			3.37	46.69			4.37	52.29
V	77.36	58.11	4.14	30.97	74.08	(d)	21.48	81.7	63.00	-13.68	4.32	(d)	88.17	1153	86.29	5.94	90.31	1116	76.64	482
N:C	1.10	5.61							5.16	66.07										
N:V	5.29	-11.29																	2.34	
C:V					3.54	(d)			2.58											
M:D	2.83	9.16			6.87	(d)	1.03	13.96												

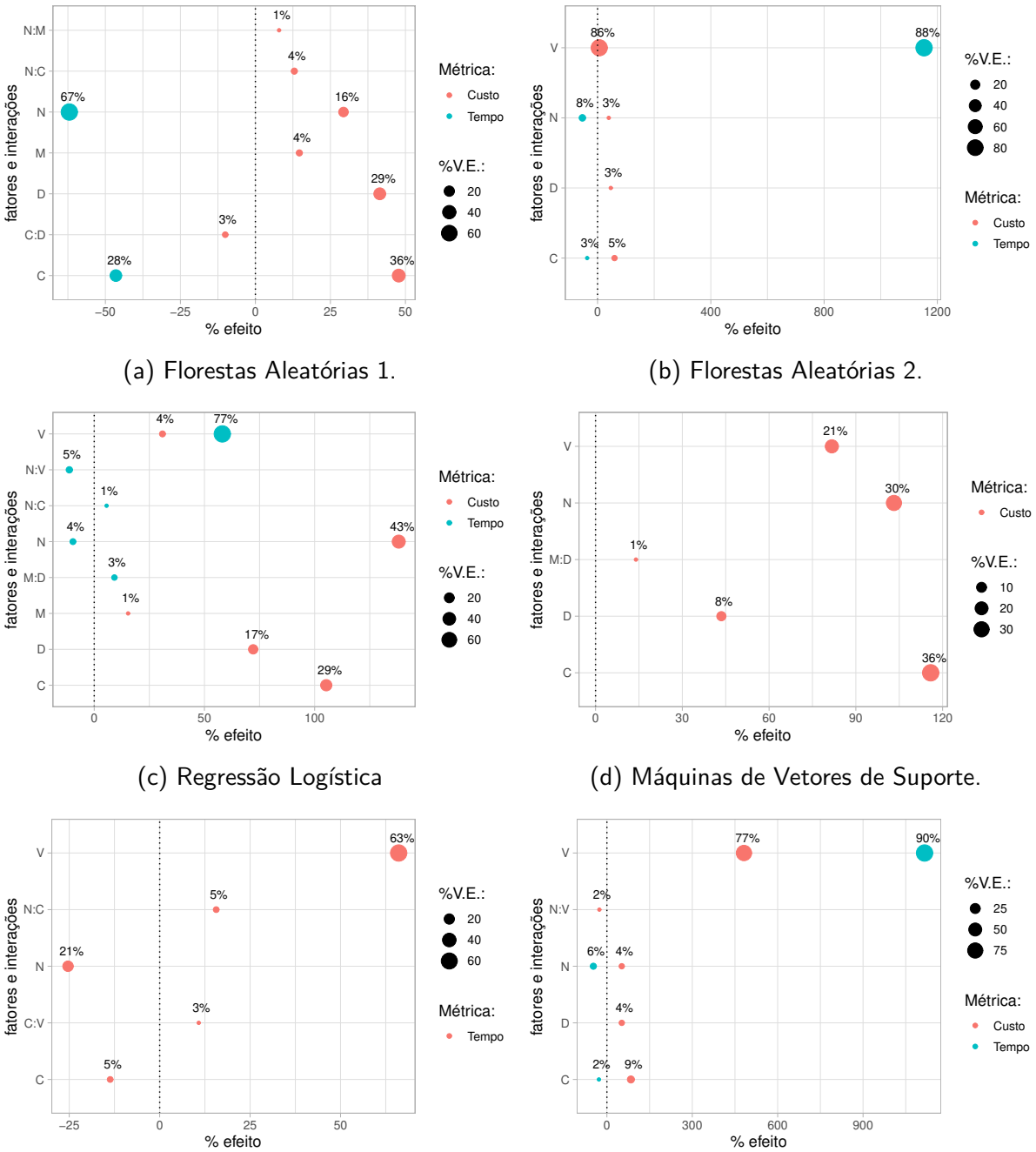
Fonte: O autor (2020)

Os dados consolidados são utilizados nas subseções a seguir para embasar a análise em relação às seguintes perspectivas:

- capacidade de explicação da variação pelo modelo;
- efeito e direção dos fatores sobre o desempenho;
- ordenamento e relevância dos fatores;
- classificação da técnica distribuída em relação ao seu comportamento majoritário.

A Figura 30 exibe os resultados dos modelos em quatro dimensões: fator ou interação significativa, proporção de variação explicada ($\%VE_X$), efeito (E_X) e variável dependente relativa a tempo ou custo. Para compreensão tome-se a Figura 30a relativa ao modelo de tempo para RF1: o número de nós (N) contribui com 67% sobre a explicação do modelo, causando melhoria de -62% sobre o desempenho quando da variação entre níveis extremos (–) e (+). Enquanto a quantidade de núcleos (C) explica 28% do modelo, produzindo ganho de tempo na ordem de -47%.

Figura 30 – Análise gráfica dos efeitos por fator



(a) Florestas Aleatórias 1.

(b) Florestas Aleatórias 2.

(c) Regressão Logística

(d) Máquinas de Vetores de Suporte.

(e) Naïve Bayes

(f) *Perceptron* de Múltiplas Camadas

Fonte: O autor (2020)

9.1.1 Capacidade de Explicação da Variação pelo Modelo (%VE$_X$)

Todos os modelos se ajustaram bem às suas respectivas amostras, com coeficiente de determinação superiores a 90%. A partir da Tabela 23 verifica-se que a perda de poder explicativo dos modelos simplificados (devido à retirada de fatores ou interações não significativas) é muito pequena quando comparada com os modelos completos. Dessa forma, as equações reduzidas possuem maior aplicabilidade prática, por serem parametrizadas apenas com os fatores relevantes, mantendo ainda assim a capacidade de captura da dinâmica das relações que afetam o desempenho.

De forma geral, o fator volume (V) majorou a proporção de explicação do modelo para tempo em todos os experimentos no qual foi inserido. Se removido o volume, a maior explicação se dá pelo número de nós, seja como fator principal ou em face a efeitos de interação com outros fatores. Dessa maneira, os fatores mais relevantes para tempo em termos de sua contribuição explicativa para o modelo são, nessa ordem: V e N.

O custo, por sua vez, não se mostrou comportado, muito embora o volume tenha majorado alguns modelos, a contribuição explicativa individual dos fatores para cada técnica distribuída foi muito variável.

9.1.2 Efeito e Direção dos Fatores sobre o Desempenho (E$_X$)

Destaque-se que a contribuição do fator sobre a explicação do modelo (%VE) e o impacto do fator sobre a resposta (E_X) são valores distintos. O primeiro é detalhado na subseção anterior; para o segundo, os coeficientes parciais de regressão dos modelos quantificam a variação no desempenho causada pela mudança nos níveis do fator. Por exemplo, a Figura 30f exibe o impacto superior a 1000% sobre o tempo e 450% sobre o custo produzido pela execução de MLP face à variação de volume (V) entre os níveis mínimo (–) e máximo (+) de dados de entrada.

Similar à perspectiva de análise $\%VE_X$, o Volume (V) também se mostrou majoritário em relação aos efeitos sobre o tempo, cujo incremento de níveis aumentou o tempo de execução em todos os experimentos em que foi considerado. Sem variação de volume, o maior impacto se deu pelo número de nós do *cluster*. Ou seja: em termos de efeito sobre a resposta, a relevância dos fatores obedece novamente a ordem: V, em seguida N.

Custo não evidenciou padrões, com comportamentos diversos para cada técnica. Foram

observados impactos relevantes causados pelo número de núcleos (para RF1, RF2, SVM), e pelo volume (para MLP). Curiosamente, este último (V) — responsável pelo maior aumento de tempo em um aspecto geral, não foi majoritário em relação ao incremento de custo para a técnica RF2, tendo até mesmo impacto irrisório inferior a 6%.

Como discutido nas seções 8.3 e 8.4, os modelos de (i) tempo para Máquinas de Vetores de Suporte e (ii) custo para *Naïve Bayes* sofrem dispersão dos efeitos entre os coeficientes parciais de regressão devido às peculiaridades de suas transformações de potência, sendo inadequados para inferência individual do impacto dos fatores, mas permanecendo úteis para predição.

9.1.3 Relevância e Ordenamento de Fatores

A perspectiva da importância relativa de cada fator por técnica fica mais evidente na Figura 31, que corrobora as análises anteriores, fornecendo ao tomador de decisão uma visão de relevância e magnitude sobre a relação fator *versus* desempenho. Por razões práticas, são rotulados apenas os fatores com $\%V.E. > 20\%$.

9.1.4 Classificação de Desempenho

A partir do efeito mais relevante pode-se criar uma classificação da técnica de ML distribuída com relação ao fator limitante de desempenho, ou seja, aquele fator que estabelece os limites de variação, conforme Quadro 15, a seguir:

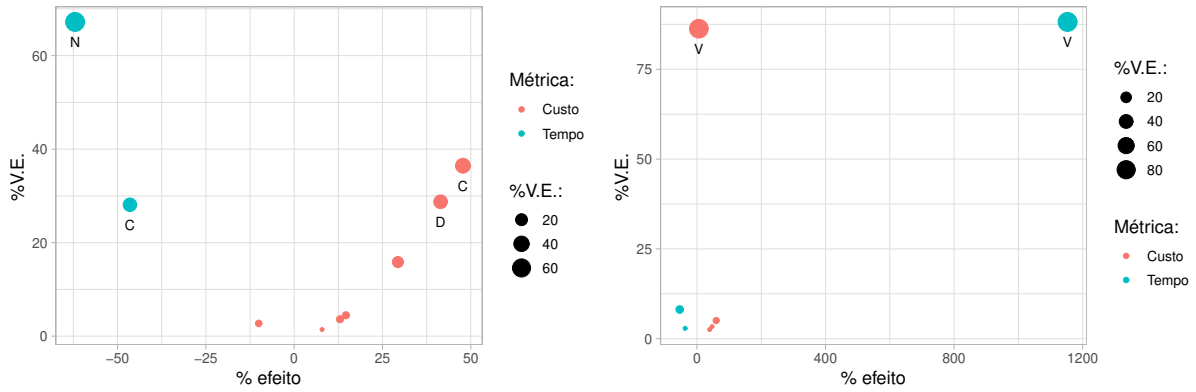
Quadro 15 – Classificação de desempenho por técnica

	Tempo	Custo
Florestas Aleatórias (RF 1 / RF2)	*node-bound*	*core-bound*
Regressão Logística (LR)	*node-bound*	*node-bound*
Máquinas de Vetores de Suporte (SVM)	- -	*core-bound*
Naïve Bayes (NB)	*node-bound*	- -
Perceptron de Múltiplas Camadas (MLP)	*node-bound*	*volume-bound*

Fonte: O autor (2020)

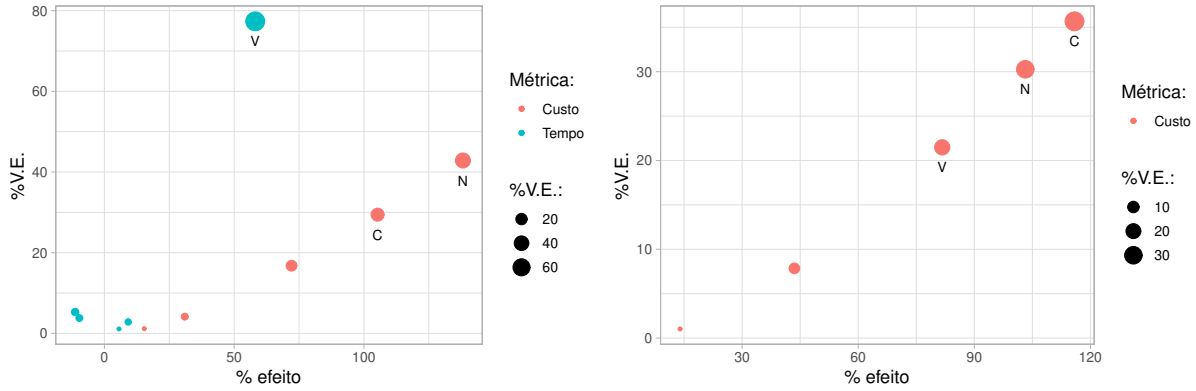
Por *node-bound* compreenda-se que o desempenho de uma determinada técnica de ML distribuída — seja em termos de tempo ou custo, depende majoritariamente da variação do número de nós, em detrimento do número de núcleos de processamento em cada nó do *cluster* ou dos demais fatores investigados, tais como número de discos, quantidade de memória ou

Figura 31 – Representação gráfica do ordenamento e relevância dos fatores



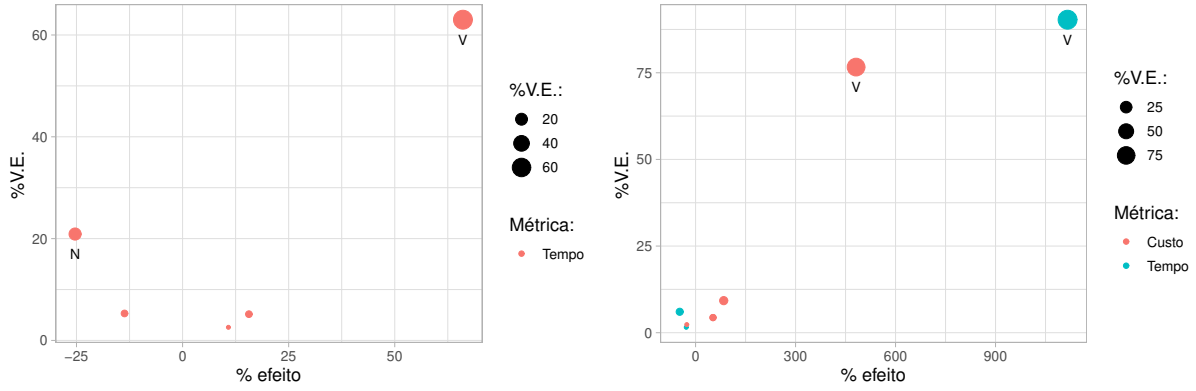
(a) Florestas Aleatórias 1.

(b) Florestas Aleatórias 2.



(c) Regressão Logística

(d) Máquinas de Vetores de Suporte.



(e) Naïve Bayes

(f) *Perceptron* de Múltiplas Camadas

Fonte: O autor (2020)

volume de entrada. Raciocínio análogo deve ser empregado para o significado das classificações *core-bound* (núcleos) e *volume-bound* (tamanho do dado de entrada).

Por exemplo, a diminuição do tempo de execução é limitada pelo número de nós, ou como definido no Quadro 15: *node-bound*. De fato, observou-se que o limite de ganho de eficiência em termos de tempo para todos os algoritmos estudados reside no número de nós do *cluster*, mais até que no número de núcleos de processamento por nó.

Já a classificação por custo, como mencionado anteriormente, não evidenciou comportamento padrão. Por exemplo, o teto do aumento de custos para MLP se dá pelo volume, enquanto RF1, RF2 e SVM são *core-bound*. Apenas LR foi classificado como *core-bound*, ou seja, o número de nós como limitador superior do custo.

Alguns importantes resultados são evidenciados a partir dos experimentos:

- não foi observada uma relação direta entre tempo e custo, ou seja, não foram encontradas evidências de que o aumento de tempo incorrerá em aumento de custo, ou vice-versa. Tal comportamento se deve à uma intricada e complexa relação entre os fatores investigados;
- o número de nós do *cluster* é o fator de maior impacto na diminuição do tempo de execução;
- as técnicas de projeto de experimentos fracionado se mostraram capazes de capturar a dinâmica das relações de variação de desempenho em ambientes de processamento distribuído com alta variabilidade, especialmente relacionadas a tarifação de recursos;
- muito embora o número ideal de réplicas para um projeto fatorial se dê por critérios bem estabelecidos, o número mínimo de replicações $r = 3$, utilizado nesta Tese por limitações de custo, mostrou-se eficiente para derivação de modelos preditivos e de inferência estatística nos cenários investigados.

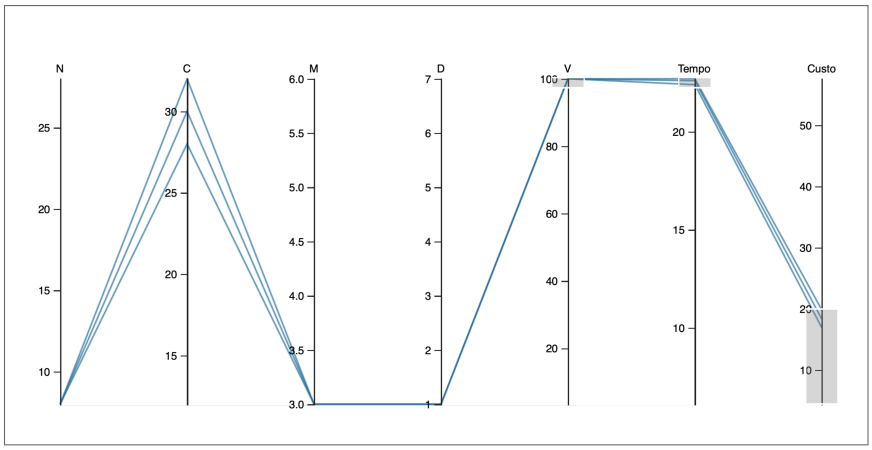
9.2 AJUSTE DE CONFIGURAÇÕES POR COORDENADAS PARALELAS

Um resultado aplicado da pesquisa residiu no desenvolvimento de uma ferramenta de análise dinâmica de desempenho através de coordenadas paralelas (INSELBERG, 1985; INSELBERG, 2009), muito úteis para visualização de problemas multidimensionais em duas dimensões, especialmente no caso em que há muitas instâncias. Como estabelecido na Seção 1 - Introdução, a multiplicidade de fatores e a quantidade de combinações resultam em um problema fatorial

impraticável de exaurir. Por exemplo, no intervalo estudado de 8 a 28 nós, tomados um a um; 12 a 32 núcleos, dois a dois; fator de memória 3x a 6x, a cada 0.5 GB; discos de 1 a 7 e volume de dados tomado a cada 10%, existem 124.506 possibilidades.

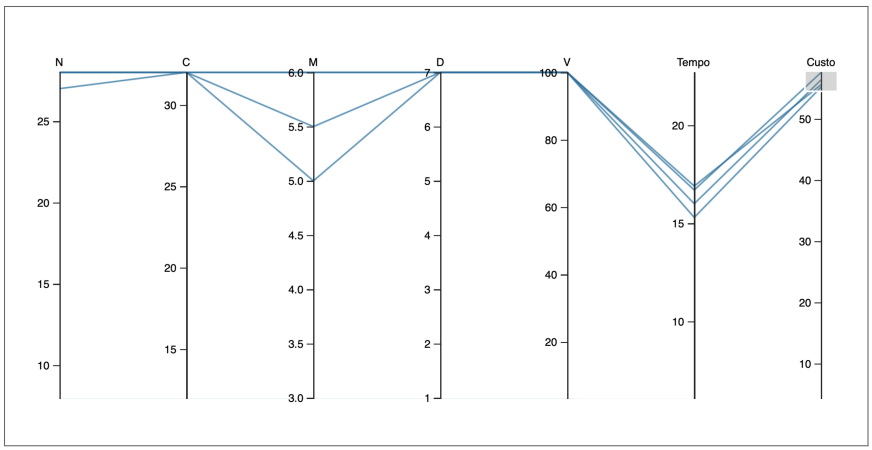
Na sequência constam demonstrações da ferramenta na análise de desempenho da técnica *SVM* a partir dos valores ajustados para os modelos reduzidos de tempo e custo. A Figura 32 apresenta os menores custos possíveis ($\approx \$20$) para executar 100% do volume de dados em um cenário em que o tempo máximo de execução é admissível. Observam-se as opções N = [8], C = [28, 30 ou 32], M = [3x] e D = [1] como alternativas de configuração para execução da tarefa em aproximadamente 22 minutos.

Figura 32 – Coordenadas paralelas para menor custo de SVM, V = 100%



Fonte: O autor (2020)

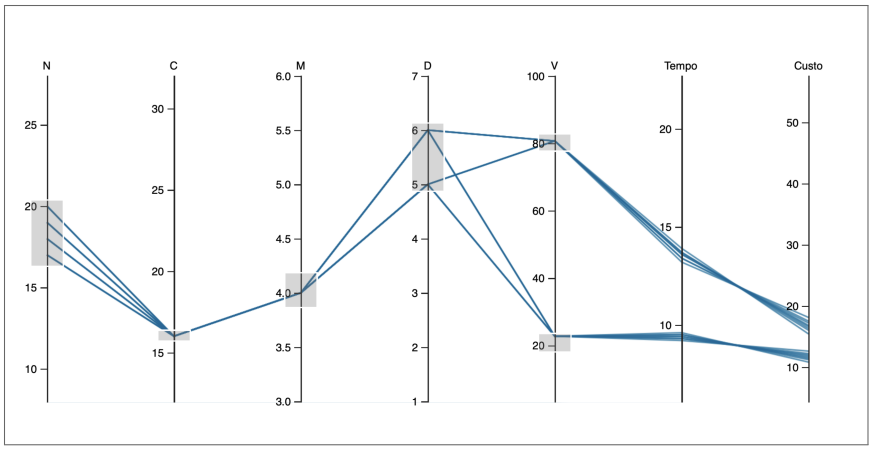
Figura 33 – Coordenadas paralelas para maior custo de SVM, V = 100%



Fonte: O autor (2020)

De maneira análoga, a Figura 33 permite identificar facilmente que o maior custo não implica no menor tempo. Já a Figura 34 exibe a faixa estimada de tempo e custo para um

Figura 34 – Tempo e custo de SVM por composição de fatores



Fonte: O autor (2020)

portfólio N= [16, 17, 18, 19 ou 20], C = [16], M = [4x], D = [5 ou 6], sobre 20% ou 80% do volume total, resultando em cerca de 10 minutos ao custo de $\approx \$12$ ou cerca de 14 minutos ao custo de $\approx \$17$, respectivamente.

A ferramenta possibilita vasto leque de investigação de possibilidades e pode ser configurada para todas as técnicas, bastando informar as combinações por fator e os resultados previstos pelos modelos de tempo e custo.

10 CONCLUSÕES

> *I had not the leisure to make it shorter than it is.*
>
> — Blaise Pascal, *1658, Mathematician*, (PASCAL; EVELYN, 1658)

Esta seção sumariza a Tese ao revisar o problema abordado, detalhando as contribuições científicas e artefatos produzidos, discutindo a aplicabilidade e a extensibilidade dos métodos e resultados, examinando as limitações da pesquisa e delineando investigações e possibilidades futuras.

10.1 VERIFICAÇÃO DE HIPÓTESE

A principal questão da pesquisa foi posta como:

É possível quantificar o impacto isolado de fatores de configuração de *cluster* sobre o desempenho de tempo e custo de tarefas distribuídas em escala *Big Data*, especialmente aquelas de natureza iterativa, tais como aprendizagem de máquina?

Para verificação da hipótese esta Tese investigou o impacto da variação de nós, núcleos de processamento, memória, discos e volume de dados de entrada sobre tempo e custo de execução em *clusters* de máquinas distribuídas. Particularmente, foram utilizados projetos de experimentos fatorial completo e fracionado resolução V (sem confundimento de efeitos até interações de segunda ordem). Como demonstração foram empregadas uma técnica de AM (RF1) sobre 3.65 milhões de instâncias com 40 mil dimensões; e cinco técnicas (LR, SVM, NB, RF2 e MLP) sobre dados reais de 16 milhões de páginas Web com vetores esparsos de 2^{14} dimensões. No total, 288 *clusters* distintos foram criteriosamente organizados para capturar a variação sobre as métricas de desempenho. Foram dispensados esforços metodológicos rigorosos a todas as fases do processo investigatório, com especial atenção à configuração e isolamento das unidades experimentais, à mensuração de tempo e custo, e à aplicação de taxa de câmbio única. Conceitos estatísticos de estruturação, replicação e randomização de experimentos foram cuidadosamente aplicados para garantir análise confiável. Os resultados observados evidenciaram de forma consistente a relevância de DoE para compreensão do consumo de

recursos e estimativa do desempenho de algoritmos de AM em diferentes configurações de ambientes de computação em nuvem para processamento de *Big Data*.

10.2 SUMÁRIO DE CONTRIBUIÇÕES

A pesquisa apresentou uma metodologia capaz de produzir modelos preditores e de inferência estatística aptos a quantificar numericamente, ordenar e selecionar os efeitos relevantes de fatores de configuração sobre o desempenho de técnicas distribuídas de aprendizagem de máquina. A exemplo das Equações 7.1, 7.2 e 8.1-8.10, outros modelos podem ser obtidos a partir de execuções de um plano experimental para estimar tempo e custo de acordo com o orçamento do projeto, dando aos usuários uma nível de confiança estatística. Em um sentido amplo, o método fornece pesos que podem ser úteis para a tomada de decisão na configuração do *cluster*.

Dado que o processamento de *Big Data* tem sido endereçado com tecnologias distribuídas, havendo evidências da aplicação de técnicas de aprendizagem de máquina, sendo ativas as demandas por aprovisionamento de *hardware* e otimização de tempo e custo, as principais contribuições técnico-científicas da pesquisa e da metodologia proposta foram:

- **classificação de técnicas** – definição de precedência de fatores relevantes para cinco técnicas bem difundidas de AM e enquadramento destas segundo seu fator dominante;
- **modelos preditores** – derivação de equações completas e reduzidas capazes de inferir quantitativamente a relevância de fatores de configuração e prever desempenho de tempo e custo para técnicas em escala *Big Data* com intervalos de confiança mensuráveis;
- **suporte à decisão** – ferramenta baseada em coordenadas paralelas para análise, visualização e tomada de decisão sobre os resultados dos modelos preditores nas múltiplas combinações fatoriais possíveis.

10.2.1 Descobertas

Na sequência, observações extraídas da experiência do Autor durante o processo de experimentação e a análise dos resultados no cenário específico investigado:

- normalmente, o fator volume (V) mascara a proporção de variação explicada dos demais fatores sobre o modelo;
- de forma geral, os fatores mais importantes foram volume (V), número de nós (N) e núcleos de processamento (C), nessa ordem;
- há evidências de interações acima da segunda ordem possuem pouca utilidade prática e o pesquisador pode considerar projetos fracionados de menor custo, negligenciando ordens superiores;
- mesmo as interações de segunda ordem mostraram pouca relevância, sendo significativas apenas as combinações N:C, N:V, C:V e M:D, de dez possíveis, com pequena contribuição sobre o coeficiente de determinação do modelo;
- 40% dos experimentos são influenciados apenas pelos fatores principais (sem interações de nenhuma ordem);
- 80% dos experimentos evidenciaram distribuição logarítmica para tempo e custo;
- não foi possível quantificar o efeito individual de fatores para o desempenho de tempo de SVM e de custo para NB, devido à dispersão de efeitos nos coeficientes parciais de regressão dadas as transformações não logarítmicas;
- o nível de replicação ($r = 3$) mostrou-se viável para derivar modelos com alto coeficiente de determinação ($> 90\%$) sobre a variação da resposta;
- a incerteza de disponibilidade de nós preemptivos em *clusters* efêmeros não inviabilizou a capacidade de modelagem e inferência estatística.

10.3 PÚBLICO-ALVO E APLICABILIDADE DA PESQUISA

Os métodos e resultados desta Tese são de interesse de cientistas de dados e administradores de recursos em nuvem, sob a perspectiva de aprovisionamento de recursos. O público-alvo pode ser estendido a projetistas de técnicas distribuídas, uma vez que o ordenamento de fatores de impacto pode direcionar decisões de implementação.

Ressalte-se que a derivação de modelos carece de ampla gama de unidades experimentais para que as evidências sejam consideradas seguras. Assim, a aplicação dos métodos faz sentido

em cenários nos quais a técnica é executada várias vezes, em ambientes de computação em nuvem tarifados por fatia de tempo; ou em ambientes com *hardware* físico a ser montado especificamente para o seu processamento. Nestes casos, o esforço, empregado no processo experimental compensaria execuções subsequentes.

É possível citar, ao menos, três aplicações diretas da metodologia. Primeiramente, o uso do modelo preditivo no contexto estudado para descoberta da dinâmica de execução, permitindo a configuração de *hardware* e arquitetura do *cluster* para uso ótimo de recursos no caso concreto. Nesta visão, a condução do processo experimental caberia ao usuário interessado.

O segundo exemplo reside na oferta de modelos de negócio para precificação justa em provedores de serviço computacionais. Uma vez que estes possuem parque computacional de alta capacidade, quando submetida uma tarefa pelo usuário, o provedor lançaria o processo experimental em paralelo, oferecendo proposta de utilização de recursos em cenários ótimos. Nesta visão, a condução do processo experimental caberia ao provedor. Múltiplos outros fatores, tais como GPU, largura de banda de rede interna, tipo de armazenamento e parâmetros de *software* [1] podem ser incluídos, introduzindo maior capacidade de sugestão de carteiras boas ou ótimas.

Em terceiro lugar, os resultados podem servir de base estimada para problemas similares. Embora os modelos garantam intervalo de confiança de 95% apenas para o contexto executado; intuitivamente espera-se que a ordem relativa dos fatores, especialmente aqueles de maior impacto, se aproxime de outros contextos para a mesma técnica. Portanto, há a hipótese de que as cinco técnicas apresentadas possuam dinâmicas similares em diferentes situações.

10.4 EXTENSIBILIDADE DO MÉTODO

A natureza agnóstica da metodologia — baseada em fatores de *hardware* e volume de dados, torna-a um método reproduzível, que pode ser aplicado em outros algoritmos de aprendizagem de máquina, em análises de grafos e tarefas de consulta sobre *Big Data*, tanto na plataforma *Spark* quanto em outras estruturas de processamento ou outros provedores de serviços em nuvem. Os administradores de *cluster* podem planejar o consumo de recursos com base nos efeitos mais significativos do algoritmo sob investigação. Variações sobre outras métricas de desempenho (variável dependente) também podem ser investigadas. Em um sentido

[1] Tais como na plataforma *Spark*: quantidade de tarefas por nó, grau de paralelismo, tamanho do bloco distribuído, limite de embaralhamento de dados *shuffle*, dentre tantas outras.

amplo, o método fornece pesos para os fatores que podem ser úteis para a tomada de decisão na configuração do *cluster*.

10.5 LIMITAÇÕES

Entre as principais limitações da pesquisa, citam-se:

- **sensibilidade ao contexto** [2] – a dinâmica de execução é particular ao problema que este endereça, aos hiperparâmetros para treinamento dos modelos, às características dos dados — volume, localidade no nó do *cluster*, número de instâncias e dimensionalidade, tipos de dados, distribuição de valores; dentre outros;
- **volatilidade monetária** – as flutuações de precificação de recurso, devido a oferta e demanda de mercado, tornam os modelos de custo muito específicos para o momento e a plataforma investigada;
- **dependência de implementação algorítmica** - embora o método seja extensível e aplicável a qualquer tarefa de análise em lote sobre *Big Data*, os modelos gerados possui dependência estrita da implementação computacional distribuída e da plataforma utilizada (*Hadoop*, *Spark*, *Storm* etc);
- **número de replicações** – é prática comum e estabelecida a utilização de limite mínimo de três replicações para detecção de variação. Todavia, o cálculo da quantidade exata é específico de cada experimento, ambiente ou problema; e demanda, por si só, experimentação científica. A determinação do número ideal de réplicas esteve fora do escopo de pesquisa.

10.6 TRABALHOS FUTUROS

A pesquisa pode ser ampliada nas seguintes linhas:

- **extensão dos tipos de projeto** - adição de pontos centrais e axiais no plano experimental para detecção de curvaturas no modelo e construção de superfícies de resposta para desenvolvimento de modelos de otimização;

[2] No caso concreto, os resultados evidenciaram os efeitos para uma base de dados sintética e um *Corpus* da Língua Portuguesa. Outros problemas, sobre outros conjuntos de dados podem resultar em diferentes efeitos e relevância de fatores.

- **adição de novas variáveis dependentes** - o estudo se concentrou nas variáveis tempo e custo de execução. Todavia, quaisquer outras métricas de desempenho podem ser incluídas, tais como o tempo gasto pelo coletor de lixo da memória, volume de tráfego de rede e carga de processos no sistema operacional;

- **ampliação do conjunto de variáveis independentes** - os parâmetros de desempenho podem ser testados com projetos maiores, adicionando-se novos fatores de *hardware*, tais como GPU; ou configurações de *framework*, como o grau de paralelismo de operações *shuflle* entre nós, o tamanho da partição dos dados, a quantidade de processos executores por nó, dentre outros;

- **expansão do conjunto de técnicas distribuídas** - uma vez que a Tese se concentrou em cinco técnicas de aprendizado de máquina, outros experimentos se fazem necessários sobre demais algoritmos, inclusive sobre diferentes classes de problemas, tais como clusterização e associação de regras, e mesmo ainda, tarefas computacionais além da aprendizagem de máquina;

- **desenvolvimento de ferramenta de suporte** - embora a metodologia possua aplicação direta, com fácil assimilação dos princípios, as fases envolvem definição de plano, cálculo de resolução, análise da variação, verificação de aderência de premissas, além da construção e simplificação de modelos. Nesse contexto, uma importante contribuição futura consistiria na criação de uma ferramenta com interface gráfica e fluxos de execução para abstrair e conduzir o utilizador por estas fases;

- **investigação de outros provedores e plataformas** - todas as unidades experimentais foram testadas em ambiente de nuvem da empresa *Google*. Outros provedores, a exemplo de *Amazon AWS* ou *Microsoft Azure* podem ser investigados em busca de confrontar ou reforçar as hipóteses aqui levantadas;

- **validação da extensibilidade** – a metodologia possui bases teóricas bem estabelecidas, havendo amplo suporte de ferramentas de *software* proprietárias ou de código-aberto. Presume-se que a proposta pode ser facilmente compreendida, adotada e replicada, seguidos os passos apresentados na Seção 6 - Metodologia. Para tal, demanda-se a investigação da extensibilidade da metodologia, com base em aplicação e validação da mesma por terceiros em situações diversas.

REFERÊNCIAS

ABRAHAM, T. H. (physio) logical circuits: The intellectual origins of the mcculloch–pitts neural networks. *Journal of the History of the Behavioral Sciences*, Wiley Online Library, v. 38, n. 1, p. 3–25, 2002.

AGRAWAL, D.; BUTT, A.; DOSHI, K.; LARRIBA-PEY, J. L.; LI, M.; REISS, F. R.; RAAB, F.; SCHIEFER, B.; SUZUMURA, T.; XIA, Y. Sparkbench – A spark performance testing suite. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 9508, p. 26–44, 2016. ISSN 16113349.

ALHAM, N. K.; LI, M.; LIU, Y. Parallelizing multiclass support vector machines for scalable image annotation. *Neural Computing and Applications*, Springer, v. 24, n. 2, p. 367–381, 2014.

AMANULLAH, M. A.; HABEEB, R. A. A.; NASARUDDIN, F. H.; GANI, A.; AHMED, E.; NAINAR, A. S. M.; AKIM, N. M.; IMRAN, M. Deep learning and big data technologies for iot security. *Computer Communications*, Elsevier, 2020.

AMIN, M. M.; KIANI, A. Multi-disciplinary analysis of a strip stabilizer using body-fluid-structure interaction simulation and design of experiments (doe). *Journal of Applied Fluid Mechanics*, v. 13, n. 1, p. 261–273, 2020.

AMINI, M. M.; BARR, R. S. Network reoptimization algorithms: A statistically designed comparison. *ORSA Journal on Computing*, INFORMS, v. 5, n. 4, p. 395–409, 1993.

ANDERSON, M. J.; KRABER, S. L. Keys to successful designed experiments. In: *American Society for Quality Conference*. [S.l.: s.n.], 1999. v. 11, p. 1–6.

ARASANAL, R. M.; RUMANI, D. U. Improving mapreduce performance through complexity and performance based data placement in heterogeneous hadoop clusters. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. [S.l.: s.n.], 2013. ISBN 9783642360701. ISSN 03029743.

ARDAGNA, D.; BERNARDI, S.; GIANNITI, E.; ALIABADI, S. K.; PEREZ-PALACIN, D.; REQUENO, J. I. Modeling performance of hadoop applications: A journey from queueing networks to stochastic well formed nets. In: SPRINGER. *International Conference on Algorithms and Architectures for Parallel Processing*. [S.l.], 2016. p. 599–613.

BAEVSKI, A.; EDUNOV, S.; LIU, Y.; ZETTLEMOYER, L.; AULI, M. Cloze-driven pretraining of self-attention networks. *arXiv preprint arXiv:1903.07785*, 2019.

BAHMANI, B.; MOSELEY, B.; VATTANI, A.; KUMAR, R.; VASSILVITSKII, S. Scalable K-Means++. 2012. ISSN 2150-8097. Disponível em: <http://arxiv.org/abs/1203.6402>.

BAKER, M.; BUYYA, R. Cluster computing: The commodity supercomputing. *SOFTWARE—PRACTICE AND EXPERIENCE*, v. 1, n. 1, p. 1–4, 1988.

BALDACCI, L.; GOLFARELLI, M. A cost model for spark sql. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 31, n. 5, p. 819–832, 2018.

BALESTRASSI, P. P.; POPOVA, E.; PAIVA, A. d.; LIMA, J. M. Design of experiments on neural network's training for nonlinear time series forecasting. *Neurocomputing*, Elsevier, v. 72, n. 4-6, p. 1160–1178, 2009.

BANDLER, R.; GRINDER, J. Frogs into princes, utah. *Real People Press. Barnier, AJ, & McConkey, KM (1992). Reports of real and false memories: The relevance of hypnosis, hypnotizability, and context of memory test. Journal of Abnormal Psychology*, v. 101, n. 3, p. 521–527, 1979.

BARR, R. S.; GOLDEN, B. L.; KELLY, J. P.; RESENDE, M. G.; STEWART, W. R. Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, v. 1, n. 1, p. 9–32, 1995. ISSN 13811231.

BATES, S.; SIENZ, J.; TOROPOV, V. Formulation of the optimal latin hypercube design of experiments using a permutation genetic algorithm. In: *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*. [S.l.: s.n.], 2004. p. 2011.

BAYES, T. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, The Royal Society London, n. 53, p. 370–418, 1763.

BELSON, W. A. A technique for studying the effects of a television broadcast. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, Wiley Online Library, v. 5, n. 3, p. 195–202, 1956.

BORTHAKUR, D. The hadoop distributed file system: Architecture and design. *Hadoop Project Website*, v. 11, n. 2007, p. 21, 2007.

BOSTOCK, M.; CHANG, K.; YUN, X.; RUSSELL, K.; ODISHO, A. *parcoords: 'Htmlwidget' for 'd3.js' Parallel Coordinates Chart*. [S.l.], 2019. R package version 1.0.0. Disponível em: <https://CRAN.R-project.org/package=parcoords>.

BOX, G. E. All models are wrong, but some are useful. *Robustness in Statistics*, v. 202, p. 549, 1979.

BOX, G. E.; COX, D. R. An analysis of transformations. *Journal of the Royal Statistical Society: Series B (Methodological)*, Wiley Online Library, v. 26, n. 2, p. 211–243, 1964.

BOYD, D.; CRAWFORD, K. Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon. *Information, communication & society*, Taylor & Francis, v. 15, n. 5, p. 662–679, 2012.

BREIMAN, L. Random forests. *UC Berkeley TR567*, 1999.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

BREIMAN, L.; FRIEDMAN, J.; STONE, C. J.; OLSHEN, R. A. *Classification and regression trees*. [S.l.]: CRC press, 1984.

BROWN, J.; RODGERS, T. *Doing Second Language Research: An Introduction to the Theory and Practice of Second Language Research for Graduate/Master's Students in TESOL and Applied Linguistics, and Others*. [S.l.]: OUP Oxford, 2002. (Oxford English). ISBN 9780194371742.

BU, Y.; HOWE, B.; BALAZINSKA, M.; ERNST, M. D. Haloop: Efficient iterative data processing on large clusters. *Proceedings of the VLDB Endowment*, VLDB Endowment, v. 3, n. 1-2, p. 285–296, 2010.

BUCK, C.; HEAFIELD, K.; OOYEN, B. V. N-gram counts and language models from the common crawl. In: CITESEER. *LREC*. [S.l.], 2014. v. 2, p. 4.

CARBONE, P.; KATSIFODIMOS, A.; EWEN, S.; MARKL, V.; HARIDI, S.; TZOUMAS, K. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, IEEE Computer Society, v. 36, n. 4, 2015.

CASLER, M. D. Fundamentals of experimental design: Guidelines for designing successful experiments. *Agronomy Journal*, The American Society of Agronomy, Inc., v. 107, n. 2, p. 692–705, 2015.

CHAMBERS, J. M.; CLEVELAND, W. S.; KLEINER, B.; TUKEY, P. A. Graphical methods for data analysis. wadsworth & brooks. *Cole Statistics/Probability Series*, 1983.

CHEN, K.; POWERS, J.; GUO, S.; TIAN, F. CRESP: Towards Optimal Resource Provisioning for MapReduce Computing in Public Clouds. *IEEE Transactions on Parallel and Distributed Systems*, v. 25, n. 6, p. 1403–1412, jun 2014. ISSN 1045-9219. Disponível em: <http://ieeexplore.ieee.org/document/6678508/>.

CHU, C.-T.; KIM, S. K.; LIN, Y.-A.; YU, Y.; BRADSKI, G.; NG, A. Y.; OLUKOTUN, K. Map-reduce for machine learning on multicore. In: VANCOUVER, BC. *NIPS*. [S.l.], 2006. v. 6, p. 281–288.

COHEN, J.; COHEN, P.; WEST, S. G.; AIKEN, L. S. *Applied multiple regression/correlation analysis for the behavioral sciences*. [S.l.]: Routledge, 2013.

CONFORTO, E. C.; AMARAL, D. C.; SILVA, S. L. Roteiro para revisão bibliográfica sistemática: aplicação no desenvolvimento de produtos e gerenciamento de projetos. In: *8º Congresso Brasileiro de Gestão de Desenvolvimento de Produto-CBGDP*. [S.l.: s.n.], 2011.

COOK, R. D. Detection of influential observation in linear regression. *Technometrics*, Taylor & Francis Group, v. 19, n. 1, p. 15–18, 1977.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine learning*, Springer, v. 20, n. 3, p. 273–297, 1995.

COX, M.; ELLSWORTH, D. Managing big data for scientific visualization. In: *ACM Siggraph*. [S.l.: s.n.], 1997. v. 97, p. 146–162.

CRAMER, J. S. The origins and development of the logit model. *Logit models from economics and other fields*, Cambridge University Press Cambridge, v. 2003, p. 1–19, 2003.

DEAN, J.; GHEMAWAT, S. Mapreduce: simplified data processing on large clusters. In: USENIX ASSOCIATION. *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation-Volume 6*. [S.l.], 2004. p. 10–10.

DEAN, J.; GHEMAWAT, S. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, ACM, v. 51, n. 1, p. 107, 2008. ISSN 00010782. Disponível em: <http://portal.acm.org/citation.cfm?doid=1327452.1327492>.

DHARSANDIYA, A. N.; PATEL, M. R. A review on frequent itemset mining algorithms in social network data. In: IEEE. *Wireless Communications, Signal Processing and Networking (WiSPNET), International Conference on*. [S.l.], 2016. p. 1046–1048.

DOMINGOS, P. A few useful things to know about machine learning. *Communications of the ACM*, ACM New York, NY, USA, v. 55, n. 10, p. 78–87, 2012.

DREISEITL, S.; OHNO-MACHADO, L. Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics*, v. 35, n. 5-6, p. 352–359, oct 2002. ISSN 15320464. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1532046403000340>.

DURAKOVIC, B. Design of experiments application, concepts, examples: State of the art. *Periodicals of Engineering and Natural Sciences*, v. 5, n. 3, 2017.

DUTTA, S.; GANDOMI, A. H. Chapter 15 - design of experiments for uncertainty quantification based on polynomial chaos expansion metamodels. In: SAMUI, P.; BUI], D. T.; CHAKRABORTY, S.; DEO, R. C. (Ed.). *Handbook of Probabilistic Models*. Butterworth-Heinemann, 2020. p. 369 – 381. ISBN 978-0-12-816514-0. Disponível em: <http://www.sciencedirect.com/science/article/pii/B9780128165140000151>.

EKANAYAKE, J.; LI, H.; ZHANG, B.; GUNARATHNE, T.; BAE, S.-H.; QIU, J.; FOX, G. Twister: a runtime for iterative mapreduce. In: *Proceedings of the 19th ACM international symposium on high performance distributed computing*. [S.l.: s.n.], 2010. p. 810–818.

EL-ALFY, E.-S. M.; ALSHAMMARI, M. A. Towards scalable rough set based attribute subset selection for intrusion detection using parallel genetic algorithm in mapreduce. *Simulation Modelling Practice and Theory*, Elsevier, v. 64, p. 18–29, 2016.

EYUPOGLU, C. Big data processing: Concepts, architectures, technologies, and techniques. In: *Applications and Approaches to Object-Oriented Software Design: Emerging Research and Opportunities*. [S.l.]: IGI Global, 2020. p. 111–132.

FAHAD, A.; ALSHATRI, N.; TARI, Z.; ALAMRI, A.; KHALIL, I.; ZOMAYA, A. Y.; FOUFOU, S.; BOURAS, A. A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis. *IEEE Transactions on Emerging Topics in Computing*, v. 2, n. 3, p. 267–279, sep 2014. ISSN 2168-6750. Disponível em: <http://ieeexplore.ieee.org/document/6832486/>.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37, 1996.

FISCHER, L.; GAO, S.; BERNSTEIN, A. Machines tuning machines: Configuring distributed stream processors with bayesian optimization. In: IEEE. *2015 IEEE International Conference on Cluster Computing*. [S.l.], 2015. p. 22–31. ISSN 1552-5244.

FISHER, D.; DELINE, R.; CZERWINSKI, M.; DRUCKER, S. Interactions with big data analytics. *interactions*, ACM, v. 19, n. 3, p. 50–59, 2012.

FISHER, R. A. The design of experiments (hafner). *New York*, 1935.

FISHER, R. A.; WISHART, J. *The arrangement of field experiments and the statistical reduction of the results*. [S.l.]: HM Stationery Office, 1930.

FOX, A.; GRIFFITH, R.; JOSEPH, A.; KATZ, R.; KONWINSKI, A.; LEE, G.; PATTERSON, D.; RABKIN, A.; STOICA, I. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, v. 28, n. 13, p. 2009, 2009.

FOX, J.; WEISBERG, S. *An R Companion to Applied Regression*. Third. Thousand Oaks CA: Sage, 2019. Disponível em: <https://socialsciences.mcmaster.ca/jfox/Books/Companion/>.

FREIESLEBEN, J.; KEIM, J.; GRUTSCH, M. Machine learning and design of experiments: Alternative approaches or complementary methodologies for quality improvement? *Quality and Reliability Engineering International*, Wiley Online Library, 2020.

GANTZ, J.; REINSEL, D. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the future*, v. 2007, n. 2012, p. 1–16, 2012.

GELMAN, A.; HILL, J. *Data analysis using regression and multilevel/hierarchical models*. [S.l.]: Cambridge university press, 2006.

GENUER, R.; POGGI, J.-M.; TULEAU-MALOT, C.; VILLA-VIALANEIX, N. Random forests for big data. *Big Data Research*, Elsevier, v. 9, p. 28–46, 2017.

GHEMAWAT, S.; GOBIOFF, H.; LEUNG, S.-T. The google file system. In: *Proceedings of the nineteenth ACM symposium on Operating systems principles*. [S.l.: s.n.], 2003. p. 29–43.

GIANNITI, E.; RIZZI, A. M.; BARBIERATO, E.; GRIBAUDO, M.; ARDAGNA, D. Fluid Petri nets for the performance evaluation of MapReduce applications. *ValueTools 2016 - 10th EAI International Conference on Performance Evaluation Methodologies and Tools*, v. 44, n. 4, p. 243–250, 2017.

GOLDBERG, D.; NICHOLS, D.; OKI, B. M.; TERRY, D. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, ACM, v. 35, n. 12, p. 61–70, 1992.

GOMES, D.; MIRANDA, J.; COSTA, M. A survey on web archiving initiatives. In: SPRINGER. *International Conference on Theory and Practice of Digital Libraries*. [S.l.], 2011. p. 408–420.

GRAVE, E.; BOJANOWSKI, P.; GUPTA, P.; JOULIN, A.; MIKOLOV, T. Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893*, 2018.

GRÖNMPING, U. R package frf2 for creating and analyzing fractional factorial 2-level designs. *Journal of Statistical Software*, v. 56, n. 1, p. 1–56, 2014.

GROOM, W. *Forrest Gump*. Garden City, N.Y: Doubleday, 1986. ISBN 0385231342.

GU, R.; SHEN, F.; HUANG, Y. A parallel computing platform for training large scale neural networks. In: IEEE. *Big Data, 2013 IEEE International Conference on*. [S.l.], 2013. p. 376–384.

GUECIOUER, D.; YOUCEF, G.; TAREK, N. Rheological and mechanical optimization of a steel fiber reinforced self-compacting concrete using the design of experiments method. *European Journal of Environmental and Civil Engineering*, Taylor & Francis, v. 0, n. 0, p. 1–21, 2019. Disponível em: <https://doi.org/10.1080/19648189.2019.1697758>.

HAN, R.; XIAOYI, L.; JIANGTAO, X. On big data benchmarking. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 8807, n. 1, p. 3–18, 2014. ISSN 16113349.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. [S.l.]: Springer New York, 2013. (Springer Series in Statistics). ISBN 9780387216065.

HAWKINS, D. M. *Topics in applied multivariate analysis*. [S.l.]: Cambridge University Press, 1982. v. 1.

HERODOTOU, H.; LIM, H.; LUO, G.; BORISOV, N.; DONG, L.; CETIN, F. B.; BABU, S. Starfish: A self-tuning system for big data analytics. In: *Cidr*. [S.l.: s.n.], 2011. v. 11, n. 2011, p. 261–272.

HEY, T.; TANSLEY, S.; TOLLE, K. M. et al. *The fourth paradigm: data-intensive scientific discovery*. [S.l.]: Microsoft research Redmond, WA, 2009. v. 1.

HOOKER, J. Testing heuristics: we have it all wrong. journal of heuristics, 1. 33 42. doi: 10.1007. *BF02430364*, 1995.

HOSSAIN, M. S.; MUHAMMAD, G. Emotion recognition using deep learning approach from audio–visual emotional big data. *Information Fusion*, Elsevier, v. 49, p. 69–78, 2019.

HUAI, Y.; LEE, R.; ZHANG, S.; XIA, C. H.; ZHANG, X. Dot: a matrix model for analyzing, optimizing and deploying software for big data analytics in distributed systems. In: ACM. *Proceedings of the 2nd ACM Symposium on Cloud Computing*. [S.l.], 2011. p. 4.

HUANG, S.; HUANG, J.; DAI, J.; XIE, T.; HUANG, B. The HiBench benchmark suite: Characterization of the MapReduce-based data analysis. In: *2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010)*. IEEE, 2010. p. 41–51. ISBN 978-1-4244-6522-4. Disponível em: <http://ieeexplore.ieee.org/document/5452747/>.

HUANG, S.; HUANG, J.; DAI, J.; XIE, T.; HUANG, B. The hibench benchmark suite: Characterization of the mapreduce-based data analysis. In: *New Frontiers in Information and Software as Services*. [S.l.]: Springer, 2011. p. 209–228.

HUSH, D. R.; HORNE, B. G. Progress in supervised neural networks. *IEEE signal processing magazine*, IEEE, v. 10, n. 1, p. 8–39, 1993.

INSELBERG, A. The plane with parallel coordinates. *The visual computer*, Springer, v. 1, n. 2, p. 69–91, 1985.

INSELBERG, A. *Parallel coordinates: visual multidimensional geometry and its applications*. [S.l.]: Springer Science & Business Media, 2009. v. 20.

IQBAL, M. H.; SOOMRO, T. R. Big data analysis: Apache storm perspective. *International journal of computer trends and technology*, v. 19, n. 1, p. 9–14, 2015.

ISO, I. 28500: 2009 information and documentation-warc file format. *International Organization for Standardization*, 2009.

JAIN, A. K. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, Elsevier B.V., v. 31, n. 8, p. 651–666, 2010. ISSN 01678655. Disponível em: <http://dx.doi.org/10.1016/j.patrec.2009.09.011>.

JAIN, R. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. [S.l.]: Wiley & Sons, 1990.

JAN, B.; FARMAN, H.; KHAN, M.; IMRAN, M.; ISLAM, I. U.; AHMAD, A.; ALI, S.; JEON, G. Deep learning in big data analytics: a comparative study. *Computers & Electrical Engineering*, Elsevier, v. 75, p. 275–287, 2019.

JIANG, X.; SUN, G. MapReduce-based frequent itemset mining for analysis of electronic evidence. In: *2013 8th International Workshop on Systematic Approaches to Digital Forensics Engineering (SADFE)*. [s.n.], 2013. p. 1–6. ISBN 978-1-4799-4061-5. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6911549>.

JOSEFSSON, S. et al. *The base16, base32, and base64 data encodings*. [S.l.], 2006.

JOY, R.; SHERLY, K. Parallel frequent itemset mining with spark rdd framework for disease prediction. In: IEEE. *Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on*. [S.l.], 2016. p. 1–5.

JR, D. W. H.; LEMESHOW, S.; STURDIVANT, R. X. *Applied logistic regression*. [S.l.]: John Wiley & Sons, 2013. v. 398.

KHAN, S.; IQBAL, K.; FAIZULLAH, S.; FAHAD, M.; ALI, J.; AHMED, W. Clustering based privacy preserving of big data using fuzzification and anonymization operation. *arXiv preprint arXiv:2001.01491*, 2020.

KHAN, S.; KHAN, A.; MAQSOOD, M.; AADIL, F.; GHAZANFAR, M. A. Optimized gabor feature extraction for mass classification using cuckoo search for big data e-healthcare. *Journal of Grid Computing*, Springer, v. 17, n. 2, p. 239–254, 2019.

KIM, S.-B.; HAN, K.-S.; RIM, H.-C.; MYAENG, S. H. Some effective techniques for naive bayes text classification. *IEEE transactions on knowledge and data engineering*, IEEE, v. 18, n. 11, p. 1457–1466, 2006.

KINT, J.; CONSTALES, D.; VANDERBAUWHEDE, A. Pierre-françois verhulst's final triumph. In: *The Logistic Map and the Route to Chaos*. [S.l.]: Springer, 2006. p. 13–28.

KOVACS, F.; ILLÉS, J. Frequent itemset mining on hadoop. In: IEEE. *Computational Cybernetics (ICCC), 2013 IEEE 9th International Conference on*. [S.l.], 2013. p. 241–245.

KUHN, M.; JOHNSON, K. *Applied predictive modeling*. [S.l.]: Springer, 2013. v. 26.

KUO, C.-C.; LIU, H.-A.; CHANG, C.-M. Optimization of vacuum casting process parameters to enhance tensile strength of components using design of experiments approach. *The International Journal of Advanced Manufacturing Technology*, Springer, p. 1–11, 2020.

LACHIHEB, O.; GOUIDER, M. S.; SAID, L. B. An improved mapreduce design of kmeans with iteration reducing for clustering stock exchange very large datasets. In: IEEE. *Semantics, Knowledge and Grids (SKG), 2015 11th International Conference on*. [S.l.], 2015. p. 252–255.

LAI, C.-F.; CHANG, J.-H.; HU, C.-C.; HUANG, Y.-M.; CHAO, H.-C. Cprs: A cloud-based program recommendation system for digital tv platforms. *Future Generation Computer Systems*, Elsevier, v. 27, n. 6, p. 823–835, 2011.

LANDSET, S.; KHOSHGOFTAAR, T. M.; RICHTER, A. N.; HASANIN, T. A survey of open source tools for machine learning with big data in the Hadoop ecosystem. *Journal of Big Data*, v. 2, n. 1, p. 24, 2015. ISSN 2196-1115. Disponível em: <http://www.journalofbigdata.com/content/2/1/24>.

LANEY, D. 3d data management: Controlling data volume, velocity and variety. *META Group Research Note*, v. 6, n. 70, 2001.

LAUGHTON, M. A.; SAY, M. G. *Electrical engineer's reference book*. [S.l.]: Elsevier, 2013.

LAWSON, J. *Design and Analysis of Experiments with R*. [S.l.]: Chapman and Hall/CRC, 2014.

LEMNARU, C.; CUIBUS, M.; BONA, A.; ALIC, A.; POTOLEA, R. A distributed methodology for imbalanced classification problems. In: IEEE. *Parallel and Distributed Computing (ISPDC), 2012 11th International Symposium on*. [S.l.], 2012. p. 164–171.

LEVY, Y.; ELLIS, T. J. A systems approach to conduct an effective literature review in support of information systems research. *Informing Science: International Journal of an Emerging Transdiscipline*, Citeseer, v. 9, n. 1, p. 181–212, 2006.

LI, K.; DEOLALIKAR, V.; PRADHAN, N. Big data gathering and mining pipelines for crm using open-source. In: IEEE. *Big Data (Big Data), 2015 IEEE International Conference on*. [S.l.], 2015. p. 2936–2938.

LI, M.; TAN, J.; WANG, Y.; ZHANG, L.; SALAPURA, V. Sparkbench: a comprehensive benchmarking suite for in memory data analytic platform spark. In: *Proceedings of the 12th ACM International Conference on Computing Frontiers - CF '15*. New York, New York, USA: ACM Press, 2015. (CF '15), p. 1–8. ISBN 9781450333580. Disponível em: <http://dl.acm.org/citation.cfm?doid=2742854.2747283>.

LIANG, F.; FENG, C.; LU, X.; XU, Z. Performance benefits of datampi: a case study with bigdatabench. In: SPRINGER. *Workshop on Big Data Benchmarks, Performance Optimization, and Emerging Hardware*. [S.l.], 2014. p. 111–123.

LIN, C. Y.; TSAI, C. H.; LEE, C. P.; LIN, C. J. Large-scale logistic regression and linear support vector machines using spark. *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, p. 519–528, 2015.

LIN, J.; DYER, C. Data-intensive text processing with mapreduce. *Synthesis Lectures on Human Language Technologies*, Morgan & Claypool Publishers, v. 3, n. 1, p. 1–177, 2010.

LIN, W.-h.; LEI, Z.-m.; LIU, J.; YANG, J.; LIU, F.; HE, G.; WANG, Q. MapReduce optimization algorithm based on machine learning in heterogeneous cloud environment. *The Journal of China Universities of Posts and Telecommunications*, v. 20, n. 6, p. 77–121, dec 2013. ISSN 10058885. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1005888513601120>.

LIU, D. C.; NOCEDAL, J. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, Springer, v. 45, n. 1-3, p. 503–528, 1989.

LIU, Y.; YANG, J.; HUANG, Y.; XU, L.; LI, S.; QI, M. Mapreduce based parallel neural networks in enabling large scale machine learning. *Computational intelligence and neuroscience*, Hindawi Publishing Corp., v. 2015, p. 1, 2015.

LJUNG, L. Black-box models from input-output measurements. In: IEEE. *Imtc 2001. proceedings of the 18th ieee instrumentation and measurement technology conference. rediscovering measurement in the age of informatics (cat. no. 01ch 37188)*. [S.l.], 2001. v. 1, p. 138–146.

LUVIZAN, S.; MEIRELLES, F.; DINIZ, E. H. Big data: publication evolution and research opportunities. In: *Anais da 11a Conferência Internacional sobre Sistemas de Informação e Gestão de Tecnologia. São Paulo, SP*. [S.l.: s.n.], 2014.

MA, C.; ZHANG, H. H.; WANG, X. Machine learning for big data analytics in plants. *Trends in plant science*, Elsevier, v. 19, n. 12, p. 798–808, 2014.

MAILLO, J.; RAMÍREZ, S.; TRIGUERO, I.; HERRERA, F. knn-is: An iterative spark-based design of the k-nearest neighbors classifier for big data. *Knowledge-Based Systems*, Elsevier, v. 117, p. 3–15, 2017.

MAITREY, S.; JHA, C. K. MapReduce: Simplified Data Analysis of Big Data. *Procedia Computer Science*, v. 57, p. 563–571, 2015. ISSN 1877-0509. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050915019213>.

MALEWICZ, G.; AUSTERN, M. H.; BIK, A. J.; DEHNERT, J. C.; HORN, I.; LEISER, N.; CZAJKOWSKI, G. Pregel: a system for large-scale graph processing. In: *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. [S.l.: s.n.], 2010. p. 135–146.

MANN, K. S.; KAUR, N. Cloud-deployable health data mining using secured framework for clinical decision support system. In: IEEE. *Computing and Communication (IEMCON), 2015 International Conference and Workshop on*. [S.l.], 2015. p. 1–6.

MARSLAND, S. *Machine learning: an algorithmic perspective*. [S.l.]: Chapman and Hall/CRC, 2014.

MCCALLUM, A.; NIGAM, K. et al. A comparison of event models for naive bayes text classification. In: CITESEER. *AAAI-98 workshop on learning for text categorization*. [S.l.], 1998. v. 752, n. 1, p. 41–48.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943.

MELL, P.; GRANCE, T. et al. The nist definition of cloud computing. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology Gaithersburg, 2011.

MENG, X.; BRADLEY, J.; YAVUZ, B.; SPARKS, E.; VENKATARAMAN, S.; LIU, D.; FREEMAN, J.; TSAI, D.; AMDE, M.; OWEN, S. et al. Mllib: Machine learning in apache spark. *The Journal of Machine Learning Research*, JMLR. org, v. 17, n. 1, p. 1235–1241, 2016.

MILLER, H. Big-data in cloud computing: A taxonomy of risks. Professor TD Wilson, 2013. Information Research, 18(1) paper 571. [Available at http://InformationR.net/ir/18-1/paper571.html].

MILLER, J. N. Tutorial review—outliers in experimental data and their treatment. *Analyst*, The Royal Society of Chemistry, v. 118, n. 5, p. 455–461, 1993.

MING, Z.; LUO, C.; GAO, W.; HAN, R.; YANG, Q.; WANG, L.; ZHAN, J. BDGS: A scalable big data generator suite in big data benchmarking. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 8585, p. 138–154, 2014. ISSN 16113349.

MISHRA, V.; THAKUR, S.; PATIL, A.; SHUKLA, A. Quality by design (qbd) approaches in current pharmaceutical set-up. *Expert opinion on drug delivery*, Taylor & Francis, v. 15, n. 8, p. 737–758, 2018.

MITCHELL, T. *Machine Learning*. [S.l.]: McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673.

MITCHELL, T.; CARBONELL, J.; MICHALSKI, R. *Machine Learning: A Guide to Current Research*. [S.l.]: Springer US, 2012. (The Springer International Series in Engineering and Computer Science). ISBN 9781461322795.

MOESSNER, R. *Outlier correction*. [S.l.]: Google Patents, 2007. US Patent 7,239,984.

MOHAPATRA, S. K.; MOHANTY, M. N. Big data analysis and classification of biomedical signal using random forest algorithm. In: *New Paradigm in Decision Science and Management*. [S.l.]: Springer, 2020. p. 217–224.

MONTGOMERY, D. Design and analysis of experiments (6th edn.,) john wiley and sons. *New York, NY*, 2005.

MONTGOMERY, D. *Design and Analysis of Experiments*. [S.l.]: John Wiley & Sons, Incorporated, 2017. ISBN 9781119113478.

MONTGOMERY, D. C.; PECK, E. A.; VINING, G. G. *Introduction to linear regression analysis*. [S.l.]: John Wiley & Sons, 2012. v. 821.

MOORE, R.; DENERO, J. L1 and l2 regularization for multiclass hinge loss models. In: *Symposium on machine learning in speech and language processing*. [S.l.: s.n.], 2011.

MORALES, G. D. F. Samoa: A platform for mining big data streams. In: *Proceedings of the 22nd International Conference on World Wide Web*. [S.l.: s.n.], 2013. p. 777–778.

MUNIR, R. F.; ABELLÓ, A.; ROMERO et al. Automatically configuring parallelism for hybrid layouts. In: SPRINGER. *European Conference on Advances in Databases and Information Systems*. [S.l.], 2019. p. 120–125.

NAMITHA, K.; JAYAPRIYA, A.; KUMAR, G. S. Rainfall prediction using artificial neural network on map-reduce framework. In: ACM. *Proceedings of the Third International Symposium on Women in Computing and Informatics*. [S.l.], 2015. p. 492–495.

NELDER, J. A.; MATHER, K. The analysis of randomized experiments with orthogonal block structure. ii. treatment structure and the general analysis of variance. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, v. 283, n. 1393, p. 163–178, 1965. Disponível em: <https://royalsocietypublishing.org/doi/abs/10.1098/rspa.1965.0013>.

NGUYEN, N.; KHAN, M. M. H.; ALBAYRAM, Y.; WANG, K. Understanding the Influence of Configuration Settings: An Execution Model-Driven Framework for Apache Spark Platform. In: *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*. IEEE, 2017. v. 2017-June, p. 802–807. ISBN 978-1-5386-1993-3. ISSN 21596190. Disponível em: <http://ieeexplore.ieee.org/document/8030677/>.

NIST. Nist big data interoperability framework: Definitions. *NIST Special Publication 1500-1*, Vol. 1, p. 4–5, apr 2015. Disponível em: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-1.pdf.

ODERSKY, M.; SPOON, L.; VENNERS, B. *Programming in scala*. [S.l.]: Artima Inc, 2008.

OSTLE, B. et al. Statistics in research. *Statistics in research.*, The Iowa State University Press, Ames., n. 2nd Ed, 1963.

OUSTERHOUT, K.; RASTI, R.; RATNASAMY, S.; SHENKER, S.; CHUN, B.-G. Making sense of performance in data analytics frameworks. In: *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*. [S.l.: s.n.], 2015. p. 293–307.

PACKIANATHER, M.; DRAKE, P.; ROWLANDS, H. Optimizing the parameters of multilayered feedforward neural networks through taguchi design of experiments. *Quality and reliability engineering international*, Wiley Online Library, v. 16, n. 6, p. 461–473, 2000.

PAIS, M. S.; PERETTA, I. S.; YAMANAKA, K.; PINTO, E. R. Factorial design analysis applied to the performance of parallel evolutionary algorithms. *Journal of the Brazilian Computer Society*, Springer, v. 20, n. 1, p. 6, 2014.

PARAGI, B. Divide et impera? foreign aid interventions in the middle east and north africa region. *Journal of Intervention and Statebuilding*, Taylor & Francis, v. 10, n. 2, p. 200–221, 2016.

PARK, H. W.; LEYDESDORFF, L. Decomposing social and semantic networks in emerging "big data" research. *Journal of Informetrics*, Elsevier, v. 7, n. 3, p. 756–765, 2013.

PASCAL, B.; EVELYN, J. *Les Provinciales, Or, The Mystery of Jesuitisme: Discovered in Certain Letters, Written Upon Occasion of the Present Differences at Sorbonne, Between the Jansenists and the Molinists : Displaying the Pernicious Maximes of the Late Casuists:*. Clavel, 1658. Disponível em: <https://books.google.com.br/books?id=hEVPAQAAIAAJ>.

PEK, J.; WONG, O.; WONG, C. M. Data Transformations for Inference with Linear Regression: Clarifications and Recommendations. *Practical Assessment, Research and Evaluation*, v. 22, n. 9, p. 1–11, 2017. ISSN 15317714.

PIATETSKY-SHAPIRO, G. Knowledge discovery in real databases: A report on the ijcai-89 workshop. *AI Magazine*, American Association for Artificial Intelligence, v. 11, n. 5, p. 68–70, 1991.

PILHO, K. Transaction processing performance council (tpc). *Guide d'installation*, 2014.

POGGI, N.; BERRAL, J. L.; CARRERA, D.; CALL, A.; GAGLIARDI, F.; REINAUER, R.; VUJIC, N.; GREEN, D.; BLAKELEY, J. From performance profiling to predictive analytics while evaluating hadoop cost-efficiency in aloja. In: *2015 IEEE International Conference on Big Data*. IEEE, 2015. p. 1220–1229. Disponível em: <http://ieeexplore.ieee.org/document/7363876/>.

POLITIS, S. N.; COLOMBO, P.; COLOMBO, G.; REKKAS, D. M. Design of experiments (doe) in pharmaceutical development. *Drug development and industrial pharmacy*, Taylor & Francis, v. 43, n. 6, p. 889–901, 2017.

POPESCU, A. D.; ERCEGOVAC, V.; BALMIN, A.; BRANCO, M.; AILAMAKI, A. Same Queries, Different Data: Can We Predict Runtime Performance? In: *2012 IEEE 28th International Conference on Data Engineering Workshops*. IEEE, 2012. p. 275–280. ISBN 978-0-7695-4748-0. Disponível em: <http://ieeexplore.ieee.org/document/6313693/>.

POSPELOVA, M. *Real Time Autotuning for MapReduce on Hadoop/YARN*. Tese (Doutorado) — Carleton University Ottawa, 2015.

RATHEE, S.; KAUL, M.; KASHYAP, A. R-apriori: An efficient apriori based algorithm on spark. In: *Proceedings of the 8th Workshop on Ph.D. Workshop in Information and Knowledge Management*. ACM, 2015. (PIKM '15), p. 27–34. ISBN 978-1-4503-3782-3. Disponível em: <http://doi.acm.org/10.1145/2809890.2809893>.

RATHOD, M.; SUTHAR, D.; PATEL, H.; SHELAT, P.; PAREJIYA, P. Microemulsion based nasal spray: A systemic approach for non-cns drug, its optimization, characterization and statistical modelling using qbd principles. *Journal of Drug Delivery Science and Technology*, Elsevier, v. 49, p. 286–300, 2019.

REDDY, G. T.; REDDY, M. P. K.; LAKSHMANNA, K.; KALURI, R.; RAJPUT, D. S.; SRIVASTAVA, G.; BAKER, T. Analysis of dimensionality reduction techniques on big data. *IEEE Access*, IEEE, v. 8, p. 54776–54788, 2020.

REILLY, T. O.; WINGE, S.; SCHNEIDER, S. Big Data: Release 2.0 Issue 2.0.11. *O'Reilly Media, Inc.*, n. 2, p. 44, 2009.

RIDGE, E. *Design of experiments for the tuning of optimisation algorithms*. [S.l.]: Citeseer, 2007.

RIDGE, E.; CURRY, E. A roadmap of nature-inspired systems research and development. *Multiagent and Grid Systems*, IOS Press, v. 3, n. 1, p. 3–8, 2007.

RODRIGUES, J.; VASCONCELOS, G.; MACIEL, P. *PT7 Web, an Annotated Portuguese Language Corpus*. IEEE Dataport, 2020. Disponível em: <http://dx.doi.org/10.21227/fhrm-n966>.

ROMSAIYUD, W.; PREMCHAISWADI, W. An adaptive machine learning on Map-Reduce framework for improving performance of large-scale data analysis on EC2. In: IEEE. *ICT and Knowledge Engineering (ICT KE), 2013 11th International Conference on*. [S.l.], 2013. p. 1–7. ISBN 978-1-4799-2294-9. ISSN 2157-0981.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958.

ROUSSEAU, R. et al. A view on "big data" and its relation to informetrics. 2012.

RUAN, J.; ZHENG, Q.; DONG, B. Optimal Resource Provisioning Approach based on Cost Modeling for Spark Applications in Public Clouds. In: ACM. *Proceedings of the Doctoral Symposium of the 16th International Middleware Conference on - Middleware Doct*

Symposium '15. New York, New York, USA: ACM Press, 2015. p. 1–4. ISBN 9781450337281. Disponível em: <http://dl.acm.org/citation.cfm?doid=2843966.2843972>.

SALZA, P.; FERRUCCI, F.; SARRO, F. elephant56: Design and implementation of a parallel genetic algorithms framework on hadoop mapreduce. In: ACM. *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*. [S.l.], 2016. p. 1315–1322.

SELTMAN, H. J. *Experimental Design and Analysis*. 2018. College of Humanities and Social Sciences at Carnegie Mellon University. Disponível em: <http://www.stat.cmu.edu/~hseltman/309/Book/Book.pdf>. Acesso em: 13 jan. 2020.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). *Biometrika*, JSTOR, v. 52, n. 3/4, p. 591–611, 1965.

SHI, J.; QIU, Y.; MINHAS, U. F.; JIAO, L.; WANG, C.; REINWALD, B.; ÖZCAN, F. Clash of the titans: Mapreduce vs. spark for large scale data analytics. *Proceedings of the VLDB Endowment*, VLDB Endowment, v. 8, n. 13, p. 2110–2121, 2015.

SHIRKHORSHIDI, A. S.; AGHABOZORGI, S.; WAH, T. Y.; HERAWAN, T. Big data clustering: a review. In: SPRINGER. *International Conference on Computational Science and Its Applications*. [S.l.], 2014. p. 707–720.

SIEGMUND, N.; GREBHAHN, A.; APEL, S.; KÄSTNER, C. Performance-influence models for highly configurable systems. *2015 10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ESEC/FSE 2015 - Proceedings*, p. 284–294, 2015.

SIMON, P. F. *When Numbers Get Serious*. 1983. Hearts and Bones, Warner Bros. Recording Studios.

SIMONET, A.; FEDAK, G.; RIPEANU, M. Active data: A programming model to manage data life cycle across heterogeneous systems and infrastructures. *Future Generation Computer Systems*, Elsevier, v. 53, p. 25–42, 2015.

SMITH, J.; SAINT-AMAND, H.; PLAMADĂ, M.; KOEHN, P.; CALLISON-BURCH, C.; LOPEZ, A. Dirt cheap web-scale parallel text from the common crawl. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. [S.l.: s.n.], 2013. p. 1374–1383.

STAELIN, C. Parameter selection for support vector machines. *Hewlett-Packard Company, Tech. Rep. HPL-2002-354R1*, v. 1, 2003.

STITSON, M.; WESTON, J.; GAMMERMAN, A.; VOVK, V.; VAPNIK, V. Theory of support vector machines. *University of London*, v. 117, n. 827, p. 188–191, 1996.

SUÁREZ, P. J. O.; SAGOT, B.; ROMARY, L. Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. *Challenges in the Management of Large Corpora (CMLC-7) 2019*, p. 9, 2019.

TAGUCHI, G.; CLAUSING, D. et al. Robust quality. *Harvard business review*, v. 68, n. 1, p. 65–75, 1990.

TEKINDAL, M. A.; BAYRAK, H.; ÖZKAYA, B.; YAVUZ, Y. Second-order response surface method: factorial experiments an alternative method in the field of agronomy. *Turkish Journal Of Field Crops*, v. 19, n. 1, p. 35–45, 2014.

THOMAS, L.; MANJAPPA, K.; ANNAPPA, B.; REDDY, G. R. M. Parallelized k-means clustering algorithm for self aware mobile ad-hoc networks. In: *Proceedings of the 2011 International Conference on Communication, Computing & Security*. New York, NY, USA: ACM, 2011. (ICCCS '11), p. 152–155. ISBN 978-1-4503-0464-1. Disponível em: <http://doi.acm.org/10.1145/1947940.1947973>.

TKÁČ, M.; VERNER, R. Artificial neural networks in business: Two decades of research. *Applied Soft Computing*, Elsevier, v. 38, p. 788–804, 2016.

TROYAN, S. D. *Medieval rhetoric: a casebook*. [S.l.]: Routledge, 2004.

TSAI, C.-W.; LAI, C.-F.; CHAO, H.-C.; VASILAKOS, A. V. Big data analytics: a survey. *Journal of Big Data*, v. 2, n. 1, p. 21, dec 2015. ISSN 2196-1115. Disponível em: <http://www.journalofbigdata.com/content/2/1/21>.

VAPNIK, V. *Estimation of Dependences Based on Empirical Data Berlin*. [S.l.]: Springer-Verlag, 1982.

VELLEMAN, P. F.; HOAGLIN, D. C. *Applications, basics, and computing of exploratory data analysis*. [S.l.]: Duxbury Press, 1981.

VENKATARAMAN, S.; YANG, Z.; FRANKLIN, M.; RECHT, B.; NSDI, I. Ernest : Efficient Performance Prediction for Large-Scale Advanced Analytics. *NSDI'16 Proceedings of the 13th USENIX conference on Networked Systems Design and Implementation*, p. 363–378, 2016.

VERHULST, P. *Notice sur la loi que la population suit dans son accroissement. Correspondance mathématique et physique 10: 113–121*. 1838.

VILLE, B. de. Decision trees. *Wiley Interdisciplinary Reviews: Computational Statistics*, v. 5, n. 6, p. 448–455, 2013. ISSN 19395108.

WALD, A. Statistical decision functions. Wiley, 1950.

WANG, G.; XU, J.; HE, B. A novel method for tuning configuration parameters of spark based on machine learning. In: *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2016. p. 586–593. ISBN 978-1-5090-4297-5. Disponível em: <http://ieeexplore.ieee.org/document/7828429/>.

WANG, K.; KHAN, M. M. H. Performance prediction for apache spark platform. In: IEEE. *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*. [S.l.], 2015. p. 166–173.

WEIDT, F.; SILVA, R. Revisão sistemática da literatura em ciência da computação-um guia prático. *Relatórios Técnicos do DCC/UFJF*, v. 1, 2016.

WENZEK, G.; LACHAUX, M.-A.; CONNEAU, A.; CHAUDHARY, V.; GUZMAN, F.; JOULIN, A.; GRAVE, E. Ccnet: Extracting high quality monolingual datasets from web crawl data. *arXiv preprint arXiv:1911.00359*, 2019.

WINEBERG, M.; CHRISTENSEN, S. An introduction to statistics for ec experimental analysis. In: *Tutorial at the ieee congress on evolutionary computation*. [S.l.: s.n.], 2004.

WITHANAWASAM, J. *Apache Mahout Essentials*. [S.l.]: Packt Publishing Ltd, 2015.

WU, W.; LEE, B. C. Inferred models for dynamic and sparse hardware-software spaces. In: IEEE COMPUTER SOCIETY. *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*. [S.l.], 2012. p. 413–424.

XUN, Y.; ZHANG, J.; QIN, X.; ZHAO, X. Fidoop-dp: Data partitioning in frequent itemset mining on hadoop clusters. *IEEE Transactions on Parallel and Distributed Systems*, IEEE, v. 28, n. 1, p. 101–114, 2017.

YANG, L.; ZHANG, J.; ZHANG, Q. Pesc: A parallel system for clustering ecg streams based on mapreduce. In: IEEE. *Global Communications Conference (GLOBECOM), 2013 IEEE*. [S.l.], 2013. p. 2604–2609.

YANG, X.-S.; Suash Deb. Cuckoo Search via Lévy flights. In: *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*. IEEE, 2009. p. 210–214. ISBN 978-1-4244-5053-4. Disponível em: <http://ieeexplore.ieee.org/document/5393690/>.

YATES, F. A new method of arranging variety trials involving a large number of varieties. *The Journal of Agricultural Science*, Cambridge University Press, v. 26, n. 3, p. 424–455, 1936.

ZAHARIA, M.; CHOWDHURY, M.; DAS, T.; DAVE, A. *Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing*. [S.l.], 2012. 2–2 p. Disponível em: <https://www.usenix.org/system/files/conference/nsdi12/nsdi12-final138.pdf>.

ZAHARIA, M.; KONWINSKI, A.; JOSEPH, A. D.; KATZ, R.; STOICA, I. Improving MapReduce Performance in Heterogeneous Environments. In: *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation*. Berkeley, CA, USA: USENIX Association, 2008. (OSDI'08), p. 29–42. Disponível em: <http://dl.acm.org/citation.cfm?id=1855741.1855744>.

ZHANG, G. P. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, IEEE, v. 30, n. 4, p. 451–462, 2000.

ZHANG, R.; LI, M.; HILDEBRAND, D. Finding the big data sweet spot: Towards automatically recommending configurations for hadoop clusters on docker containers. In: *Proceedings - 2015 IEEE International Conference on Cloud Engineering, IC2E 2015*. IEEE, 2015. p. 365–368. ISBN 9781479982189. Disponível em: <http://ieeexplore.ieee.org/document/7092945/>.

ZHANG, Z.; LI, W.; JIA, H. A fast face recognition algorithm based on mapreduce. In: IEEE. *Computational Intelligence and Design (ISCID), 2014 Seventh International Symposium on*. [S.l.], 2014. v. 2, p. 395–399.

ZHOU, L.; ZHONG, Z.; CHANG, J.; LI, J.; HUANG, J. Z.; FENG, S. Balanced parallel fp-growth with mapreduce. In: IEEE. *Information Computing and Telecommunications (YC-ICT), 2010 IEEE Youth Conference on*. [S.l.], 2010. p. 243–246.

ZHU, X.; WANG, B. Community mining in complex network based on parallel genetic algorithm. In: IEEE. *Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on*. [S.l.], 2010. p. 325–328.

APÊNDICE A – *CORPUS* DA LÍNGUA PORTUGUESA

A.1 COMMON CRAWL

Consiste de um repositório [1] com Petabytes (PB) de dados textuais em 160 línguas coletados ao longo de oito anos de *web crawling*. Os dados são disponibilizados para acesso público na infraestrutura *Amazon Web Services — AWS* [2]. De acordo com Wenzek et al. (2019), o *Corpus* é uma base de dados em múltiplas linguagens com misturas de extratos temporais de páginas *Web* liberados mensalmente, obtidas através da exploração randômica de amostras de URL's, e disponibilizadas em diferentes formatos: texto original, processado e metadados (SMITH et al., 2013).

Diferente das iniciativas tradicionais de arquivamento da Internet, o *Common Crawl* não prioriza o armazenamento de imagens, áudios, vídeos ou *scripts*, tais como: CSS ou *JavaScript*. Os dados armazenados se concentram prioritariamente no conteúdo textual de arquivos *HTML*. Portanto, o objetivo não é preservar a aparência exata das páginas *Web* em um determinado instante de tempo, mas sim coletar, concentrar e disponibilizar uma vasta seção transversal de texto *Web* em um único local para permitir a mineração de dados em grande escala.

Neste ínterim, o repositório tem sido utilizado em diversas iniciativas de processamento de linguagem, tais como os trabalhos de Smith et al. (2013), Buck, Heafield e Ooyen (2014), Grave et al. (2018), Baevski et al. (2019) e Suárez, Sagot e Romary (2019). Uma lista completa de iniciativas, projetos, artigos e tutoriais pode ser obtida no site do projeto [3].

Devido às características de acesso público e gratuito, escala em Petabytes e conteúdo em diferentes línguas, o *Common Crawl* foi escolhido como fonte para mineração de dados e criação do *PT7 Web Corpus* (RODRIGUES; VASCONCELOS; MACIEL, 2020) utilizado nos projetos experimentais descritos na Seção 8 - Projeto de Experimentos $2^{k-p}r$ Fatorial Fracionado.

A.1.1 Estrutura de Arquivos

Segundo Gomes, Miranda e Costa (2011), o uso de formatos padrão para arquivamento na *Web* facilita a criação colaborativa de ferramentas, como mecanismos de pesquisa ou mecanismos de replicação, para processar os dados arquivados.

1 <https://commoncrawl.org>
2 <http://aws.amazon.com>
3 <https://commoncrawl.org/the-data/examples/>

Nesse contexto, o *Common Crawl* armazena dados em formato *WARC* (ISO, 2009), publicado pela *Internet Organization for Standardization* em 2009 como o formato oficial para arquivamento de conteúdo da *Web*. Basicamente, são disponibilizados três tipos de arquivos:

- *WARC* - armazenam os dados originais brutos;
- *WAT* - armazenam metadados computados a partir dos arquivos *WARC*;
- *WET* - armazenam texto simples extraído dos arquivos *WARC*.

Desde o ano 2013, o projeto *Common Crawl* libera mensalmente grandes quantidades de dados relativos aos formatos *WARC*, *WAT* e *WET*, no formato [*s3://commoncrawl/* | *https://commoncrawl.s3.amazonaws.com/*] *CC-MAIN-YYYY-DD/* [*segment* | *warc* | *wat* | *wet*].*paths.gz*. Para efeitos de compreensão, o Quadro 16 detalha os endereços, quantidade de arquivos e tamanho total dos registros coletados no mês de Fevereiro do ano 2020 (*CC-MAIN-2020-16*):

Quadro 16 – Detalhamento do extrato *CC-MAIN-2020-16* da *Common Crawl*

Objeto	Caminho	Arquivos	Tam. (TB)
Segments	CC-MAIN-2020-16/segment.paths.gz	100	–
WARC's	CC-MAIN-2020-16/warc.paths.gz	56000	62.67
WAT's	CC-MAIN-2020-16/wat.paths.gz	56000	20.37
WET's	CC-MAIN-2020-16/wet.paths.gz	56000	8.97
Robots.txt	CC-MAIN-2020-16/robotstxt.paths.gz	56000	0.19
Non-200 status	CC-MAIN-2020-16/non200responses.paths.gz	56000	1.39
URL index	CC-MAIN-2020-16/cc-index.paths.gz	302	0.21

Fonte: <https://commoncrawl.s3.amazonaws.com/crawl-data/CC-MAIN-2020-16/index.html>

A.1.1.1 Formato WARC

Corresponde aos dados brutos coletados. Armazena a resposta *HTTP* dos sites que contata *(WARC-Type:response)* e também informações sobre como foram solicitados *(WARC-Type:request)*, além de metadados *(WARC-Type:metadados)* no próprio processo de rastreamento. No tocante à solicitação *HTTP*, a resposta bruta é armazenada junto com as informações de cabeçalho. Um típico exemplo pode ser conferido no arquivo *WARC* referente

ao endereço <https://www3.cin.ufpe.br/br/pos-graduacao/programa-academico/doutorado/sobre>, coletado a partir do extrato CC-MAIN-2020-10 [4], exibido a seguir:

```
WARC/1.0
WARC-Type: request
WARC-Date: 2020-02-28T14:25:25Z
WARC-Record-ID: <urn:uuid:51fbc611-3d43-4614-bba8-471b19f729fc>
Content-Length: 353
Content-Type: application/http; msgtype=request
WARC-Warcinfo-ID: <urn:uuid:29e06aba-f35b-4de5-b491-4d5225381a58>
WARC-IP-Address: 150.161.2.3
WARC-Target-URI: https://www3.cin.ufpe.br/br/pos-graduacao/programa-academico/doutorado
    /sobre

GET /br/pos-graduacao/programa-academico/doutorado/sobre HTTP/1.1
User-Agent: CCBot/2.0 (https://commoncrawl.org/faq/)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
If-Modified-Since: Tue, 28 Jan 2020 22:27:59 UTC
Accept-Encoding: br,gzip
Host: www3.cin.ufpe.br
Connection: Keep-Alive



WARC/1.0
WARC-Type: response
WARC-Date: 2020-02-28T14:25:25Z
WARC-Record-ID: <urn:uuid:b8c41878-7a9e-4c9f-a229-625ce38b2063>
Content-Length: 38798
Content-Type: application/http; msgtype=response
WARC-Warcinfo-ID: <urn:uuid:29e06aba-f35b-4de5-b491-4d5225381a58>
WARC-Concurrent-To: <urn:uuid:51fbc611-3d43-4614-bba8-471b19f729fc>
WARC-IP-Address: 150.161.2.3
WARC-Target-URI: https://www3.cin.ufpe.br/br/pos-graduacao/programa-academico/doutorado
    /sobre
WARC-Payload-Digest: sha1:ISWXP5OFO6GAQLEEI4WHCIEADRYQPOFQ
WARC-Block-Digest: sha1:G3KEGWH5C6BK6ZOWF7TJO3HDMPZJSZIT
WARC-Identified-Payload-Type: text/html

HTTP/1.1 200 OK
Date: Fri, 28 Feb 2020 14:25:24 GMT
Server: Apache/2.4.18 (Ubuntu)
X-Powered-By: PHP/7.0.33-0ubuntu0.16.04.12
Vary: Accept-Encoding
X-Crawler-Content-Encoding: gzip
X-Crawler-Content-Length: 5329
Content-Length: 38458
```

[4] Fevereiro de 2020

```
    Keep-Alive: timeout=5, max=100
45  Connection: Keep-Alive
    Content-Type: text/html; charset=UTF-8
47
    <!DOCTYPE html>
49  <html class="no-js" lang="pt-br">
    <head>
51  <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
53  <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Doutorado - Sobre o Doutorado | CIn - Centro de InformÃitica da UFPE</title>
55  <base href="https://www3.cin.ufpe.br/br/content/" />
    <meta name="description" content="Doutorado - Sobre o Doutorado | CIn - Centro de
        InformÃitica da UFPE">
57  <meta name="author" content="CIn - Centro de InformÃitica da UFPE">

59  ...
    ...
61  ...
```

No exemplo é possível verificar informações sobre a data de coleta em 28/02/2020 às 14:25:25, o endereço IP 150.161.2.3, servidor Apache hospedeiro Apache/2.4.18 em sistema operacional Ubuntu, bem como o código de resposta *HTTP 200* indicando sucesso da solicitação de coleta, dentre várias outras informações de cabeçalho de pedido e resposta. Por fim, o registro WARC detalha o código HTML original do objeto *Web* solicitado, propositalmente truncado por limitações de espaço.

A.1.1.2 Formato WAT

Contêm metadados importantes sobre os registros armazenados no formato *WARC*. A seguir, o registro *WAT* para o endereço canônico do exemplo da Seção A.1.1.1, acima.

```
1 {
    "Container": {
3       "Filename": "CC-MAIN-20200228135132-20200228165132-00547.warc.gz",
        "Compressed": true,
5       "Offset": "946121593",
        "Gzip-Metadata": {
7           "Deflate-Length": "485",
            "Header-Length": "10",
9           "Footer-Length": "8",
            "Inflated-CRC": "1441872284",
11          "Inflated-Length": "749"
        }
```

```
    },
    "Envelope": {
        "Payload-Metadata": {
            "Actual-Content-Type": "application/http; msgtype=request",
            "HTTP-Request-Metadata": {
                "Request-Message": {
                    "Method": "GET",
                    "Path": "/br/pos-graduacao/programa-academico/doutorado/sobre",
                    "Version": "HTTP/1.1"
                },
                "Headers-Length": "351",
                "Headers": {
                    "User-Agent": "CCBot/2.0 (https://commoncrawl.org/faq/)",
                    "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q
                        =0.8",
                    "Accept-Language": "en-US,en;q=0.5",
                    "If-Modified-Since": "Tue, 28 Jan 2020 22:27:59 UTC",
                    "Accept-Encoding": "br,gzip",
                    "Host": "www3.cin.ufpe.br",
                    "Connection": "Keep-Alive"
                },
                "Entity-Length": "0",
                "Entity-Digest": "sha1:3I42H3S6NNFQ2MSVX7XZKYAYSCX5QBYJ",
                "Entity-Trailing-Slop-Length": "0"
            },
            "Actual-Content-Length": "353",
            "Block-Digest": "sha1:Y2DQRYFKBS3KQQ22ELMMMUXGDC4FOT55",
            "Trailing-Slop-Length": "4"
        },
        "Format": "WARC",
        "WARC-Header-Length": "392",
        "WARC-Header-Metadata": {
            "WARC-Type": "request",
            "WARC-Date": "2020-02-28T14:25:25Z",
            "WARC-Record-ID": "<urn:uuid:51fbc611-3d43-4614-bba8-471b19f729fc>",
            "Content-Length": "353",
            "Content-Type": "application/http; msgtype=request",
            "WARC-Warcinfo-ID": "<urn:uuid:29e06aba-f35b-4de5-b491-4d5225381a58>",
            "WARC-IP-Address": "150.161.2.3",
            "WARC-Target-URI": "https://www3.cin.ufpe.br/br/pos-graduacao/programa-
                academico/doutorado/sobre"
        }
    }
}
```

A.1.1.3 Formato WET

Dado que muitas tarefas exigem apenas informações textuais puros, o formato *WET* contêm apenas texto simples extraído. Os dados são armazenados de maneira bastante simples e direta: metadados contendo detalhes como URL e comprimento dos dados textuais, seguidos pelo texto puro processado. A seguir, o registro *WET* referente ao endereço fornecido no exemplo da Seção A.1.1.1, acima.

WARC/1.0
WARC-Type: conversion
WARC-Target-URI: https://www3.cin.ufpe.br/br/pos-graduacao/programa-academico
/doutorado/sobre
WARC-Date: 2020-02-28T14:25:25Z
WARC-Record-ID: <urn:uuid:c04d6833-82df-438a-af43-fd86fe0bfe8b>
WARC-Refers-To: <urn:uuid:b8c41878-7a9e-4c9f-a229-625ce38b2063>
WARC-Block-Digest: sha1:VRCAFCXQRD7SZLZLCEBO5Y7P4P4OCQZV
Content-Type: text/plain
Content-Length: 3102

Doutorado - Sobre o Doutorado | CIn - Centro de Informática da UFPE
Portal do Governo Brasileiro

...
...
...

O curso de Doutorado em Ciência da Computação do Centro de Informática (CIn) da UFPE começou a funcionar em março de 1992. O objetivo do doutorado é desenvolver e aprimorar a forma de fazer pesquisa e gerar novos conhecimentos a partir do estímulo ao aprofundamento acadêmico para quem deseja ensinar.
Com duração esperada de 48 meses, é parte fundamental da carreira de pesquisador. Nesse caso, são necessárias nove disciplinas da área de pesquisa do candidato e dois trabalhos individuais.

Durante o curso, são considerados problemas originais de pesquisa e há amplas oportunidades de publicação e interação com a comunidade. Também é prática comum realizar parte do curso em outra instituição, normalmente estrangeira. O egresso do programa de doutorado normalmente vai para universidades, institutos federais e/ou centros de pesquisa. Mesmo concentrado em pesquisa, o doutorando tem ao longo do curso um foco na prática.

Fone: + 55 81 2126.8430

Fax: + 55 81 2126.8438

Contato: contato@cin.ufpe.br

A.2 MINERAÇÃO DE CORPUS EM LÍNGUA PORTUGUESA

Foi realizada dupla checagem de pertencimento do conteúdo da página à Língua Portuguesa. Primeiro, a estrutura do *Common Crawl* permite a filtragem de dados por *TLD*, o que permite a separação por *URL* de país. Em segundo lugar, cada entrada possui metadados com informação de até três linguagens detectadas no texto. Foram capturadas páginas *Web* dos países: **.ao* (Angola), **.br* (Brasil), **.pt* (Portugal), **.cv* (Cabo Verde), **.gw* (Guiné-Bissau), **.mo* (Macau) e **. mz* (Moçambique).

Outros três países adotam Português como língua oficial — **gq* (Guiana Equatorial), **.st* (São Tomé e Príncipe) e **.tl* (Timor Leste). Todavia, estes não foram incluídos na construção do *Corpus*, dado que estão disponíveis através de revendedores de todo o mundo, sem necessidade de presença no território do país. Tal fato incorre em alta mistura de línguas, como de fato foi observado, pois a maioria das páginas para estes países possuem predominância de textos em língua inglesa, dentre outras.

Foram capturados registros do Common Crawl dos sete países acima descritos no período Setembro/2018 a Março/2020, compreendendo 19 meses de páginas *Web* em língua portuguesa. As páginas mineradas do *TLD *.br* foram nomeadas como classe 1: português brasileiro, as demais como classe 0: português não brasileiro.

Para efeitos de classificação de texto, o interesse reside nos arquivos em formato *WET*. Estes foram minerados a partir máquinas hospedadas na *Amazon AWS* e armazenados em

formato *CSV*, onde cada registro (conteúdo textual simples de uma página) foi transformado para codificação *Base64* (JOSEFSSON et al., 2006) para manipular apropriadamente quebras de linha e caracteres especiais. O conjunto de dados originais coletados do *Common Crawl* totalizou 249.74 GB relativos a 16.346.693 de páginas *Web* coletadas dos sete países acima mencionados.

Após processamento para adaptação ao formato de entrada dos algoritmos de aprendizagem de máquina, o *Corpus* em Língua Portuguesa passou a ter as seguintes características:

Tabela 24 – Características dos conjuntos de treinamento e teste do *Corpus*

		Classe 0	Classe 1	Total	Tamanho
100%	**Treinamento**	6.319.173 (55.21%)	5.126.086 (44.79%)	11.445.259	25.63 GB
	Teste	2.705.011 (55.19%)	2.196.423 (44.81%)	4.901.434	10.86 GB
3.4%	**Treinamento**	315.685 (55.18%)	256.453 (44.82%)	572.138	861.9 MB
	Teste	134.901 (55.16%)	109.674 (44.84%)	244.575	421.9 MB
Total		9.024.184 (55.20%)	7.322.509 (44.80%)	16.346.693	36.49 GB

Fonte: O autor (2020)

A.3 PRÉ-PROCESSAMENTO DO *CORPUS*

Código Fonte 1 – Pré-processamento do *Corpus*

```
1  package br.ufpe.cin.bigdata.cc.etl
2
3  import org.apache.spark.ml.feature.StopWordsRemover
4  import org.apache.spark.ml.feature.{HashingTF, IDF}
5  import org.apache.spark.ml.feature.{RegexTokenizer, Tokenizer}
6  import org.apache.spark.ml.Pipeline
7  import org.apache.spark.sql.SparkSession
8  import org.apache.spark.sql.functions._
9
10 object PreProcessingCommonCrawl {
11
12     def main(args: Array[String]): Unit = {
13
14         if (args.length != 2) {
15             System.err.println(s"Usage: $PreProcessingCommonCrawl <INPUT_PATH> <OUTPUT_PATH
                   >")
16             System.exit(1)
17         }
18
19         var SEED = 101L
```

```
20          var prefix = args(0)
21          var output_path = args(1)
22          val spark = SparkSession.builder.appName("PreProcessingCommonCrawl").getOrCreate()
23
24          val data = spark.read.format("csv").
25                      option("delimiter", "\t").
26                      load(prefix + "/wet-data").
27                      withColumnRenamed("_c0","label").
28                      withColumnRenamed("_c1","url").
29                      withColumnRenamed("_c2","digest").
30                      withColumnRenamed("_c3","text64byte")
31
32          val raw = data.filter("text64byte is not null").
33                      filter("text64byte <> 'DQo='").
34                      withColumn("raw" , unbase64(col("text64byte"))).
35                      cast("string")).
36                      drop("text64byte").
37                      dropDuplicates("digest").
38                      dropDuplicates("url").cache()
39
40          val tokenizer = new RegexTokenizer().setGaps(false).
41                              setPattern("[\\p{L}\\w&&[^\\d]]+").
42                              setMinTokenLength(3).
43                              setInputCol("raw").
44                              setOutputCol("tokens")
45
46          val swRemover = new StopWordsRemover().
47                              setStopWords(StopWordsRemover.loadDefaultStopWords("portuguese"
                                    )).
48                              setCaseSensitive(false).
49                              setInputCol("tokens").
50                              setOutputCol("filtered")
51
52          val hashTF = new HashingTF().
53                          setNumFeatures(scala.math.pow(2,14).toInt).
54                          setInputCol("filtered").
55                          setOutputCol("hash-tf")
56
57          val hashIDF = new IDF().setInputCol("hash-tf").setOutputCol("hash-features")
58          val pipeline = new Pipeline().setStages(Array(tokenizer, swRemover, hashTF, hashIDF
                ))
59          val all = pipeline.fit(raw).transform(raw).cache()
60          all.write.parquet(prefix + output_path + "/pt7-corpus.parquet")
61
62          var columnNames = Seq("label","hash-features")
63          val result_hash = all.select(columnNames.head, columnNames.tail: _*).
                withColumnRenamed("hash-features", "features").cache()
64          val Array(trainh, testh) = result_hash.randomSplit(Array(0.7, 0.3), seed = SEED)
65          val Array(trainh5, trainh95) = trainh.randomSplit(Array(0.05, 0.95), seed = SEED)
```

```
66          val Array(testh5, testh95) = testh.randomSplit(Array(0.05, 0.95), seed = SEED)
67
68          trainh.write.parquet(prefix + output_path + "/train.parquet")
69          testh.write.parquet(prefix + output_path + "/test.parquet")
70          trainh5.write.parquet(prefix + output_path + "/train-small.parquet")
71          testh5.write.parquet(prefix + output_path + "/test-small.parquet")
72
73          spark.stop()
74      }
75  }
```

APÊNDICE B – IMPLEMENTAÇÃO DAS TAREFAS DE APRENDIZAGEM DE MÁQUINA

Código Fonte 2 – Regressão Logística (LR)

```
1  package br.ufpe.cin.bigdata.cc.ml
2
3  import org.apache.spark.ml.classification.{LogisticRegression}
4  import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
5  import org.apache.spark.ml.tuning.{ ParamGridBuilder, TrainValidationSplit }
6  import org.apache.spark.ml.linalg.SparseVector
7  import org.apache.spark.sql.SparkSession
8  import org.apache.spark.sql.functions._
9
10 object LogisticRegressionPT7 {
11     def main(args: Array[String]): Unit = {
12
13         if (args.length != 3) {
14             System.err.println(s"LogisticRegressionPT7 <prefix> <train_path> <test_path>")
15             System.exit(1)
16         }
17
18         val spark = SparkSession.builder.appName("LogisticRegressionPT7").getOrCreate()
19         var prefix = args(0)
20         var train_path = args(1)
21         var test_path = args(2)
22
23         val train = spark.read.load(prefix + train_path).
24                                 withColumn("label", expr("CAST(label AS INTEGER)")).
25                                 filter(line => line.getAs[SparseVector](1).indices.size >
                                       0).
26                                 cache()
27
28         val test = spark.read.load(prefix + test_path).
29                                 withColumn("label", expr("CAST(label AS INTEGER)")).
30                                 filter(line => line.getAs[SparseVector](1).indices.size >
                                       0).
31                                 cache()
32
33         val lr = new LogisticRegression().setFamily("binomial").
34                     setAggregationDepth(2).
35                     setElasticNetParam(0.0).
36                     setRegParam(0).
37                     setMaxIter(20).
38                     setTol(1.0E-6)
39
40         val paramGrid = new ParamGridBuilder().build()
41         val trainValidationSplit = new TrainValidationSplit().
```

```
                    setEstimator(lr).
                    setEvaluator(new BinaryClassificationEvaluator).
                    setEstimatorParamMaps(paramGrid).
                    setTrainRatio(0.8)

        val model = trainValidationSplit.fit(train)
        evaluate(train, model)
        evaluate(test, model)
        spark.stop()
    }
}
```

Código Fonte 3 – Máquinas de Vetores de Suporte (SVM)

```
package br.ufpe.cin.bigdata.cc.ml

import org.apache.spark.ml.classification.LinearSVC
import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
import org.apache.spark.ml.tuning.{ CrossValidator, CrossValidatorModel, ParamGridBuilder }
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._

object LSVMPT7 {

    def main(args: Array[String]): Unit = {
        if (args.length != 3) {
            System.err.println(s"LSVMPT7 <prefix> <train_path> <test_path>")
            System.exit(1)
        }

        val spark = SparkSession.builder.appName("LSVMPT7").getOrCreate()
        var prefix = args(0)
        var train_path = args(1)
        var test_path = args(2)

        val train = spark.read.load(prefix + train_path).
                        withColumn("label", expr("CAST(label AS INTEGER)")).
                        cache()

        val test = spark.read.load(prefix + test_path).
                        withColumn("label", expr("CAST(label AS INTEGER)")).
                        cache()

        val lsvm = new LinearSVC().
                    setMaxIter(10).
                    setAggregationDepth(2).
                    setRegParam(0).
                    setTol(1.0E-6)

        val model = lsvm.fit(train)
```

```
37          evaluate(train, model)
38          evaluate(test, model)
39          spark.stop()
40      }
41  }
```

Código Fonte 4 – *Naïve Bayes* (NB)

```
1   package br.ufpe.cin.bigdata.cc.ml
2
3   import org.apache.spark.ml.classification.NaiveBayes
4   import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
5   import org.apache.spark.ml.tuning.{ CrossValidator, CrossValidatorModel, ParamGridBuilder }
6   import org.apache.spark.sql.SparkSession
7   import org.apache.spark.sql.functions._
8
9   object NaiveBayesPT7 {
10
11      def main(args: Array[String]): Unit = {
12          if (args.length != 3) {
13              System.err.println(s"NaiveBayesPT7 <prefix> <train_path> <test_path>")
14              System.exit(1)
15          }
16
17          val spark = SparkSession.builder.appName("NaiveBayesPT7").getOrCreate()
18          var prefix = args(0)
19          var train_path = args(1)
20          var test_path = args(2)
21
22          val train = spark.read.load(prefix + train_path).
23              withColumn("label", expr("CAST(label AS INTEGER)")).
24              cache()
25
26          val test = spark.read.load(prefix + test_path).
27              withColumn("label", expr("CAST(label AS INTEGER)")).
28              cache()
29
30          val bayes = new NaiveBayes().setSmoothing(1.0)
31          val paramGrid = new ParamGridBuilder().build()
32
33          val model = bayes.fit(train)
34          evaluate(train, model)
35          evaluate(test, model)
36          spark.stop()
37      }
38  }
```

Código Fonte 5 – Florestas Aleatórias (RF2)

```scala
package br.ufpe.cin.bigdata.cc.ml

import org.apache.spark.ml.classification.RandomForestClassifier
import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
import org.apache.spark.ml.tuning.{ CrossValidator, CrossValidatorModel, ParamGridBuilder }
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._

object RandomForestPT7 {

    def main(args: Array[String]): Unit = {

        if (args.length != 3) {
            System.err.println(s"RandomForestPT7 <prefix> <train_path> <test_path>")
            System.exit(1)
        }

        val spark = SparkSession.builder.appName("RandomForestPT7").getOrCreate()
        var prefix = args(0)
        var train_path = args(1)
        var test_path = args(2)
        val train = spark.read.load(prefix + train_path).
                        withColumn("label", expr("CAST(label AS INTEGER)")).
                        cache()
        val test = spark.read.load(prefix + test_path).
                        withColumn("label", expr("CAST(label AS INTEGER)")).
                        cache()

        val rforest = new RandomForestClassifier().
                        setMaxDepth(4).
                        setImpurity("gini").
                        setNumTrees(20).
                        setSeed(304)

        val model = rforest.fit(train)
        evaluate(train, model)
        evaluate(test, model)
        spark.stop()
      }
}
```

Código Fonte 6 – *Perceptron* Multicamadas (MLP)

```scala
package br.ufpe.cin.bigdata.cc.ml

import org.apache.spark.ml.classification.MultilayerPerceptronClassifier
import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
import org.apache.spark.ml.tuning.{ ParamGridBuilder, TrainValidationSplit }
import org.apache.spark.ml.linalg.SparseVector
import org.apache.spark.sql.SparkSession
```

```
8  import org.apache.spark.sql.functions._
9
10 object MLPPT7 {
11
12     def main(args: Array[String]): Unit = {
13         if (args.length != 3) {
14             System.err.println(s"MLPPT7 <prefix> <train_path> <test_path>")
15             System.exit(1)
16         }
17
18         val spark = SparkSession.builder.appName("MLPPT7").getOrCreate()
19         var prefix = args(0)
20         var train_path = args(1)
21         var test_path = args(2)
22         val train = spark.read.load(prefix + train_path).
23                         withColumn("label", expr("CAST(label AS INTEGER)")).
24                         filter(line => line.getAs[SparseVector](1).indices.size > 0).cache
                               ()
25         val test = spark.read.load(prefix + test_path).
26                         withColumn("label", expr("CAST(label AS INTEGER)")).
27                         filter(line => line.getAs[SparseVector](1).indices.size > 0).cache
                               ()
28         val layers = Array[Int](16384, 30, 5, 2)
29         val mlp = new MultilayerPerceptronClassifier().
30                         setLayers(layers).
31                         setSeed(42L).
32                         setBlockSize(64).
33                         setMaxIter(10)
34         val paramGrid = new ParamGridBuilder().build()
35         val trainValidationSplit = new TrainValidationSplit().
36                         setEstimator(mlp).
37                         setEvaluator(new BinaryClassificationEvaluator).
38                         setEstimatorParamMaps(paramGrid).
39                         setTrainRatio(0.8)
40         val model = trainValidationSplit.fit(train)
41         evaluate(train, model)
42         evaluate(test, model)
43         spark.stop()
44     }
45 }
```

Código Fonte 7 – Avaliação de modelos de treinamento e teste

```
1  package br.ufpe.cin.bigdata.cc
2
3  import org.apache.spark.sql.Dataset
4  import org.apache.spark.mllib.evaluation.MulticlassMetrics
5  import org.apache.spark.mllib.evaluation.BinaryClassificationMetrics
6  import org.apache.spark.ml.Model
7  import results.sparkSession.implicits._
```

```
8  import org.apache.spark.sql.functions._
9
10 package object ml {
11
12     def accuracy(tp: Double, fp: Double, tn: Double, fn: Double): Double = {
13         return (tp + tn)/(tp + fn + fp + tn)
14     }
15
16     def precision(tp: Double, fp: Double): Double = {
17         return tp/(tp + fp)
18     }
19
20     def recall(tp: Double, fn: Double): Double = {
21         return tp/(tp + fn)
22     }
23
24     def specificity(fp: Double, tn: Double): Double = {
25         return tn/(fp + tn)
26     }
27
28     def auc(tp: Double, fp: Double, tn: Double, fn: Double): Double = {
29         return 0.5 * (  (tp/(tp + fn)) + (tn/(tn + fp)) )
30     }
31
32     def f_score(tp: Double, fp: Double, fn: Double, beta:Double = 1): Double =  {
33         val b2 = scala.math.pow(beta, 2)
34
35         return ((b2+1) * tp) / (((b2+1)*tp) + (b2*fn) + fp)
36     }
37
38     def f1_score(tp: Double, fp: Double, fn: Double): Double =  {
39         val pr = precision(tp, fp)
40         val rc = recall(tp, fn)
41         return (2 * (pr * rc) / (pr + rc) )
42     }
43
44     def evaluate(set: Dataset[_], bestModel: Model[_]) {
45         var cNames = Seq("prediction","label")
46         val results = bestModel.transform(set).cache()
47
48         val predAndLabels = results.select(cNames.head, cNames.tail: _*).as[(Double, Double
               )].rdd
49         val metrics = new MulticlassMetrics(predAndLabels)
50         val confusionMatrix = metrics.confusionMatrix
51         val m = new BinaryClassificationMetrics(predAndLabels)
52         val tn = confusionMatrix.apply(0,0)
53         val fp = confusionMatrix.apply(0,1)
54         val fn = confusionMatrix.apply(1,0)
55         val tp = confusionMatrix.apply(1,1)
```

```
56
57          println(confusionMatrix)
58          println("AUC PR: \t%.4f".format(m.areaUnderPR))
59          println("AUC ROC: \t%.4f".format(m.areaUnderROC))
60          println("Accuracy: \t%.4f".format(accuracy(tp, fp, tn, fn)))
61          println("Error: \t%.4f".format(1 - accuracy(tp, fp, tn, fn)))
62          println("Precision: \t%.4f".format(precision(tp, fp)))
63          println("Recall: \t%.4f".format(recall(tp, fn)))
64          println("Specificity: \t%.4f".format(specificity(fp, tn)))
65          println("F1-score: \t%.4f".format(f1_score(tp, fp, fn)))
66          println("F-score (beta 0.5): \t%.4f".format(f_score(tp, fp, fn, 0.5)))
67      }
68  }
```

ANEXO A – NOTAS TEÓRICAS COMPLEMENTARES

Este anexo complementa as Seções 4 - Projeto de Experimentos e 5 - Regressão Linear para o leitor não familiarizado com os fundamentos e termos básicos do processo experimental e da análise de modelos lineares.

A.1 CONCEITOS EXPERIMENTAIS

Detalhamento teórico aprofundado pode ser conferido em Jain (1990) e Lawson (2014). Na sequência, conceitos básicos acerca do processo experimental:

- **experimento** ou **execução** - ato de mudar ao menos uma variável sob estudo e observar seu(s) resultado(s);
- **unidade experimental** - entidade concreta utilizada no *experimento* sobre a qual alguma mudança é realizada;
- **variável dependente** ou **variável de resposta** - resultado de um *experimento* em relação àquilo que se deseja observar. Geralmente refere-se a uma medida de desempenho como tempo, custo ou qualquer outra métrica de interesse do problema;
- **variável independente**, **fator** ou **preditor** - elementos do estudo cujos valores podem ser sistematicamente controlados durante o *experimento* com o intuito de determinar o *efeito* sobre a *variável de resposta*;
- **variável oculta** - *fator* desconhecido ou impossível de controlar, mas que pode produzir *efeito* sobre a *variável de resposta* do *experimento*;
- **replicação** - dois ou mais *experimentos* conduzidos sob as mesmas condições (*fatores* e *níveis*) em diferentes *unidades experimentais*;
- **repetição** - coleta da *variável de resposta* para o mesmo *experimento* sob a mesma *unidade experimental*. Procedimento comum em cenários nos quais existe a possibilidade de mudanças no valor medido devido a erros de mensuração;
- **randomização** - organização aleatória da ordem de execução dos *experimentos* com o intuito de evitar ou minimizar o viés causado por *variáveis ocultas*;

- **nível** ou **tratamento** - os valores categóricos, discretos ou contínuos que um *fator* pode assumir;
- **efeito** - mudança no valor da *variável de resposta* causada por um *fator*;
- **erro experimental** - a diferença entre o valor da *variável de resposta* de um *experimento* em particular e a média de todos os *experimentos* sob as mesmas condições;
- **interação** - quando o *efeito* de um *fator* depende da variação de *nível* de outro;

Acerca da importância das replicações e randomizações, Anderson e Kraber (1999) destacam que a replicação de unidades experimentais melhora a chance de detectar um efeito estatisticamente significativo no meio da variação natural do processo; enquanto a execução randomizada das unidades experimentais evita a influência de variáveis ocultas, geralmente relacionadas com a sucessão no tempo e que podem influenciar significativamente a resposta. A prática de conduzir experimentos replicados e randomizados foi introduzida por Fisher e Wishart (1930) como técnica de controle de erros para lidar com as variações intrínsecas do ambiente de execução dos experimentos.

A.2 CONSTRUÇÃO DA MATRIZ DE YATES

Considerando-se um projeto fatorial de k fatores em dois níveis, o primeiro fator base tem seus níveis alterados a cada execução (–1, + 1, –1, + 1, ...); o segundo fator a cada duas execuções (–1, –1, + 1, + 1, –1, – 1, + 1, + 1, ...); o terceiro a cada quatro, e assim por diante. Antes de um novo fator base ser adicionado, todas as interações dos fatores base já presentes são incluídas na matriz (GRÖNMPING, 2014). Para compreensão, considere-se o exemplo do Quadro 17.

É importante destacar que os níveis das interações entre fatores conferem com a ordem de *Yates*, ou seja, os contrastes para a interação A:B são formados pela multiplicação da coluna de contrastes do fator A com a coluna de contrastes do fator B. A mesma lógica é aplicada para as interações A:C, B:C e A:B:C. Geralmente, esses arranjos permitem a análise dos efeitos da variação dos fatores e suas interações através dos modelos de regressão linear multivariada. O princípio da ortogonalidade [1] obtido através desta distribuição e as vantagens da Matriz de *Yates* podem ser conferidos em mais detalhes em Nelder e Mather (1965) e Jain (1990).

[1] Efeitos de qualquer fator somam zero nos efeitos de outros fatores. A análise de um projeto ortogonal é muitas vezes direta, porque é possível estimar cada efeito principal e interação independentemente.

Quadro 17 – Matriz de *Yates*, 2^3 fatorial completo

	A	B	AB	C	AC	BC	ABC
1	−1	−1	+1	−1	+1	+1	−1
2	+1	−1	−1	−1	−1	+1	+1
3	−1	+1	−1	−1	+1	−1	+1
4	+1	+1	+1	−1	−1	−1	−1
5	−1	−1	+1	+1	−1	−1	+1
6	+1	−1	−1	+1	+1	−1	−1
7	−1	+1	−1	+1	−1	+1	−1
8	+1	+1	+1	+1	+1	+1	+1

Fonte: O autor (2020), adaptado de Grönmping (2014)

A.3 TESTE DE HIPÓTESE

É de fundamental importância que o modelo a ser utilizado para predição de uma variável dependente ou inferência de efeitos de fatores seja estatisticamente significativo. Do contrário, a confiança nos valores previstos pelo modelo pode ser interpretada como um evento meramente aleatório. Assim, torna-se necessário comprovar a hipótese de que existe uma relação entre fatores e resultado.

Consoante Montgomery (2017), uma hipótese estatística é uma afirmação sobre os parâmetros de uma distribuição de probabilidade ou os parâmetros de um modelo. A hipótese reflete uma conjectura sobre um dado problema. No contexto apresentado nesta Tese — ou seja, projetos 2^k fatoriais; tomadas duas formulações como sendo os dois níveis de um fator, onde $y_{11}, y_{12}, ..., y_{1n}$ representam n_1 observações do nível 1 enquanto $y_{21}, y_{22}, ..., y_{2n}$ representam n_2 observações do nível 2, os resultados de um experimento descrito por um modelo estatístico pode ser dado por:

$$y_{ij} = \mu_i + \epsilon_{ij} = \begin{cases} i = 1, 2 \\ \\ j = 1, 2, ..., n \end{cases} \tag{A.1}$$

, na qual y_{ij} é a *j*-ésima observação do nível *i* para o fator hipotético, μ_i é a média das respostas do grupo referente ao *i*-ésimo nível e ϵ_{ij} é uma variável randômica normal associada com a observação y_{ij}.

A hipótese da existência de uma relação consistente entre fatores e resultado pode ser formulada sobre a conjectura da igualdade das médias para os dois grupos. Formalmente,

define-se a Hipótese Nula (H_0) para $\mu_1 = \mu_2$ e a Hipótese Alternativa (H_1) para $\mu_1 \neq \mu_2$. O teste de hipótese gira em torno de (i) rejeitar ou (ii) falhar em rejeitar H_0 a partir de um teste estatístico, no qual pode acontecer o Erro Tipo I (rejeitar H_0 quando H_0 é verdadeira) ou o Erro Tipo II (falhar em rejeitar H_0 quando H_0 é falsa). Esses dois erros possuem símbolos e nomenclaturas especiais, a saber:

$$\alpha = P(\textit{Erro Tipo I}) = P(\textit{rejeitar } H_0 | H_0 \textit{ é verdadeira})$$

$$\beta = P(\textit{Erro Tipo II}) = P(\textit{falha em rejeitar } H_0 | H_0 \textit{ é falsa})$$

O procedimento geral no teste de hipótese consiste em especificar um valor da probabilidade (do original em inglês *p-value*) de α, frequentemente chamado **nível de significância**, e então projetar o procedimento de teste de forma que a probabilidade de β tenha um valor adequadamente pequeno. Normalmente, uma maneira de informar os resultados do teste é afirmar se a hipótese nula foi ou não rejeitada com um determinado nível de significância. A abordagem *p-value* tem sido amplamente adotada na prática científica e conceitualmente representa a probabilidade de que a estatística de teste assumirá um valor que é pelo menos tão extremo quanto o valor observado da estatística quando a hipótese nula H_0 for verdadeira. Posto em outras palavras, o *p-value* é o menor nível de significância que levaria à rejeição da hipótese nula H_0 (MONTGOMERY, 2017).

Quão maior for a estatística *t*, menor será o *p-value*. Quando a hipótese nula é rejeitada, pode-se afirmar que não existe uma relação entre as variáveis dependentes (fatores) e a resposta (resultado). Quando não se pode rejeitar a hipótese nula, não se pode negar a relação entre preditores e resultado. É prática normalizada a utilização de *p-value* < 0.05 para assegurar que não se possa rejeitar a consistência dos dados e a suposição de que a hipótese nula seja verdadeira.

Explanações matemáticas aprofundadas, que incluem mais variações de teste de significância e demonstrações detalhadas podem ser conferidas pelo leitor interessado em Jain (1990), Montgomery, Peck e Vining (2012) e Montgomery (2017).

A.4 BASES MATEMÁTICAS DA REGRESSÃO LINEAR

Na regressão linear a Hipótese Nula H_0 admite que os coeficientes de regressão (β) associados com as variáveis preditoras são iguais a zero. A Hipótese Alternativa H_1 supõe que os coeficientes não equivalem a zero, ou seja: há uma relação entre a variável dependente e

as variáveis preditoras (Seção A.3). Valores de probabilidade *p-value* < 0.05 informarão nesse caso que se pode rejeitar a hipótese nula, ou seja, os coeficientes de regressão são significativos; mais ainda, não se pode negar que haja relacionamento entre as variáveis preditoras e a variável de resposta. Para valores maiores de *p-value* os coeficientes não serão significativos e podem ser desconsiderados como variáveis influentes no modelo. Na sequência, conceitos relacionados com o cálculo dos valores da regressão linear:

- y_i - valor real da variável de resposta na i-ésima observação;
- $\hat{y}$ - valor da variável de resposta ajustado pelo modelo na i-ésima observação;
- $\overline{y}$ - média da variável de resposta;
- $SSE = \sum_i^n (y_i - \hat{y})^2$ - soma dos quadrados residuais, diferença entre o valor real (y_i) e o ajustado ($\hat{y}$), e equivale à variação não explicada pelo modelo;
- $SST = \sum_i^n (y_i - \overline{y})^2$ - soma dos quadrados totais, diferença entre o valor real (y_i) e a média da variável dependente ($\overline{y}$), e equivale ao total de variação na variável dependente;
- $SSR = SST - SSE$ - soma dos quadrados de regressão, equivale ao total de variação explicada pelos coeficientes de regressão.

Isto posto, uma importante métrica de desempenho de modelos consiste no coeficiente de determinação R^2, que quantifica o total de variação sobre a variável dependente explicado pelas variáveis independentes ou fatores. Importa destacar que $R^2 \in \mathbb{R}, 0 \leq R^2 \leq 1$, onde valores próximos a 0 (zero) denotam um modelo que não explica bem a variação, enquanto valores próximos a 1 (um) sugerem modelos que se ajustam e explicam a variação da resposta com muita precisão. O coeficiente de determinação é dado por:

$$R^2 = 1 - \frac{SSE}{SST} \tag{A.2}$$

Todavia, a medida que variáveis são adicionadas ao modelo, o valor R^2 se torna maior, dado que todas as variáveis presentes no subconjunto de variáveis anteriores estarão ainda presentes no superconjunto de novas variáveis. Assim, a métrica R^2_{adj} (R^2 ajustado) é formulada de maneira a penalizar o número de variáveis preditoras:

$$R^2_{adj} = 1 - \frac{(n-1) \times (1 - R^2)}{n - q} \tag{A.3}$$

, onde

- *n* - número de observações;
- *q* - número de coeficientes de regressão [2] do modelo.

Por fim, também são medidas de qualidade do modelo o Erro Padrão (*se)* e a estatística *F*, formuladas pelas Equações A.4 e A.5, a seguir:

$$se = \sqrt{\frac{SSE}{n-q}} \tag{A.4}$$

$$F = \frac{SSR}{(q-1) \times se^2} \tag{A.5}$$

O Quadro 18 resume as métricas de avaliação de modelos utilizadas nesta Tese.

Quadro 18 – Métricas de qualidade de modelos lineares

Métrica	Descrição
$SSE = \sum_i^n (y_i - \hat{y})^2$	Soma dos quadrados residuais, quantifica a variação não explicada pelo modelo.
$SST = \sum_i^n (y_i - \overline{y})^2$	Soma dos quadrados total, quantifica a variação sobre a variável dependente.
$SSR = SST - SSE$	Soma dos quadrados de regressão, variação explicada pelos coeficientes da equação.
$R^2 = 1 - \frac{SSE}{SST}$	Coeficiente de determinação, mede o percentual da soma de quadrados total que pode ser explicado pela equação de regressão estimada. Possui valores $0 \leq R^2 \leq 1$; para $R^2 \approx 1$ tem-se o melhor ajuste, enquanto $R^2 \approx 0$ indica que o não há relacionamento entre a resposta e as variáveis preditoras.
$R^2_{adj} = 1 - \frac{(1-R^2)\times(n-1)}{n-q}$	R^2 penalizado devido à adição de fatores.
$se = \sqrt{\frac{SSE}{n-q}}$	Erro padrão ($se > 0$), quantifica a precisão da estimativa dos coeficientes de regressão. Em geral, quão menor, melhor o modelo.
p-value	Nível de significância, comumente esperado valor < 0.05

Fonte: Jain (1990), Montgomery, Peck e Vining (2012)

[2] A quantidade de fatores e de interações investigadas.