UNIVERSIDADE FEDERAL DO PAMPA

GUILHERME AFONSO HALAL

AGROBAYES – PACOTE R PARA AGRICULTURA DE PRECISÃO COM REDES BAYESIANAS

Bagé
2023

GUILHERME AFONSO HALAL

AGROBAYES – PACOTE R PARA AGRICULTURA DE PRECISÃO COM REDES BAYESIANAS

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Engenharia de Computação como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Orientadora: Ana Paula Lüdtke Ferreira

Bagé
2023

Ficha catalográfica elaborada automaticamente com os dados fornecidos pelo(a) autor(a) através do Módulo de Biblioteca do Sistema GURI (Gestão Unificada de Recursos Institucionais).

A257a Halal, Guilherme Afonso

AGROBAYES - Pacote R para agricultura de precisão com redes Bayesianas / Guilherme Afonso Halal.

168 f.: il.

Orientadora: Ana Paula Lüdtke Ferreira

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal do Pampa, Engenharia de Computação, 2023.

1 Agricultura digital2. Inferência probabilística. 3. Ciência de dados. I. Título.

SERVIÇO PÚBLICO FEDERAL

MINISTÉRIO DA EDUCAÇÃO

Universidade Federal do Pampa

GUILHERME AFONSO HALAL

AGROBAYES - PACOTE R PARA AGRICULTURA DE PRECISÃO COM REDES BAYESIANAS

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Trabalho de Conclusão de Curso defendido e aprovado em: 1 de fevereiro de 2023.

Banca examinadora:

_____

Profa. Dra. Ana Paula Lüdtke Ferreira
Orientadora

UNIPAMPA

_____

Prof. Dr. Leonardo Bidese de Pinho
UNIPAMPA

_____

Prof. Dr. Sandro da Silva Camargo

UNIPAMPA

Assinado eletronicamente por **LEONARDO BIDESE DE PINHO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 07/02/2023, às 16:19, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **ANA PAULA LUDTKE FERREIRA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 07/02/2023, às 22:05, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **SANDRO DA SILVA CAMARGO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 07/02/2023, às 23:00, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1049207** e o código CRC **2A52BDF3**.

Referência: Processo nº 23100.002573/2023-40 SEI nº 1049207

Dedico este trabalho a minha filha Giovana, desejando que a curiosidade dela leve-a sempre a grandes descobertas e que o meu exemplo conduza ela sempre pelo caminho do bem.

AGRADECIMENTO

Agradeço, primeiramente ao Criador por fazer a vida o universo e tudo mais, espalhando por toda parte os mistérios que movem a curiosidade humana a descobrir, inventar e inovar. Aos meus pais, por sempre me incentivarem a estudar, as minhas irmãs que sempre foram meus exemplos, em especial à Vanessa que me ajudou e me deu força nos momentos mais difíceis do desenvolvimento desse trabalho. A minha filha, por alegrar e a minha vida e me impulsionar a ser sempre um exemplo melhor. Aos meus amigos, sempre presentes, mesmo que remotamente, nesse período complexo em que vivemos, sempre disposto a ouvir quando o desespero bate. E um especial agradecimento à minha orientadora que sempre gentilmente corrigiu e auxiliou o desenvolvimento deste trabalho, além de me encoraja a continuar quando eu pensava em desistir. Agradeço também ao professor David Quesada López, autor do pacote dbnR que gentil e atenciosamente me auxiliou na correção de bugs e dirimindo dúvidas.

RESUMO

Muitos historiadores consideram a agricultura como a tecnologia que moldou o desenvolvimento humano, permitindo a concentração populacional e o surgimento das sociedades modernas. A segurança alimentar depende fortemente da agricultura enfrentar os desafios colocados pelo crescimento populacional e pelas mudanças climáticas. A tecnologia da informação é uma importante aliada para superar essas dificuldades de forma sustentável, aumentando a produtividade com melhor aproveitamento dos recursos disponíveis. A gestão adequada dos recursos depende da disponibilização dos dados necessários para produtores e agentes públicos. Nesse contexto, as técnicas de previsão e estimativa da produtividade agrícola são ferramentas relevantes para a tomada de decisão. Porém, além de gerar valores de visualização de produtividade, um modelo deve ser capaz de descrever os principais fatores que afetam essa produtividade e verificar a distribuição espaço-temporal desses fatores. Os modelos de inferência probabilística são técnicas que podem ser adequados à esta tarefa, pois são modelos explicáveis e podem incorporar novos dados à medida que surgem. Este trabalho apresenta um estudo das principais técnicas aplicadas à estimativa e previsão dos resultados da produção agrícola e desenvolve um pacote R para inferência de dados de produção com redes Bayesianas estáticas distribuídas espacialmente e redes Bayesianas dinâmicas, capazes de prever resultado de produção agrícola. Nos testes executados com dados gerados de forma controlada, os modelos criados por algoritmos de inferência nem sempre refletiram a estrutura esperada. As redes criadas a partir de conjuntos de arcos pré definidos apresentaram desempenho superior em relação às redes inferidas por dados, ainda assim com desempenho aquém do desejável para um modelo destinado para uso em casos reais. O trabalho desenvolvido introduz um novo método para tratar de um problema que é simultaneamente importante e difícil, mas que mostra que ainda existe necessidade de melhorias nos modelos usados.

Palavras-chave: Agricultura digital; Inferência probabilística; Ciência de dados.

ABSTRACT

Many historians consider agriculture as the technology that shaped human development, allowing population concentration and the emergence of modern societies. Food security depends heavily on agriculture meeting the challenges posed by population growth and climate change. Information technology is an important ally for overcoming these difficulties in a sustainable way, increasing productivity with better use of available resources. Proper resource management depends on the availability of necessary data for producers and public agents. In this context, techniques for forecasting and estimating agricultural productivity are relevant tools for decision-making. However, in addition to generating productivity preview values, a model must be able to describe the main factors that affect this productivity and verify the space-time distribution of these factors. Probabilistic inference models are techniques that can provide the best results, as they are explainable models and can incorporate new data as they arise. This work presents a study of the main techniques applied to estimating and predicting agricultural production results and develops an R package for production data inference with spatially distributed static Bayesian networks and dynamic Bayesian networks, capable of predicting agricultural production results. In tests performed with data generated in a controlled manner, the models created by inference algorithms did not always reflect the expected structure. The networks created from sets of pre-defined arcs presented a superior performance in relation to the networks inferred by data, even so with less than desirable performance for a model intended for use in actual cases. The developed work introduces a new method to deal with a problem that is both important and difficult, which shows that there is still a need for improvements in the models used.

Keywords: Digital Agriculture; Probabilistic Inference; Data Science.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AGDD	*Accumulated Growing Degree Days*
A-ES	*Adaptive Evolution Strategies*
AI	*Aridity Index*
BPNN	*Backpropagation Neural Network*
CART	*Classification and Regression Tree*
CPP	*Crop Progress Percentage*
DAG	*Directed Acyclic Graph*
DBN	*Dynamic Bayesian Networks*
DNN	*Deep Neural Network*
EMBRAPA	Empresa Brasileira de Pesquisa Agropecuária
EVI	*Enhanced Vegetation Index*
GPR	*Gaussian Process Regression*
GBM	*Gradient Boosting Model*
HC	*Hill-Climbing*
HDP-HMM	*Hierarchical Dirichlet Process Hidden Markov Model*
HMM	*Hidden Markov Models*
IA	Inteligência Artificial
INMET	Instituto Nacional de Meteorologia
IoC ou IoT	Internet das Coisas ou *Internet of Things*
KNN	*k-Nearest Neighbors*
LRS	*Length of the Rainy Season*
MAE	*Mean Absolute Error*
MAPE	*Mean Absolute Percent Error*
MARS	*Multivariate Adaptive Regression*

MMHC	*Max-Min Hill-Climbing*
MPE	*Mean Percentage Error*
MSE	*Mean Squared Error*
MLP	*Multilayer Perceptron*
NDVI	*Normalized Difference Vegetation Index*
NN	*Neural Network*
PRE	*Proportional Reduction of Error*
PLSR	*Partial Least Square Regression*
R^2	Coeficiente de Determinação
RF	*Random Forest*
RNA	*Rede Neural Artificial*
RRMSE	*Relative Root Mean Squared Error*
RMSE	*Root Mean Squared Error*
RMSD	*Root Mean Squared Deviation*
SNN	*Self Normalizing Neural Networks*
SOI	*Southern Oscillation Index*
SPEI	*Standardised Precipitation Evapotranspiration Index*
SVM	*Support Vector Machine*
TVDI	*Temperature Vegetation Dryness Index*
UNIPAMPA	Universidade Federal do Pampa
XgBoost	*Extreme Gradient Boosting*

SUMÁRIO

1 INTRODUÇÃO

1.1 Previsão de resultados de produção agrícola

A agricultura é considerada por muitos historiadores como a tecnologia que moldou o desenvolvimento humano, permitindo a transição de pequenas tribos de caçadores coletores para grandes assentamentos sedentários. O surgimento das cidades deu-se a partir daí, graças ao crescimento e à concentração populacional propiciados pela maior capacidade de produção de alimentos. A evolução das técnicas de exploração do solo tornou-se, pois, obrigatória para a nossa sobrevivência como espécie, levando a uma interdependência entre a agricultura e o crescimento demográfico (HARARI, 2017; BOCQUET-APPEL, 2011).

Os obstáculos vencidos na agricultura pavimentaram o caminho para a sociedade moderna e globalizada. Entretanto, apresentam-se novos desafios a superar, como a redução de desperdício de recursos e insumos, o aumento de produtividade com minimização dos impactos ambientais, tais como emissão dióxido de carbono, erosão dos solos e contaminação de recursos hídricos.

As projeções do crescimento da demanda por alimentos para as próximas décadas e o impacto ambiental positivo da redução do desflorestamento não são uma ameaça à disponibilidade de alimentos, desde que a agricultura adapte-se por meio da intensificação da produção nas terras já disponíveis para cultivo (SCHNEIDER et al., 2011).

A chamada agricultura de precisão é um conjunto de práticas baseadas em tecnologias no campo objetivando principalmente o aumento da produtividade, por meio da análise dos resultados produtivos, levando em conta a variabilidade espacial existente nas áreas plantadas (MAPA, 2011). A agricultura digital caracteriza-se pelo uso de recursos de Tecnologias de Informação e Comunicação (TIC) atuando no levantamento e no processamento de grande quantidade de dados coletados ao longo do tempo nas cadeias produtivas. Com vistas à agricultura de precisão, a agricultura digital envolve a aplicação de técnicas como sensoriamento remoto ou *in loco* fazendo uso de Internet das Coisas (IoT), sistemas de telecomunicação, *Global Positioning System* (GPS) e sistemas de gestão e análise de dados, entre outras (SOUZA et al., 2020).

Apesar dos investimentos na modernização dos processos na agricultura, seus resultados dependem de um conjunto de variáveis que nem sempre são de fácil previsibilidade ou mensuração. Fatores como a composição do solo, condições climáticas

e relevo influenciam no desempenho das safras. As variáveis climatológicas, relativas à dinâmica de grande escala temporal e espacial, tais como a média histórica de precipitação, temperatura média etc., e as variáveis meteorológicas, associadas às variações de curto espaço de tempo, como precipitação, temperatura mínima e máxima diária, entre outros fatores, são marcadas por incerteza em razão da complexidade inerente ao sistema atmosférico, além das mudanças climáticas acarretadas pelo aquecimento global que tornam cada vez mais frequente a ocorrência de eventos meteorológicos extremos como estiagens prologadas, tempestades de grande intensidade e alteração na dinâmica de distribuição de chuvas (BELLPRAT et al., 2019; OTTO et al., 2018).

A obtenção de dados meteorológicos pode ser realizada por meio de bancos de dados muitas vezes disponíveis de maneira pública, como por exemplo o Banco de Dados Meteorológicos do Instituto Nacional de Meteorologia (INMET), sendo possível acessar dados de umidade relativa do ar, temperatura, insolação e pluviosidade. A obtenção de dados pedológicos pode ser realizada pela coleta de amostras e análise laboratorial, entretanto existem técnicas de sensoriamento remoto que permitem estimar características de solo com técnicas não invasivas.

Dentre os aspectos meteorológicos, a precipitação é crucial para a agricultura, afetando diretamente os resultados das colheitas. A variação na média anual nas chuvas influi também nesses resultados. Entretanto, cabe notar que somente o dado da média pluviométrica anual não é suficiente para a previsão do resultado da colheita, pois ainda que a média não sofra grande variação, sua distribuição pode ter sido irregular, com chuvas fortes e longos períodos de estiagem, o que obviamente surtirá efeitos na quantidade e qualidade das colheitas. A razão para isso é que a necessidade por água pelas cultivares varia ao longo do seu ciclo de vida (TORRES; HOWITT; RODRIGUES, 2019). Assim, é relevante que os modelos desenvolvidos para uma adequada previsão de resultados e análises de variabilidade sejam capazes de obter dados meteorológicos detalhados sobre a distribuição de chuvas durante as safras. Outra questão importante a ser considerada é a extensão da área plantada. Tanto características meteorológicas como pedológicas podem variar dentro de uma mesma plantação. Identificar as áreas onde a produtividade é muito baixa ou nula pode gerar informações importantes para a tomada de decisões relativas ao planejamento do plantio.

Previsão é o esforço para antever cenários futuros com base em dados presentes ou passados, usando técnicas de inferência. Previsão é diferente do planejamento, pois o planejamento se preocupa em dizer como o mundo deve parecer enquanto a previsão

descreve como ele irá se parecer, sendo a previsão instrumento para um planejamento eficiente (ARMSTRONG, 2001). Embora previsão (i.e., ver antecipadamente) e predição (i.e., dizer antecipadamente) sejam considerados sinônimos na Língua Portuguesa, algumas referências apresentam os termos como conceitualmente diferentes: previsão é o esforço de realizar estimativas de cenários futuros com base em cálculos e dados, enquanto predição indica que acontecimentos futuros ocorrerão com ou sem base em dados passados (KUNGU, 2018). A diferença conceitual colocada na referência anterior não é significativa no contexto deste trabalho, visto que modelos probabilísticos funcionam de forma equivalente em abordagens frequentistas (quando as probabilidades são calculadas a partir de dados coletados sobre situações passadas) ou Bayesianas (quando as probabilidades são definidas a partir da crença dos especialistas na ocorrência conjunta ou condicional de eventos distintos). Note-se que, no contexto da Agropecuária, a abordagem frequentista pode ser inviável pela dificuldade de coleta de dados e pela falta de representatividade dos dados coletados em cobrir todas as situações possíveis. De toda forma, a ideia é a previsão (pré-visão) de um resultado de colheita, quando a diferença for significativa.

Atualmente, ferramentas computacionais baseadas em Inteligência Artificial (IA), como redes neurais e outras estratégias de aprendizado profundo (ou *deep learning*), são aplicadas na previsão de cenários de várias atividades. As áreas de Biologia (HIE et al., 2021), construção civil (MUKHERJEE; Nag Biswas, 1997), Medicina (KHAN et al., 2001), Química (JORJANI; Chehreh Chelgani; MESROGHLI, 2008) e Agricultura (UNO et al., 2005) são exemplos de áreas em que técnicas de IA estão sendo correntemente aplicadas. Entretanto, tais técnicas possuem uma abordagem hermética, onde a relação aprendida entre as entradas e as saídas desses sistemas não são facilmente identificadas e, por essa razão, não possuem a propriedade de explicabilidade. Uma técnica cujo resultado possa ser explicado aos usuários tem uma probabilidade maior de ser considerada útil, pois a mudança em uma forma de manejo ou execução de uma atividade deve fazer sentido para quem vai receber as informações sobre os possíveis resultados dessa mudança.

Conforme abordado neste texto, a agricultura se coloca como uma atividade de vital importância para a humanidade, por conta de sua importância na segurança alimentar. A não redução dos impactos causados pelo mau emprego ou desperdício de recursos gera prejuízos ecológicos, sociais e financeiros, de maneira que novos estudos e técnicas estão sendo desenvolvidos no sentido de modernizar os processos produtivos. No contexto da agricultura digital, a coleta e o tratamento de dados e desenvolvimento

de sistemas de inferência auxiliam a tomada de decisões pelos agricultores. Diante deste panorama, o trabalho parte da hipótese de que é viável desenvolver um pacote para a linguagem R, baseado em redes Bayesianas, que permita prever o resultado da produção agrícola, bem como identificar as variações espaciais e temporais dos resultados.

1.2 Objetivos do trabalho

O objetivo geral deste trabalho é desenvolver um pacote R, com base em redes Bayesianas estáticas e dinâmicas, que permita a previsão de resultado de colheita, bem como identifique a variabilidade espacial e temporal das culturas a partir de divisões arbitrárias da área de produção e das fases fenológicas.

Os objetivos específicos do trabalho estão listados a seguir:

1. Apresentar uma revisão da literatura, referente aos 5 anos anteriores a data de sua realização (15 de fevereiro de 2021), sobre aplicação de modelos probabilísticos em previsão de resultados e análise de variabilidade de colheitas.
2. Construção de um modelo para a análise espacial e temporal da variabilidade da produção em partes específicas da área de produção, que possa ser disponibilizado na forma de um pacote R.
3. Validar o modelo com dados gerados a partir de distribuições probabilísticas bem definidas, para verificação das capacidades do modelo proposto, avaliando a acurácia e a precisão do modelo.

São resultados esperados deste trabalho:

1. A criação de uma ferramenta de previsão de resultados de colheita que possa apoiar produtores rurais em seus processos de tomada de decisão, demonstrando a variabilidade temporal e espacial da produção.
2. A submissão do pacote para distribuição junto ao repositório oficial da linguagem R[1] e sua disponibilização no *Github*[2].
3. O aumento da produção tecnológica socialmente relevante dos discentes do curso de Engenharia de Computação da UNIPAMPA.

O texto está estruturado em seis capítulos, incluindo esta introdução (Capítulo 1),

[1] https://cran.r-project.org
[2] https://github.com

em que foram apresentados o problema a ser atacado nesta pesquisa, a justificativa, e a estratégia de abordagem para execução da solução proposta.

No Capítulo 2 estão detalhadas as fases da pesquisa (Seção 2.1), o método utilizado na revisão de escopo da literatura (Seção 2.2), além de apresentar o ferramental tecnológico (Seção 2.3) aplicado na construção e validação do modelo proposto (Seção 2.4).

No Capítulo 3, a partir dos trabalhos encontrados como resultado da revisão de escopo da literatura, são apresentadas as reflexões e as bases teóricas sobre as quais este trabalho se sustenta. O capítulo está estruturado em três seções: na Seção 3.1 é abordada a fundamentação teórica com apresentação de algumas definições e conceitos. Na Seção 3.2 são apresentadas as principais métricas utilizadas para validação de modelos. Já a Seção 3.3 se dedica à apresentação dos trabalhos correlatos que retratam quais são as técnicas utilizadas na previsão de resultado de colheita (Seção 3.3.1) e as aplicações de modelos probabilísticos a temas ligados à agricultura (Seção 3.3.2).

No Capítulo 4 o modelo proposto é detalhado. O capítulo está estruturado em duas seções: a Seção 4.1 apresenta as funcionalidades disponíveis e, na Seção 4.2, o processo de geração de dados para validação e testagem é detalhado.

No Capítulo 5 são discutidos os resultados obtidos, analisando a geração e tratamento de dados (Seção 5.1), as topologias das redes geradas (Seção 5.2) e as métricas calculadas (Seção 5.3), comparando o desempenho dos modelos construídos entre si e com modelos de trabalhos correlatos.

Por fim, o Capítulo 6 apresenta as considerações finais sobre o trabalho, com a revisão dos objetivos atingidos e reflexões a respeito de possíveis desdobramentos do trabalho.

2 MATERIAL E MÉTODOS

2.1 Caracterização e fases da pesquisa

O trabalho desenvolvido neste TCC caracteriza-se como uma pesquisa de natureza aplicada com objetivos exploratórios, sendo composto das seguintes fases e procedimentos:

1. Revisão de escopo da literatura, focada em dois aspectos principais: (i) técnicas usadas para previsão e análise de variabilidade dos resultados de produção agrícola e (ii) utilização de modelos probabilísticos em sistemas de produção agrícola.
2. Estudo de modelos probabilísticos e de seus algoritmos de inferência dedutiva e abdutiva.
3. Construção do pacote R para previsão e análise espacial e temporal dos dados de produção.
4. Geração de dados a partir de distribuições probabilísticas conhecidas, para teste e validação do modelo.
5. Análise e síntese dos resultados obtidos.
6. Elaboração do texto ora exposto nesta monografia, concomitante com os procedimentos acima.

2.2 Revisão da literatura

A revisão da literatura tem o intuito de reforçar o conhecimento dos termos, definições e conceitos relacionados à área de conhecimento estudada, bem como situar os objetivos de pesquisa levantados na Seção 1.2 em relação ao estado da arte. O procedimento escolhido foi a revisão de escopo da literatura. Os trabalhos de Conforto, Amaral e Silva (2011) e Neiva e Silva (2016) apresentam roteiros com orientações para a realização de revisão sistemática da literatura, adaptados neste texto para uma revisão de escopo. Com base nestes roteiros, foram adotados os seguintes procedimentos: inicialmente definiram-se as questões norteadoras da revisão, em seguida determinou-se as palavras e termos referentes ao tema da pesquisa, na sequência estabeleceu-se as *strings* de busca, sendo estas testadas para verificação de sua cobertura e precisão. Em seguida foram definidos os repositórios de publicações, realizadas as buscas e aplicados

os critérios de exclusão/inclusão formando assim a base de publicações que são estudadas mais aprofundadamente, permitindo compor a base teórica necessária para este trabalho, conforme melhor detalhado no Capítulo 3.

A revisão da literatura foi organizada a partir das seguintes questões de pesquisa:

Q1 Quais são as técnicas utilizadas na previsão de resultado de colheita descritas na literatura?

Q2 Existem implementações que utilizam modelos probabilísticos como abordagem a temas ligados à agricultura?

A Tabela 1 apresenta o conjunto de palavras-chave escolhidas para a busca do referencial bibliográfico usado neste trabalho. A coluna da esquerda apresenta os termos-chave para a busca e a coluna da direita apresenta os seus respectivos sinônimos. Os sinônimos são importantes, visto que a busca é realizada somente nos campos de título, palavras-chave e resumo, e termos distintos podem ser usados para expressar a mesma ideia.

Tabela 1 – Definição dos termos de busca

Termo	**Sinônimos**
Previsão	previsibilidade, previsão, *forecast*, *prediction*
Agricultura	lavoura, cultivo, agrícola, agrário, *agriculture*, *farming*
Colheita	produção, safra, *harvest*, *crop yield*
Variabilidade	oscilação, variação, *variability*
Modelos probabilísticos	HMM, *Hidden Markov Model*, Bayesian networks, belief networks, cadeias escondidas de Markov, modelo oculto de Markov, cadeias ocultas de Markov, modelo de Markov, redes de Bayes, redes Bayesianas

Fonte: Autor (2022)

As *strings* de busca nas fontes de pesquisa foram construídas por meio da conjunção de cláusulas que contêm a disjunção dos sinônimos em seu interior. Resultando na seguinte construção disponibilizada aqui a fim de possibilitar a reprodutibilidade do levantamento, esclarecendo que a *string* 1 busca levantar trabalhos para elucidar Q1 e as *strings* 2 e 3 ligam-se à Q2:

***String* 1** : (predição OR previsibilidade OR previsão OR forecast OR prediction) AND (agricultura OR lavoura OR cultivo OR agrícola OR agrário OR agriculture OR farming) AND (colheita OR produção OR safra OR harvest OR "crop yield") AND (variabilidade OR oscilação OR variação OR variability)

***Strings* 2** : ("Cadeias Ocultas de Markov" OR "Cadeias Escondidas de Markov" OR

"modelos Escondidos de Markov" OR HMM OR "hidden Markov Model" OR "modelo oculto de Markov" OR "Cadeias de Markov" OR "Markov Model" OR "modelo de Markov") AND (agricultura OR lavoura OR cultivo OR agrícola OR agrário OR agriculture OR farming OR colheita OR produção OR safra OR harvest OR "crop yield")

***Strings* 3** : ("redes bayesianas" OR "Bayesian networks" OR "Bayes network" OR "Bayes net" OR "belief network" OR "decision network") AND (agricultura OR lavoura OR cultivo OR agrícola OR agrário OR agriculture OR farming OR colheita OR produção OR safra OR harvest OR "crop yield")

As fontes de referências bibliográficas utilizadas foram *ACM Digital Library* (https://dl.acm.org/), *IEEE Xplorer* (https://ieeexplore.ieee.org/), *ScienceDirect* (https://www.sciencedirect.com/) e *Wiley Online Library* (https://onlinelibrary.wiley.com/) .

A busca pelos referenciais foi feita somente nos campos de título, palavras-chave e resumo, com exceção do repositório da *IEEE Xplorer* onde foi necessário definir a busca para todo texto, pois ocorreram inconsistências no retorno das buscas.

Essa escolha foi feita para minimizar a quantidade de trabalhos retornados que mencionam os termos de busca como parte do levantamento bibliográfico ou como exemplos de aplicação de alguma técnica alheia aos problemas de pesquisa definidos neste trabalho. Assume-se que os termos deverão aparecer em algum desses campos se o trabalho for relacionado. Priorizou-se os trabalhos publicados nos últimos cinco anos aproximadamente, ou seja janeiro de 2016 até a data da realização das buscas 15 de fevereiro de 2021. Não foram levantadas novas publicações em respeito ao cronograma de desenvolvimento do trabalho. Os critério de inclusão do trabalho na revisão são os seguintes:

- Trabalhos relativos à previsão de resultado de colheita, independente da técnica utilizada.
- Trabalhos que abordem análises sobre variabilidade espacial ou temporal de colheitas.
- Trabalhos que usem modelos probabilísticos na agricultura, mesmo que fora do domínio de previsão de resultados de colheita.

Foram realizadas as buscas nos repositórios listados obtendo-se 993 resultados. Ao concluir a primeira triagem, restaram apenas 214 artigos, visto que foram observados

os títulos para então descartar aqueles não são relacionados aos temas tratados neste trabalho. Por último, foi realizada a leitura dos resumos dos trabalhos restantes, mais uma vez eliminando as publicações que não tratam das questões aqui abordadas. Desta segunda triagem restaram 59 publicações que, estudadas mais aprofundadamente, permitem a formação do embasamento teórico necessário e as discussões a serem realizadas no capítulo seguinte de referencial teórico. A Tabela 2 detalha os resultados das buscas e da aplicação dos critérios de inclusão por repositório.

Tabela 2 – Resultados por repositório

Repositório	
***ACM Library* (https://dl.acm.org)**	
Bruto	372
Revisão do título	51
Revisão do resumo	9
***IEEE Xplorer* (https://ieeexplore.ieee.org/)**	
Bruto	74
Revisão do título	24
Revisão do resumo	13
ScienceDirect (https://www.sciencedirect.com/)	
Bruto	164
Revisão do título	53
Revisão do resumo	26
***Wiley Online Library* (https://onlinelibrary.wiley.com/)**	
Bruto	383
Revisão do título	86
Revisão do resumo	11

Fonte: Autor (2022)

2.3 Ferramental tecnológico

No que toca ao material necessário para o desenvolvimento deste trabalho, considerando tratar-se de pesquisa fortemente baseada em dados, com necessidade de

análises estatísticas, deliberou-se pela utilização da linguagem de programação R[1] e do ambiente integrado de desenvolvimento RStudio[2]. Além de sua versatilidade em lidar com dados e funções estatísticas, ser *open source* com comunidade ativa e um grande repositório de pacotes, conta com pacotes específicos para aplicações que usam redes Bayesianas disponíveis na Internet.

Dentre os pacotes utilizados é preciso destacar o *bnlearn*[3](versão 4.8.1). O pacote desenvolvido por Scutari (2010) permite a criação de redes Bayesianas a partir do grafo dirigido acíclico (DAG) desenhado pelo usuário ou gerado a partir de algoritmos de aprendizado de estrutura de rede. O aprendizado de estrutura consiste em encontrar o DAG que codifica a estrutura de dependência de um conjunto de dados com *n* observações. Os algoritmos disponíveis no pacote *bnlearn* enquadram em três classes: (i) algoritmos baseados em restrições, que identificam restrições de independência condicional com testes estatísticos e vinculam nós que não são encontrados para que sejam independente; (ii) algoritmos baseados em pontuação onde são aplicadas técnicas gerais de otimização, em que cada DAG candidato recebe uma pontuação de rede maximizada como a função objetivo; e (iii) algoritmos híbridos possuem uma fase restrita implementando uma estratégia baseada em restrições para reduzir o espaço de DAG candidatos e uma fase de maximização implementando uma estratégia baseada em pontuação para encontrar o DAG ideal no espaço restrito.

Além da funcionalidade de aprendizado de estrutura de rede, o pacote também oferece funções para tratamento do conjunto de dados, permitindo a discretização e distribuição em classes dos dados contínuos. Por meio da função *dicretize* é possível dividir o conjunto de dados a partir dos métodos *interval*, *quantile* e *hartemink*[4]. O método *interval* foi o utilizado neste trabalho por ter gerado menos distorções, em relação aos outros modos, nas distribuições de classes das variáveis ao longo das fases fenológicas.

Para a criação das redes dinâmicas foi utilizado o pacote *dbnR*[5](versão 0.7.8), desenvolvido por López e Castilla (2019) a partir de extensões das funções do pacote *bnlearn*, que permite gerar redes dinâmicas de ordem markovianas arbitrárias. O pacote oferece três algoritmos de aprendizado de rede: *dynamic max-min hill climbing*, *swarm optimization algorithm for higher-order* e uma variação deste último, escalável.

[1]https://www.r-project.org/
[2]https://www.rstudio.com/
[3]https://cran.r-project.org/web/packages/bnlearn/index.html
[4]Método de informação mútua par a par de Hartemink (2001)
[5]https://CRAN.R-project.org/package=dbnR

Conta também com função de previsão e ferramenta de visualização gráfica das redes construídas.

As funções disponíveis no repositório do *Github* *KaikeWesleyReis/bnlearn-multivar-prediction-metrics*[6] foram utilizadas para a gerar as métricas de avaliação dos modelos de redes estáticas. A avaliação das redes dinâmicas foi realizada com o auxílio da função *defaultSummary* do pacote *caret*[7](versão 6.0-93). As métricas calculadas referentes ao desempenho das redes são apresentadas no Capítulo 5.

No tocante ao tratamento dos dados, foram utilizadas as funções *sample.split* do pacote *caTools*[8](versão 1.18.2) para a divisão do conjunto de dados em treino e teste, *preProcess* e *predict* do pacote *caret* para normalização dos dados, redimensionando os dados num intervalo entre zero e um e a função *discretize* do pacote *bnlearn* foi utilizada para discretização dos dados, classificando o conjunto de dados em três classes (baixa, média e alta) ou cinco classes (baixa, média-baixa, média, média-alta e alta), conforme parâmetros do usuário final.

2.4 Construção e validação do modelo

Dados agropecuários provêm de diferentes fontes e nem sempre são coletados no mesmo tempo/espaço. Dados pedológicos não são coletados em todas as regiões de uma área, por exemplo, tanto pelo custo como pela inviabilidade física. Sensores são posicionados em diferentes locais do espaço e os dados que não dizem respeito à sua exata posição devem ser inferidos ou interpolados. Alguns dados são coletados periodicamente (altura das plantas, por exemplo), mas nem sempre nos mesmos intervalos ou locais. Essas características fazem com que os dados estejam disponíveis em certas posições do espaço/tempo, mas não em outras. Modelos para inferência de dados, nessas condições, precisam ser flexíveis e lidar com essas diferenças.

Modelos probabilísticos quase sempre possuem esta flexibilidade: o número e o tipo das variáveis que podem ser usadas não é limitado. Versões diferentes do mesmo modelo podem fazer uso de variáveis distintas, quando uma ou mais delas não estiverem disponíveis. Redes Bayesianas tem essa característica e, por isso, foram escolhidas

[6]https://github.com/KaikeWesleyReis/bnlearn-multivar-prediction-metrics/
[7]https://CRAN.R-project.org/package=caret
[8]https://cran.r-project.org/web/packages/caTools/index.html

para este trabalho. Adicionalmente, permitem processos de inferência tanto dedutivo, estabelecendo um processo de previsão, como abdutivo, estabelecendo a capacidade de análise do processo produtivo. Entende-se como dedutivo a análise que flui das causas para as consequências, enquanto que o abdutivo parte das consequências – ou observações – para as causas.

Inicialmente, previa-se usar os bancos de dados do INMET para obtenção dos dados meteorológicos e climáticos da região de estudo e os dados da produção de soja e das características pedológica de uma área da EMBRAPA. No entanto, a área em questão deixou de ser usada para produção de soja e os dados disponibilizados diziam respeito a somente quatro safras, de forma que os dados não cobriam uma fração significativa das possibilidades de valores das variáveis envolvidas. Como resultado, provavelmente não existiriam dados amostrais suficientes que permitissem realizar a validação do modelo. Sendo assim, optou-se por uma abordagem baseada em dados pseudo-aleatórios gerados de forma controlada, onde as relações de causa-consequência estejam bem estabelecidas. Esta nova abordagem se justifica pela necessidade de validação das topologias das redes inferidas, o que não seria possível com um conjunto de dados em que não se conhece a relação entre as variáveis e a produção.

A aplicação do modelo à dados reais disponíveis em bancos públicos não se justifica pois, ainda que se possa validar o modelo a partir das métricas de desempenho, em relação a previsão de resultados, o desconhecimento prévio das relações causais entre as variáveis disponíveis e o resultados da produção inviabilizaria validar os modelos gerados na realização da tarefa inversa, que é estimar as causas, isto é o peso de cada variável tem no resultado, a partir dos resultados de produção disponível, sendo este um dos desdobramentos propostos para este trabalho.

O modelo ainda leva em conta a divisão espacial e temporal da lavoura permitindo considerar as diferentes fases fenológicas do cultivar e a variabilidade espacial da produção. Estes aspectos são contemplados como parâmetros configuráveis pelo usuário do pacote.

Como este trabalho não utiliza dados reais, não está restrito às variáveis para as quais temos valores disponíveis. De toda forma, considerando a aplicabilidade dos modelos, a geração de dados intenta emular as variáveis que reconhecidamente influenciem a produtividade agrícola e cuja mensuração seja viável. Exemplifica-se a precipitação, a altura das plantas e insolação total, cujos valores acumulados crescem ao longo da evolução da plantação, a temperatura cuja variação se dá de forma mais aleatória

e as variáveis pedológicas que se mantém constantes ao longo de toda a plantação. No Capítulo 4 a construção do pacote e a geração dos dados para a validação do modelo é detalhada.

3 REFERENCIAL TEÓRICO

3.1 Modelos de inferência probabilísticos

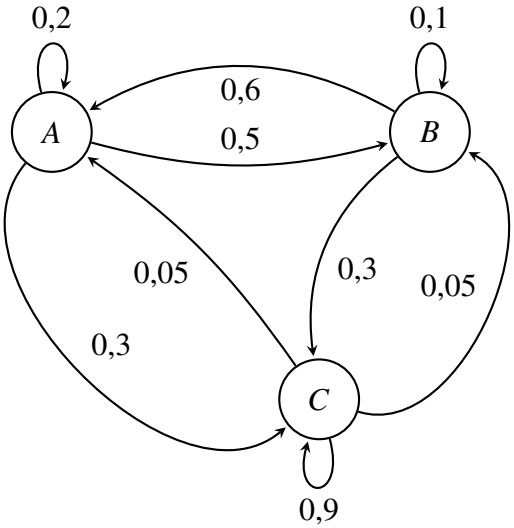
Batizados em homenagem ao matemático russo Andrei Andreyevich Markov (1856-1922), os modelos de Markov são um tipo de processo estocástico em que o estado atual do processo depende apenas do estado imediatamente anterior. Um processo estocástico é aquele que pode ser descrito como uma cadeia de eventos probabilísticos em que eventos futuros da cadeia dependem da sequência dos eventos anteriores, em contraponto aos processos determinísticos em que a evolução do sistema é totalmente definida ou aos sistemas não determinísticos em que a probabilidade dos eventos futuros não dependem dos eventos passados.

Uma Cadeia de Markov é um sistema de transição de estados em que as transições entre seus possíveis estados representam a probabilidade daquela transição ser gerada ao longo do andamento do processo. A Definição 1 (FERREIRA, 2019) apresenta formalmente essa ideia.

> **Definição 1 (Cadeia de Markov)** *Uma* cadeia de Markov *é uma tupla* $M = (K, \pi_0, \pi)$*, onde*
>
> - K *é um conjunto contável de estados,*
> - $\pi_0 : K \to [0,1]$ *é uma função de probabilidade, em que* $\pi_0(k)$ *representa a probabilidade do sistema iniciar no estado k, para todo* $k \in K$*,*
> - $\pi : K \times K \to [0,1]$ *é a função de transição, em que* $\pi(k,k')$ *é o valor da probabilidade condicional* $P(k'|k)$*, ou a probabilidade do sistema transitar para o estado* k' *dado que ele está no estado k e para todo* $k \in K$ *tem-se que* $\sum_{k' \in K} \pi(k,k') = 1$*.*

Uma cadeia de Markov também pode ser representada como um grafo, facilitando assim a visualização de seus estados e transições. A Figura 1 apresenta uma cadeia de Markov, em sua representação como um grafo. A cadeia representada tem três estados (A, B e C) indicados como os vértices ou nodos e as probabilidades de transição de estados estão dispostas sobre as arestas. Ou seja, para quaisquer dois nodos $x, y \in \{A, B, C\}$, o valor que rotula o arco entre os nodos x e y é a probabilidade do sistema passar para o estado y dado que ele está no estado x, ou a probabilidade condicional $P(y|x)$.

Figura 1 – Representação de uma Cadeia de Markov como um grafo



Fonte: Ferreira (2021)

Cadeias de Markov são úteis para determinar a probabilidade da ocorrência de uma sequência de eventos, bastando conhecer a distribuição de probabilidade em um determinado momento, sem a necessidade de conhecer os estados anteriores. Assim, é possível predizer a probabilidade de atingir estados subsequentes. Cadeias de Markov têm muitas aplicações para modelagem de processos do mundo real, Yang et al. (2019), em seu levantamento, identificaram a aplicação de cadeias de Markov em áreas como Bioinformática, Economia e finanças, Engenharia Genética, Ecologia, geração de energia, manutenção de vias de transporte, Agricultura, Mineração, Engenharia Civil, entre outras.

Cadeias de Markov podem ser usadas quando todos os estados possíveis de um sistema são bem definidos e as transições entre eles podem ser computadas. Os estados de um modelo de Markov usualmente representam eventos[1] relacionados ao sistema. Eventos determinam ocorrências específicas do sistema. Contudo, as relações de causa e efeito entre as variáveis de um sistema podem determinar que diferentes eventos possam ser dependentes entre si. Em sistemas de produção agropecuária, em que as variáveis do sistema solo-planta-animal-atmosfera possuem um alto grau de dependência, esse tipo de situação se apresenta com frequência. Nesses casos, as relações probabilísticas de causa e efeito podem ser descritas por meio de uma extensão do modelo de cadeias de Markov. Essa extensão, denominada de modelo oculto de Markov (do Inglês *Hidden Markov Model*, ou simplesmente HMM), é especialmente útil quando as causas podem

[1] Um evento é um subconjunto de um espaço amostral definido em um experimento. Um experimento, por sua vez, consiste em um espaço amostral e uma função de probabilidade sobre esse conjunto de pontos amostrais.

ser mensuradas e a consequência podem ser preditas instantaneamente, mas não podem ser tão facilmente determinadas em eventos futuros.

Um modelo oculto de Markov permite explicitar as relações existentes entre eventos observáveis (que envolvem as variáveis mensuráveis do sistema) e eventos ocultos (que envolvem aquilo que usualmente se quer predizer e, portanto, não é mensurável). HMM são usadas para estudar observações em uma série temporal discreta em que os estados (ocultos) possuem probabilidades de transição e cada estado emite símbolo(s) (visíveis) de acordo com a probabilidade de emissão do estado (MOR; GARHWAL; KUMAR, 2021). A Definição 2 (FERREIRA, 2019) apresenta formalmente essa estrutura.

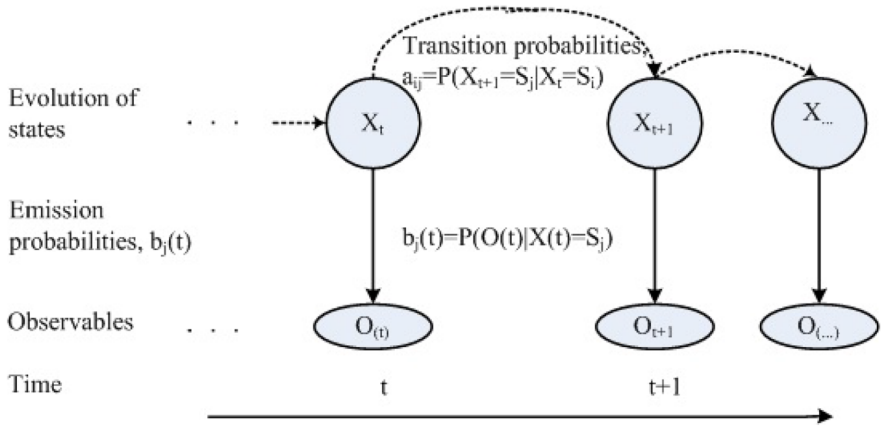
Definição 2 (Modelo oculto de Markov) *Um* modelo oculto de Markov *é uma tupla* $M = (K, V, A, B, \pi)$*, onde*

- $K = \{k_1, k_2, ..., k_n\}$ *é um conjunto contável de estados,*
- $V = \{v_1, v_2, ..., v_n\}$ *é o conjunto de valores observáveis a cada instante,*
- *A é uma matriz de transição de probabilidades, em que cada elemento* $\{a_{ij}\}$ *da matriz A corresponde ao valor da probabilidade* $P(q_{t+1} = S_j | q_t = S_i)$*, ou à probabilidade do sistema estar no estado* S_j *no instante* $t+1$ *dado que estava no estado* S_i *no instante t,*
- *B é a matriz de emissão, representando distribuição de probabilidade de observação, em que cada elemento* $\{b_j(k)\}$ *de B corresponde à probabilidade* $P(O_t = V_k | q_t = S_j)$*, ou à probabilidade do símbolo* V_k *ser observado no instante t, dado que o sistema está no estado* S_j *nesse mesmo instante,*
- $\pi = (\pi_1, \pi_2, ..., \pi_n)$*, onde cada* $pi_j = P(q_1 = S_j)$*, ou a probabilidade do sistema estar no estado* S_j *no tempo inicial t=1,*
- $\lambda = (A, B, \pi)$ *é a notação compacta de HMM.*

A emissão diz respeito àquilo que é observável no sistema. Em sistemas de reconhecimento de voz, por exemplo, a emissão é o conjunto de frequências emitidas pela voz, mensurável, e o estado oculto é o fonema correspondente àquelas frequências capturadas (oculto). A figura Figura 2 apresenta um representação gráfica de um modelo oculto de Markov, onde é possível visualizar, na parte superior, a evolução dos estados

ao longo do tempo, de acordo com as probabilidades de transição e, na parte inferior as observações, de acordo com as probabilidades de emissão.

Figura 2 – Representação gráfica de um modelo oculto de Markov



Fonte: Mor, Garhwal e Kumar (2021)

Inicialmente, este trabalho visava estender o modelo oculto de Markov usado em Ferreira (2019) para previsão da produção agrícola para capturar os aspectos espaciais e temporais desse tipo de produção. Contudo, ainda que mais informativo do que o modelo básico, o modelo oculto ainda não era capaz de capturar as complexas relações existentes entre as variáveis do processo. O resultado, com uma quantidade relativamente grande de variáveis, era que as variáveis observáveis acabavam necessitando matrizes de probabilidade conjunta de todas as variáveis envolvidas. O cálculo dessas matrizes, apesar de trivial, é computacionalmente custoso.

Os aspecto temporal de uma colheita, em todas as suas fases, pode ser adequadamente capturada por uma série temporal, representada por um modelo de Markov. Mas Redes Bayesianas Dinâmicas (ou, simplesmente, RBD) também são um modelo Markoviano, com a vantagem de explicitarem as dependências causais existentes entre as variáveis do problema. Dessa forma, o trabalho teve seu método alterado para essa abordagem.

Nomeado em homenagem ao matemático inglês Thomas Bayes (1701 – 1761), o Teorema da Bayes descreve a relação existente entre as probabilidades *a priori* e *a posteriori* de dois eventos probabilisticamente dependentes A e B. O teorema é apresentado na definição 1:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

em que $P(A)$ a probabilidade (*a priori*) do evento A ocorrer, $P(B)$ a probabilidade (*a priori*) do evento B ocorrer, $P(A|B)$ a probabilidade do evento A ocorrer dado que o evento B ocorreu e $(B|A)$ a probabilidade de B ocorrer dada a ocorrência prévia de A (probabilidades *a posteriori*). Dessa forma, o Teorema de Bayes relaciona as probabilidades dos eventos A e B com suas probabilidades condicionais (BUSSAB; DE; MORETTIN, 2010).

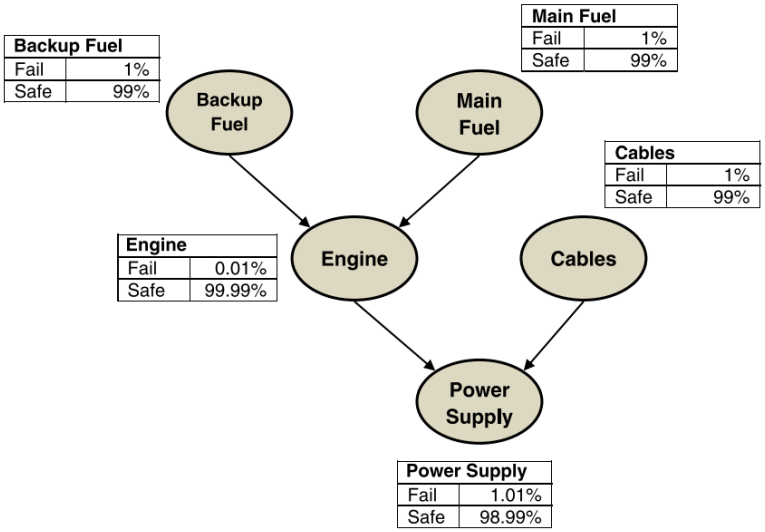
Uma rede Bayesiana, também chamada de rede de crenças Bayesianas (*Bayesian belief network*) é um modelo probabilístico que computa inferências por meio grafo dirigido acíclico (do Inglês *directed acyclic graph*, ou simplesmente DAG), onde os nós correspondem às variáveis do sistema e os arcos entre os nós representam dependências ou relações causais diretas entre variáveis.

As relações são computadas como probabilidades condicionais de variáveis discretas (NADERPOUR; LU; ZHANG, 2013; ROBERTON; LOBSEY; BENNETT, 2021). O grafo é necessariamente acíclico, visto que uma relação de causalidade usualmente é uma relação de ordem parcial (i.e., reflexiva, transitiva e antissimétrica). Na prática, contudo, causalidades podem ser cíclicas e faz parte do processo de construção da rede estabelecer quais são os arcos que serão eliminados. As Figuras 3 e 4 ilustram exemplos de redes Bayesianas, conforme apresentado abaixo.

A Figura 3 apresenta as dependências existentes entre os componentes e insumos de um carro responsáveis pelo seu movimento. No caso, a variável *Power Supply* (fonte de energia) é produzida tanto pelo correto funcionamento dos cabos (*Cables*) que entregam energia ao motor (*Engine*). O funcionamento do motor, por outro lado, também depende da existência de combustível nos tanques secundários e principal do veículo (*Backup Fuel* e *Main Fuel*). As probabilidades apresentadas na Figura 3 são as probabilidades marginais referentes a cada uma das variáveis. As distribuições de probabilidades condicionais $P(\textit{Engine}|\textit{Backup Fuel, Main Fuel})$ e $P(\textit{Power Supply}|\textit{Engine, Cables})$, expressas pelos arcos do grafo que representa a rede, não são apresentadas mas estão definidas. Note-se que o fluxo de informação da rede, na presença de evidências, pode fluir tanto em direção dos pais para os filhos (dedução) como na direção dos filhos para os pais (abdução).

A Figura 4 apresenta uma possibilidade de rede Bayesiana para modelagem de produtividade agrícola. Nessa figura, é representada a variável correspondente à produção resultante da colheita (RES), que é dependente das variáveis referentes à constituição bioquímica do solo (*soil*) teor de potássio (TK), teor de matéria orgânica (TMO) e teor de fósforo (TP); da declividade do terreno (DEC); das variáveis meteorológicas (*met*)

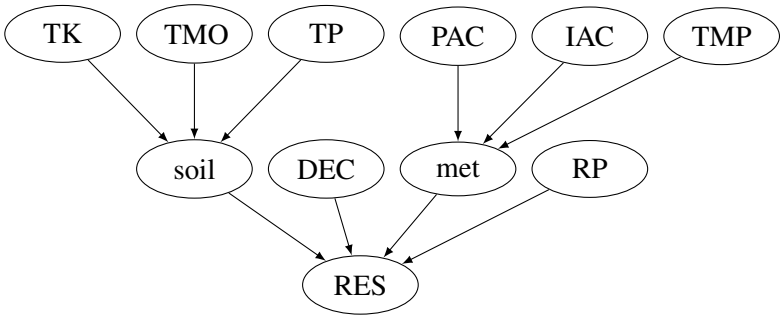
Figura 3 – Representação gráfica de uma Rede de Bayesiana



Fonte: Brooker (2011)

precipitação acumulada (PAC), insolação acumulada (IAC) e temperatura (TMP); e da resistência à penetração do solo (RP). Note-se que as variáveis *soil* e *met* agrupam variáveis relacionadas. Isso é feito para capturar em uma única variável características relacionadas e também para diminuir o grau dos nodos dos grafos, diminuindo a complexidade de treinamento e execução da rede.

Figura 4 – Exemplo Rede de Bayesiana - variáveis de escopo agrícola



Fonte: Autor (2022)

O aprendizado/treinamento de uma rede Bayesiana consiste em primeiro executar um algoritmo que aprenda a estrutura da rede, inferindo sua estrutura com base nas relações causais entre variáveis do conjunto de dados, representando tais relações como um grafo acíclico direcionado. Posteriormente, deve-se ajustar os parâmetros com base nas relações probabilísticas entre os nodos do grafo onde o valor do nó pai é usado para

calcular a densidade de probabilidade para o nó filho. Dessa forma, é possível inferir o valor de um nó filho pelos valores dos nós pais (CHAPMAN et al., 2018) (raciocínio dedutivo).

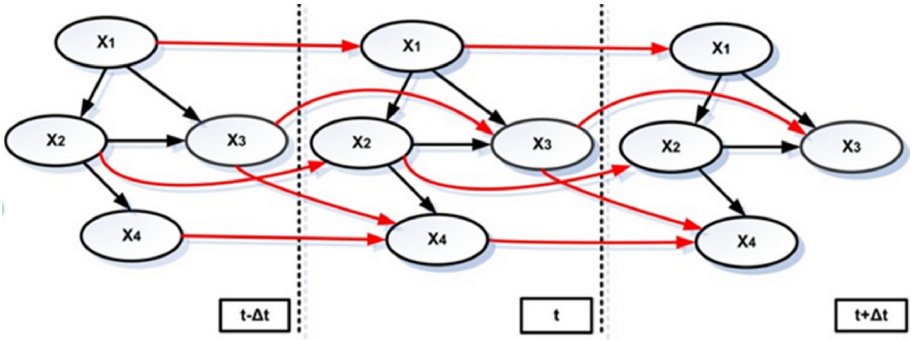
Graças ao Teorema de Bayes, o inverso também pode ser computado, permitindo também o raciocínio que tenta explicar as causas mais prováveis das consequências observadas (raciocínio abdutivo). Outra forma de treinamento consiste em definir sua estrutura e preencher as distribuições de probabilidade, usando uma abordagem frequentista, uma abordagem Bayesiana ou uma mistura das duas abordagens.

Uma restrição das redes Bayesianas é que não existe a possibilidade de retroalimentação de valores a partir das modificações que as interações do sistema geram. Essa característica é útil quando existe um atraso temporal na alteração das probabilidades *a posteriori*, que ocorre quando um valor de uma variável no passado influencia o valor de uma variável no presente. Essa característica estocástica é implementada pelas chamadas redes Bayesianas dinâmicas (*dynamic Bayesian networks*, ou simplesmente DBN).

Redes Bayesianas dinâmicas são processos markovianos de ordem superior a 1 (SHAMSHAD et al., 2005), organizadas como uma repetição de sucessivas redes Bayesianas estáticas idênticas, mas com acréscimo de arcos entre as variáveis das diferentes instâncias da rede que mostram como as variáveis de uma instância afetam as variáveis da próxima rede (PRICE; MOODLEY; PILLAY, 2018).

A Figura 5 apresenta um exemplo de uma representação gráfica de rede de Bayesiana dinâmica. Os arcos em preto ($X_1 \rightarrow X_2$, $X_1 \rightarrow X_3$, $X_2 \rightarrow X_3$ e $X_2 \rightarrow X_4$) são as dependências estáticas existentes entre as variáveis da rede; os arcos em vermelho são as dependências dinâmicas ou temporais, mostrando quais variáveis do passado afetam o valor das variáveis no futuro.

Figura 5 – Representação gráfica de uma Rede de Bayesiana Dinâmica



Fonte: Adaptado de Khakzad, Landucci e Reniers (2017)

A construção de redes Bayesianas existe que a dependência de dados probabilísticos e suas relações de dependência estejam bem definidas. Existem três métodos de construção de redes Bayesianas: manual, sendo projetada por um ou mais especialistas no domínio do problema; automática, a partir de dados de realidade, onde não há intervenção humana e a Rede Bayesiana é criada a partir dos dados de entrada; e a metodologia mista onde a rede é construída por uma combinação das duas metodologias (DRURY et al., 2017). Neste trabalho foi verificada a diferença de resultados entre o primeiro e o segundo métodos, visto que os dados foram gerados de forma controlada. O pacote construído, por outro lado, admite a inclusão de parâmetros que permitem o aprendizado da rede pela metodologia mista.

Redes Bayesianas dinâmicas permitem que as probabilidades condicionais e também a estrutura do modelo sejam atualizados conforme novas evidências são disponibilizadas, evitando-se assim a deterioração progressiva do modelo. As probabilidades condicionais podem ser refinadas para refletir as evidências disponíveis, tal refinamento pode ser produzido somente a partir dos dados novos ou com base em especificações do usuário (DAGUM; GALPER; HORVITZ, 1992). Depois do treinamento, contudo, não há possibilidade de alterar a estrutura da rede. Se houver necessidade, ela deve ser novamente produzida a partir dos novos dados.

A previsão é uma aplicação natural para Redes Bayesianas pela capacidade de fazer inferências com base em informações incompletas e absorver novas informações (DRURY et al., 2017). Além disso, a rede é capaz de expressar dependências entre causas e efeitos, lidar com dados parciais, integrar novos dados e prover nova interpretação com novos dados (ELAVARASAN et al., 2018).

3.2 Técnicas de Validação

Na próxima seção deste trabalho algumas técnicas utilizadas para a validação dos modelos aplicados à tarefa de previsão de resultado agrícolas são apresentados. Cada modelo, de acordo com o contexto de sua aplicação e a disponibilidade de dados, utiliza métricas distintas para validação dos resultado. Esta seção apresenta uma breve caracterização das métrica utilizadas. Nas Equações (2) a (8) y'_i é o valor previsto, y_i é o valor observado e n é o tamanho da amostra.

O erro médio absoluto (*mean absolute error*, ou MAE) é uma medida que considera a média do somatório do módulo das diferenças entre o valor observado e

o valor previsto. O MAE é calculado como a soma dos erros absolutos dividido pelo tamanho da amostra, conforme indicado na Equação (2):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y'_i - y_i| \tag{2}$$

O erro percentual médio (*Mean Percentage Error* - MPE) é a média calculada dos erros pelos quais as previsões de um modelo diferem dos valores reais, sendo calculado pela Equação (3):

$$MPE = \frac{1}{n}\sum_{i=1}^{n}\frac{y'_i - y_i}{y_i} \times 100 \tag{3}$$

O erro percentual médio absoluto (*Mean Absolute Percent Error* - MAPE) é a média dos erros percentuais absolutos das previsões onde os erros percentuais são somados em módulo. Quanto menor o MAPE, melhor a previsão e é calculado conforme a Equação (4).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y'_i - y_i}{y_i}| \tag{4}$$

O erro quadrático médio (*Mean Squared Error* - MSE) mede a quantidade de erro em modelos estatísticos. Ele é apurado pela diferença média quadrada entre os valores observados e previstos. Quando um modelo não tem erro, o MSE é igual a zero. Quanto menor o MSE mais efetivo é o modelo. A Equação (5) apresenta o cálculo desta métrica

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y'_i - y_i)^2 \tag{5}$$

A raiz do erro médio quadrático (*root mean square error* - RMSE), ou raiz do desvio médio quadrático (*Root Mean Squared Deviation* - RMSD), calcula a média do somatório das diferenças entre o observado e o previsto, entretanto. A raiz aplicada ao resultado devolve a métrica à escala dos valores observados, facilitando a sua compreensão. A Equação (6) apresenta descreve o cálculo desta métrica:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y'_i - y_i)^2} \tag{6}$$

Para a determinação das variáveis a ser utilizado em modelos de regressão aplica-se a métrica do Erro Quadrático Médio de Validação Cruzada (*Root Mean Square Error of Cross Validation* - RMSECV), esta métrica é calculada pela mesma equação da RMSE, com a diferença de ser aplicada em uma etapa anterior a definição do modelo,

permitindo comparar o efeito que cada variável tem no desempenho do modelo.

A raiz do erro percentual médio quadrático (*Root Mean Square Percentage Error* - RMSPE) é a representação percentual da RMSE e é obtida multiplicando-se Equação (6) por 100.

A raiz relativa do erro quadrático médio (*Relative Root Mean Squared Error* - RRMSE), diferentemente da RMSE, que é restrita pela escala das medições originais, o RRMSE pode ser usado para comparar diferentes técnicas de medição. RRMSE expressa o erro relativamente ou em forma de porcentagem, sendo que quanto menor o valor melhor é o modelo. O calculo desta métrica é feito conforme a Equação (7):

$$RMSE = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}(y'_i - y_i)^2}{\sum_{i=1}^{n}(y_i)^2}} \times 100 \tag{7}$$

O coeficiente de determinação (R^2) é uma medida que determina o quanto um modelo de regressão está ajustado aos valores observados de uma variável aleatória. Ele é obtido pela Equação (8)

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - y'_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{8}$$

onde $\bar{y}$ é a média das observações.

A acurácia é uma métrica calculada a partir da matriz de confusão de um modelo de classificação, levando em consideração a taxa de falsos negativos (FN), falsos positivos (FP), verdadeiros negativos (VN) e verdadeiros positivos (VP), conforme exposto na Equação (9):

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{9}$$

3.3 Trabalhos correlatos

Nesta seção são apresentados os trabalhos levantados na revisão de escopo da literatura, conforme método descrito na Seção 2.2, cujo conteúdo é afim ao trabalho aqui proposto. Primeiramente, apresentam-se os trabalhos cuja abordagem visa a previsão de produtividade agrícola independente da estratégia adotada, posteriormente são expostos os trabalhos ligados à agricultura, cujo método se alicerça em modelos probabilísticos, independentemente da aplicação ser ou não ligada a previsão de resultado agrícola.

3.3.1 Previsão de produtividade agrícola

Dos trabalhos estudados no levantamento bibliográfico realizado, dentre as justificativas para realização de estimativa de produção ou predição de resultado, a construção de uma representação confiável da situação agro-produtiva visando dirimir a insegurança alimentar e gerar insumos para planejamento e gestão tanto dos produtores como do poder público são as mais citadas.

Existem dois tipos de abordagem de predição de resultado de colheita: baseadas em processos, que focam em aspectos da fisiologia do crescimento das plantas, modelando o impacto da mudança destes processos na produtividade final. Por outro lado, tem-se as baseadas em dados oriundos de variáveis que se acredita relevantes para os resultados da colheita (HUNTINGTON et al., 2020). Neste trabalho, se dá foco à perspectiva baseada em dados. Entretanto, é possível citar os trabalho de Park et al. (2017) e Zaeen et al. (2020) que trabalham com previsão de produtividade a partir de características fisiológicas das plantas, comparando os dados coletados em campo com as métricas consideradas ideais em cada fase do crescimento da planta com objetivo de estimar o resultado da produção.

Sob esta ótica, pode-se verificar que alguns autores focam suas pesquisas em determinar quais as técnicas são consideradas mais eficazes, abordando os mesmos conjuntos de dados com diferentes técnicas. Shu (2020) comparou diferentes modelos de aprendizado de máquina com o desempenho de seu próprio modelo baseado em *Self Normalizing Neural Networks (SNN)* para a previsão de produtividade de soja no Japão. A comparação foi validada por meio da RMSE relativa onde o método *k-Nearest Neighbors* (KNN) foi o mais eficaz para o conjunto de dados menor (521 ocorrências) e o SNN para o conjunto maior (3148 ocorrências).

O trabalho de Srikamdee, Rimcharoen e Leelathakul (2018) equiparou três técnicas baseadas em redes neurais – *backpropagation neural network* (BPNN), *Adaptive Evolution Strategies* (A-ES) e *Deep Neural Network* (DNN) – para verificar a capacidade de predizer a produtividade da cana de açúcar em lavouras na Tailândia entre os anos de 2010 e 2014. Nesse trabalho a validação dos resultados foi feita pelo MAPE, tendo a abordagem de redes neurais com aprendizado profundo sido a mais precisa na médias de 4 anos.

Gaso, Berger e Ciganda (2019) compararam dois modelos, um baseado em regressão simples e um "método de modelo de lavoura (*crop model method*)" utilizando

dados de vegetação obtidos por sensoriamento remoto em 22 campos no sudoeste do Uruguai, tendo sido obtida uma RMSE relativa uma vez e meia (1,5) menor na abordagem com regressão linear. Apesar disso os autores destacam que a implementação do método de modelo de lavoura permite uma melhor representação espacial da variabilidade da produção.

Outros estudos adotam uma abordagem mais direta, focando nos resultados da aplicação de uma técnica a um conjunto de dados para a predição de um ou mais cultivares. A seguir serão apresentadas as impressões sobre estes trabalhos.

Com a ferramenta WEKA[2], Gandhi et al. (2016) aplicaram otimização mínima sequencial e Gandhi, Petkar e Armstrong (2016) utilizaram redes neurais artificiais (RNA) de multicamada em um conjunto de dados de 27 distritos do estado de Maharashtra na Índia, contendo dados de produção e meteorológicos, para a previsão da produtividade de arroz. O trabalho apresenta a validação de seus resultados através de um percentual de acurácia com base na matriz de confusão do modelo.

O trabalho de Huntington et al. (2020) aplicou modelo baseado em florestas aleatórias (*random forests*) para a predizer a produção em biomassa do sorgo nos Estados Unidos, para o período de 2018-2100, considerando um conjunto de cenários climáticos possíveis. A partir de dados climatológicos, de produção, de preço do *commodity* e de características de solo, o modelo foi treinado com dados de 1958 a 1987 e validado com dados de 1988 a 2016. Apesar de não informar a eficácia do modelo, o trabalho apresenta os resultados comparando a produtividade estimada com a registrada no período de validação.

Moraes et al. (2020) apresentam um modelo, de regressão linear múltipla e clusterização para predição de produtividade do Açaí na amazônia oriental, tanto para plantação irrigadas como não irrigadas. O modelo considera a influência dos dados meteorológicos ao longo das seis fases fenológicas da planta. Os resultados foram validados através do (R^2) e MAPE, sendo que a eficácia do modelo varia conforme a época do ano.

Em Helfer et al. (2019), um sistema computacional, que usa dados de sensores para medir a composição do solo, dados meteorológicos e técnica de regressão linear, é aplicado para a predição da produtividade do trigo na região de Santo Ângelo, Rio Grande do Sul, Brasil. Os resultados foram validados através do R^2, RMSE e RMSECV.

Haghverdi, Washington-Allen e Leib (2018) usam uma série temporal anual de

[2]https://www.cs.waikato.ac.nz/ml/weka/index.html

índices de plantação, obtidos por sensoriamento remoto, para caracterizar a fenologia do algodão. Estes dados são usados como entrada de uma rede neural artificial para a previsão da safra de fibra de algodão. O modelo foi aplicado a diferentes combinações do conjunto de dados, em diferentes datas, ao longo dos anos de 2013 e 2014, totalizando 61.200 modelos. O objetivo foi estabelecer relação dos índices de safra com dados de produção de fibra de algodão medidos em campo. A validação dos resultados é feita pelo MAE e R^2.

O trabalho de Ahmad et al. (2020) aplicou a técnica de aprendizado de máquina *Least Absolute Shrinkage and Selection Operator* (LASSO) para estimar a produtividade de milho no distrito de Faisalabad no Paquistão. O modelo foi aplicado para prever a produção de 10 anos (2007-2016) objetivando avaliar as variabilidades sazonais e interanuais em relação à temperatura e precipitação. Os resultados foram validados pelos parâmetros R^2 e RMSE.

Folberth et al. (2019) empregam dois algoritmos de aprendizado de máquina – intensificação de gradientes (*extreme gradient boosting*) e florestas aleatórias – em dados de saída de modelos globais de cultura, com o objetivo de gerar um modelo que realize predição em escala espacial menor. A validação do modelo é dada por R^2, RMSE e MAE.

O trabalho de Aparecido et al. (2017) aborda variáveis agrometeorológicas com técnica de regressão linear múltipla para a predição da produtividade do café na região do Cerrado de Minas Gerais, no Brasil. Neste trabalho foram testados todas as possibilidade de combinação de variáveis independentes, buscando identificar os melhores modelos para cada área investigada. O modelos foram classificados com base no R^2 ajustado e MAPE.

Zhang et al. (2017) desenvolveram um conjunto de 8 (oito) modelos de regressão *stepwise regression* para prever variações na produtividade do trigo de inverno em Oklahoma, nos Estados Unidos, com base nas condições de vegetação, umidade e temperatura. Estes modelos foram comparados entre si sob os parâmetros de RMSE e R^2.

Farooque et al. (2020) aplicou técnicas de regressão – linear e cúbica – para estimar a produtividade de batatas de maneira não intrusiva, mediante a relação entre produtividade e condutividade elétrica do solo em New Brunswick e ilha Prince Edward no leste do Canadá. A pesquisa concluiu, com base no RMSE e R^2, que existe forte correlação entre condutividade elétrica aparente do solo e a produtividade e que o modelo de regressão cúbica se mostrou mais acurado que o de regressão linear.

A definição de um conjunto de dados relevantes para a realização de predição, bem como as técnicas para obtenção desses dados também representam um eixo de debate em relação aos trabalhos analisados. O trabalho de Holzman et al. (2018) avalia a utilização de dados de estresse hídrico, obtido por sensoriamento remoto, e de índice de radiação solar para a predição de milho, soja e trigo na Argentina. Chen et al. (2020) desenvolveram um sistema para estimar os rendimentos de trigo, cevada e canola ao longo do cinturão de trigo na Austrália utilizando dados de sensoriamento remoto com índices de estresse derivados de dados meteorológicos.

O trabalho de Peng et al. (2018) aplicou um modelo baseado em regressão a um conjunto de dados climáticos e outro conjunto de dados climáticos associados com dados de sensoriamento remoto, na predição da produtividade de milho nos Estados Unidos. O estudo concluiu que a inclusão do índice de vegetação aprimorado (*Enhanced Vegetation Index* - EVI) resultou em uma melhora no desempenho de previsão de rendimento.

O trabalho de Mladenova et al. (2017) comparou o resultados de previsão de rendimento de 12 conjuntos de dados em larga escala que empregam umidade do solo, evapotranspiração e/ou dados relacionados à vegetação, obtendo melhores resultados com a associação destes dados quando comparado com modelos exclusivamente baseados em vegetação para a previsão de rendimentos de milho e soja nos Estados Unidos.

Holzman e Rivas (2016) avaliaram a aptidão do índice de temperatura seca de vegetação (*Temperature Vegetation Dryness Index* - TVDI) obtidos por técnicas de sensoriamento remoto, em técnicas de regressão linear e polinomial para predizer a produtividade de milho na região do Pampa argentino, obtendo razoável precisão de oito a 12 semanas antes da colheita em regiões úmidas e uma menor precisão em áreas do semiárido.

Byakatonda et al. (2018) investigaram a relação entre os índices climáticos como índice de aridez (*Aridity Index* - AI), índice padronizado de evapotranspiração de precipitação (*Standardised Precipitation Evapotranspiration Index* - SPEI), índice de oscilação sul (*Southern Oscillation Index* - SOI) e duração da estação chuvosa (*Length of the Rainy Season* - LRS) nas safras de milho e sorgo no semiárido Botsuana. Tal relação foi aplicada em modelos de Rede Neural Artificial que previu probabilidade da redução do rendimento destas cultivares nos próximos 5 anos.

O trabalhos de Jha et al. (2019) avaliaram a aplicabilidade da utilização de dados meteorológicos em escala diária, estimados a partir de medidas mensais ou sazonais, em modelo de predição de produtividade de arroz no Nepal. Não houve ganho de eficiência

notável neste experimento.

O trabalho de Wang et al. (2018) utilizou árvores de decisão (*Classification and Regression Tree* - CART) para determinar o poder explicativo das propriedades de solo na variabilidade da produtividade de arroz no nordeste da China, concluindo que há potencial para o aumento da eficiência na predição da produtividade quando se agregam dados das propriedades do solo por aumentar o poder explicativo pela identificação do fator limitante mais crítico.

Khanal et al. (2018) examinaram o papel das imagens de sensoriamento remoto para predizer características de solo e o rendimento de milho em uma área no estado de Ohio nos Estados Unios. Comparando o desempenho de modelos usando regressão linear e cinco modelos de aprendizado de máquina obteve resultados que indicam que a associação de dados de sensoriamento remoto com técnicas de aprendizado de máquina são promissores para estimar produtividade de lavoura de milho em nível local. O trabalho de Leroux et al. (2019) Chegou a conclusões semelhantes quando avaliou a produção de milho em Burkina Faso usando, também, técnicas de aprendizado de máquina e dados de sensoriamento remoto.

O trabalho de Guan et al. (2017) também que utiliza dados de sensoriamento remoto, buscando associar dados de múltiplos satélites com informações de diversas faixa espectrais, do visível ao infravermelho, da região do cinturão do milho nos Estado Unidos, indicando que a utilização de dados de satélite melhora sensivelmente previsões de rendimento das safras, quando comparado ao que pode ser obtido com sensores individuais.

Já Kamir, Waldner e Hochman (2020) lançaram mão de técnicas de aprendizado de máquina para mapear a produtividade real das safras de trigo na Austrália a partir de séries temporais de imagens de satélite e clima, permitindo identificar desvios e lacunas na produtividade.

No trabalho de White et al. (2020) o efeito da introdução de informações relativas à umidade do solo, obtidas por sensoriamento remoto, é avaliado em relação a eficácia da predição da produtividade de canola nas pradarias canadenses. O estudo concluiu que houve aumento pela adição destes dados, na eficácia da predição em mais da metade dos eco-distritos estudados.

A agricultura é sensível a mudanças climáticas, especialmente a variabilidade na temperatura média e nos regimes de chuvas, tais varições podem causar alterações substanciais nos sistemas produtivos agrícolas (IIZUMI; RAMANKUTTY, 2016).

Desta feita, alguns trabalhos (LEHMANN et al., 2020; Uwizera; McSharry, 2017; NGUYEN-HUY et al., 2018; AHMAD et al., 2020; FAROOQUE et al., 2020) buscam correlacionar as alterações climáticas com eventuais variabilidades na produtividade ou então prever os efeitos a médio e longo prazo dessas mudanças (FATHI et al., 2017).

O trabalho de van Klompenburg, Kassahun e Catal (2020) não foi identificado no levantamento detalhado no capítulo anterior, mas foi incluído nesta seção por ser uma revisão bibliográfica com enfoque semelhante ao que se pretende adotar neste trabalho e não um trabalho primário. Objetivando identificar publicações que empregam aprendizado de máquina para a predição de resultado de colheita. A citada revisão identificou 50 publicações que atenderam os critérios estabelecidos. Dentre as publicações identificadas, redes neurais é a técnica mais usada, com 27 trabalhos, seguida por regressão linear, com quatorze, florestas aleatórias, doze, *support vector machines*, dez e *extreme gradient boosting* com quatro. Detre as técnicas de redes neurais as mais empregadas são *convolutional neural networks*, l*ong-short term memory* e, *deep neural networks*. No que se refere à escolha de variáveis independentes, as mais empregadas foram temperatura, tipo de solo, precipitação e informações da produção.

Comparando os levantamentos realizados por van Klompenburg, Kassahun e Catal (2020) com o realizado neste trabalho, encontramos apenas três publicações que figuram em ambos. Dentre os aspectos que justificam esta divergência apontamos para o hiato temporal definido, sendo de cinco anos no presente e dez no outra revisão. Além disso, foram investigados dois repositórios a mais do que nesta revisão, totalizando seis, sendo que apenas dois repositórios constam em ambos. Por fim, ainda é possível verificar que nos termos de busca daquela revisão fazem uso de menos conjunções, o que restringe menos a busca.

De maneira simplificada, as técnicas utilizadas nas publicações estudadas, seja com aplicação finalística ou como técnica auxiliar para organização dos dados ou escolha das variáveis independentes, podem ser agrupados de acordo com a frequência de utilização de uma ou mais classes de métodos. Foram identificados treze trabalhos que aplicam técnicas de regressão, apresentados na Tabela 3, quinze trabalhos que empregam técnicas de aprendizado de máquina elencados na Tabela 4 e quatro que utilizam técnicas que não puderam ser classificadas em nenhuma das anteriores, sendo estando descritas na Tabela 5.

As tabelas apresentam estrutura similar, indicando a referência para do trabalho correlato, a técnica aplicada, informação sobre a descrição ou não da variabilidade

espacial, ou seja, se o trabalho apresenta informações de se existe e como se dá a variação produtiva dentro de da área de estudo, informação sobre a descrição ou não da variabilidade temporal, ou seja, se o trabalho apresenta reflexão quanto avariabilidade intra e inter safras e qual foi o método de validação aplicado.

Tabela 3 – Resumo trabalhos correlatos - Modelos de regressão

Modelos de regressão				
Trabalho	**Técnica**	**Variabilidade espacial**	**Variabilidade temporal**	**Validação**
Moraes et al. (2020)	Regressão linear múltipla e Clusterização	NÃO	SIM	MAPE e R^2
Helfer et al. (2019)	Regressão linear	NÃO	SIM	R^2, RMSE e RMSECV
Gaso, Berger e Ciganda (2019)	Regressão linear	SIM	NÃO	RMSE
Aparecido et al. (2017)	Regressão linear múltipla	SIM	NÃO	R^2 e MAPE
Zhang et al. (2017)	regressão múltipla	SIM	SIM	R^2 e RMSE
Kadigi et al. (2020)	Distribuição empírica multivariada	NÃO	NÃO	Não apresentou
Holzman et al. (2018)	Regressão linear e quadrática	SIM	NÃO	RMSE
Lehmann et al. (2020)	Regressão linear	NÃO	NÃO	R^2
Park et al. (2017)	Regressão polinomial	NÃO	NÃO	RMSE
Peng et al. (2018)	Regressão não linear de múltiplas variáveis	SIM	NÃO	RMSE

Tabela 3 – continuação

Trabalho	Técnica	Variabilidade espacial	Variabilidade temporal	Validação
Zaeen et al. (2020)	Regressão não linear e regressão exponencial	NÃO	NÃO	RMSE
Holzman e Rivas (2016)	Regressão não linear e regressão quadrática	SIM	NÃO	RMSE
Farooque et al. (2020)	Regressão não linear e regressão cúbica	SIM	NÃO	R^2

Fonte: Autor (2021)

Tabela 4 – Resumo trabalhos correlatos - Modelos de Aprendizado de máquina

Modelos de Aprendizado de máquina				
Trabalho	**Técnica**	**Variabilidade espacial**	**Variabilidade temporal**	**Validação**
Shu (2020)	SNN	NÃO	NÃO	RRMSE
Srikamdee, Rimcharoen e Leelathakul (2018)	BPNN, A-ES e DNN	NÃO	NÃO	MAE percentual
Huntington et al. (2020)	RF e *extremely randomized trees regression*	SIM	SIM	Comparativo
Gandhi, Petkar e Armstrong (2016)	RNA	NÃO	NÃO	Matriz de confusão
Ahmad et al. (2020)	LASSO	SIM	NÃO	Acurácia, R^2 e RMSE
Folberth et al. (2019)	*extreme gradient boosting*) e RF	SIM	NÃO	R^2, RMSE e MAE
Nguyen-Huy et al. (2018)	*D-Vine copula*	NÃO	SIM	MAE e RRMSE

Tabela 4 – continuação

Trabalho	Técnica	Variabilidade espacial	Variabilidade temporal	Validação
Byakatonda et al. (2018)	RNA	SIM	SIM	MSE
Khanal et al. (2018)	RF, NN, SVM, GBM e CUB	SIM	NÃO	R^2 e RMSE
Leroux et al. (2019)	RF	NÃO	NÃO	R^2, RMSE, RRMSE e MAE
Kamir, Waldner e Hochman (2020)	RF, CUB, SVM, KNN, MARS, GPR, MLP, XgBoost	NÃO	NÃO	R^2 e RMSPE
Guan et al. (2017)	PLSR	NÃO	NÃO	R^2 e RMSE
Gandhi et al. (2016)	SVM	NÃO	NÃO	Matriz de confusão
Wang et al. (2018)	SVM	NÃO	NÃO	PRE
Wang et al. (2018)	SVM, RF *Decision Tree, Naive-bayes* e *Logistic Regression*	NÃO	NÃO	Não descrito

Fonte: Autor (2021)

Tabela 5 – Resumo trabalhos correlatos - outros métodos

Trabalho	Técnica	Variabilidade espacial	Variabilidade temporal	Validação
White et al. (2020)	Modelo *"Canadian Crop Yield Forecaster" (CCYF)*	SIM	NÃO	R^2
Chen et al. (2020)	Método semi-empírico	NÃO	NÃO	RMSE
Jha et al. (2019)	Modelo *"Decision Support System for Agrotechnology Transfer" (DSSAT)*	SIM	NÃO	MPE e RMSD
Schauberger, Gornott e Wechsung (2017)	Método semi-empírico	SIM	NÃO	R^2

Fonte: Autor (2021)

3.3.2 Aplicação de modelos probabilísticos na agricultura

A revisão bibliográfica, detalhada na Seção 2.2, também teve como objetivo levantar trabalhos que utilizam modelos probabilísticos aplicados à agricultura, não necessariamente restritos à previsão ou estimativa de produtividade. Foram encontrados 13 trabalhos com utilização de modelos de Markov em atividades agrícolas diversas, onde a identificação de padrões é a aplicação com maior recorrência encontradas. Além de 11 trabalhos utilizando redes Bayesianas, sendo modelagem para tomada de decisão a aplicação com maior recorrência, com cinco trabalhos, previsão de resultado de produção o segundo mais frequente, com 4 trabalhos e dois trabalhos focados em levantamento bibliográfico.

O sistema desenvolvido por Yashaswini et al. (2017) é composto por uma rede de sensores e atuadores aplicados na lavoura, objetivando implementar um sistema automatizado de irrigação. Utiliza HMM para identificar precocemente o surgimento

de doenças em plantas, a partir do conjunto de dados coletados. A inferência para definir a provável doença é feita considerando a existência das condições favoráveis para seu surgimento. Abordagem semelhante é usada em Patil e Thorat (2016) e Pawara et al. (2018), onde HMM é usado para identificar a existência de condições propícias para o surgimento de doenças em uvas e em romãs respectivamente.

O trabalho desenvolvido por Popli, Jha e Jain (2021) se refere a um tema ligado à agricultura de forma subsidiária. Objetivando reduzir custos de produção por meio da redução do consumo energético, o sistema emprega HMM para a treinar um sistema de gerenciamento de acesso à rede para transmissão de dados coletados em lavoura. Abordagem semelhante é dada em Li et al. (2020), quando utiliza HMM para otimização de sistema de monitoramento buscando reduzir o número de medições, sem perda na qualidade das classificações, em um sistema de apoia à agricultura de precisão.

Arfa et al. (2015) usa uma cadeia de Markov não estacionária para identificar o impacto de politicas públicas que afetam o preço do leite, no crescimento ou redução do numero de fazendas leiteiras na França. O modelo verificou o peso das variáveis preço do leite de vaca, rendimento de leite de vaca, existência de pagamento direto, reserva de cota de leite e pagamento de incentivo para estabelecimento de novos produtores na relação, tamanho e quantidade de fazendas.

Huang, Sklar e Parsons (2020) usam um modelo oculto de Markov combinado com um Processo de Dirichlet Hierárquico (HDP-HMM) para agrupar padrões de movimentos usados para aprendizado por demonstração, com objetivo de treinar um robô para a colheita de morangos. Modelos hierárquicos são utilizados para agrupamento de conjunto de dados com distribuição discreta. Segundo Fox et al. (2008), HDP-HMM permite desenvolver algoritmos de aprendizagem orientados a dados que outorgam inferir sobre posteriores distribuições sobre o número de estados.

O trabalho de Ghamghami et al. (2020) aplicou um método empírico de Bayes que inclui a aplicação de um HMM que auxilia na estimativa do porcentagem de progresso da cultura (CPP ou *crop progress percentage*), índice que indica o status do progresso da plantação ao longo das fases fenológicas. As sete fases fenológicas do milho são usadas como camada oculta e 2 índices de vegetação são usados como camadas visíveis do HMM aplicado neste trabalho. Os índices de vegetação aplicados são o NDVI e Grau-Dias de Crescimento Acumulado (*Accumulated Growing Degree Days* - AGDD). NDVI é obtido por meio das resposta espectrais nas imagens obtidas pelos satélites LANDSAT[3] 7 e 8 e o

[3]https://landsat.gsfc.nasa.gov/

AGDD é calculado com base nas temperaturas médias medidas por estação meteorológica considerando as temperaturas cardinais das fases fenológicas. O HMM aplicado neste trabalho é do tipo não homogêneo, pois assim que a plantação entra em uma nova fase fenológica a probabilidade de ele retornar para um estágio anterior vai decaindo até chegar a zero. Em relação aos resultados obtidos, o trabalho comparou a RMSE do modelo proposto com um modelo clássico de HMM obtendo resultados melhores. Segundo Ghamghami et al. (2020), a modelagem de CPP pode auxiliar a avaliar as variações de produtividade da lavoura.

Em Mueller-Warrant et al. (2017) imagens de satélite de ocupação de solo foram convertidas em histórico de plantação, buscando verificar se a rotação de cultivo anual atende a um padrão identificável, usando um modelos de independência de Markov de etapa única ou acompanha restrições na escolha da cultura determinada pelos produtores. O trabalho conclui que os produtores não realizam a seleção da rotação de produção randomicamente, sendo deliberadas com até cinco anos de antecedência.

Kang e Özdoğan (2019) utiliza Cadeia de Markov com simulação de Monte Carlo para calibração dos melhores parâmetros de entrada para um algoritmo de predição de produtividade a partir da assimilação de dados de sensoriamento remoto.

Ferreira (2019) usa o problema de decodificação, cuja solução ótima é dada pelo algorítimo de Viterbi (VITERBI, 1967), para o problema de previsão do resultado de produção agrícola. O autor desenvolveu um modelo baseado em HMM a partir de pacotes na linguagem de programação R utilizando dados meteorológicos, de solo e de produção de quatro safras de soja (*Glycine max*) nos anos 2012/2013, 2014/2015, 2016/2017 e 2017/2018, em um talhão da EMBRAPA Pecuária Sul localizado no município de Hulha Negra, estado do Rio Grande do Sul, Brasil, com área de 13,4 hectares e altitudes que variam entre 230 e 250 metros. O modelo tratou os resultado da produção como estados ocultos do modelo, sendo definidas três classes: baixa produção, média produção e alta produção. O modelo mostrou-se adequado para predizer a produtividade ao longo das safras com base nos dados de produção mais recente e das variáveis observáveis. Conclui indicando que sejam realizados trabalhos futuros para que se detalhe espacial e temporalmente os dados buscando o aumento da eficácia do modelo desenvolvido.

Outra abordagem probabilística aplicada à problemas relacionados à agricultura são as redes Bayesianas e as redes Bayesianas dinâmicas. A seguir são apresentados alguns trabalhos que lançaram mão destas estruturas.

Price, Moodley e Pillay (2018) desenvolveram, a partir de uma abordagem híbrida,

considerando a expertise dos produtores e dados meteorológicos, redes Bayesianas dinâmicas capaz de modelar o processo de tomada de decisões por produtores de cana de açúcar quanto a aplicação ou não da técnica de queimada pré colheita.

Também através de uma abordagem híbrida Liman Harou et al. (2021) incorporam modelos de redes Bayesianas e Monte Carlo para desenvolver um modelo de cultura levando em consideração informações qualitativas, descritas pelas redes Bayesianas e quantitativas, descritas por modelo de Monte Carlo.

No trabalho desenvolvido por Gouthami e Balaji (2017) redes Bayesianas foram usadas para estimar o resultados de plantações de arroz no distrito de Karimnagar na Índia. O trabalho utilizou WEKA para desenvolver as redes com dados de precipitação, temperatura, evaporação, tamanho da área plantada, produção total e rendimento, tendo avaliado o desempenho através de matriz de confusão. Outro trabalho semelhante foi desenvolvido por Gandhi, Armstrong e Petkar (2016), onde os autores compararam o desempenho dos algoritmos BayesNet e NaiveBayes do Bayes submenu no WEKA para previsão da produtividade de arroz para o distrito de Maharashtra, também na Índia.

Chandraprabha e Dhanaraj (2021) também utilizaram a ferramenta WEKA com os algorítimos KNN, Bayes Net, Naive Bayes e RF para analisar e prever a viabilidade do solo para agricultura a partir do conjunto de dados que incluem tipo de solo, PH do solo, disponibilidade de NPK e dados de produção. O trabalho destacou como mais eficiente KNN e RF em comparação com os algorítimos Bayesianos.

Roberton, Lobsey e Bennett (2021) desenvolveram, a partir da a metodologia híbrida, usando um modelo linear ajustado aos dados em conjunto com a opinião de especialistas, uma BBN para avaliar o risco de compactação do solo e os efeitos em relação ao rendimento das culturas nas regiões de Nova Gales do Sul e Queensland, na Austrália. O modelo foi desenvolvido no software Netica[4].

Cornet et al. (2016) fizeram uso de Modelagem de rede Bayesiana aditiva, desenvolvidas no pacote R abn[5], para revelar os fatores causadores de variabilidade produtiva inter-plantas de Inhame em plantações na África Ocidental, tendo observado uma contribuição direta do número de catafilos e da data de emergência para o peso dos tubérculos.

Kocian et al. (2020) utilizaram DBN para criar um sistema de apoio à decisão agrícola baseado na IoT, relacionando indicadores do desenvolvimento da cultura aos parâmetros de controle ambiental em estufas.

[4]https://norsys.com/netica.html
[5]https://cran.r-project.org/web/packages/abn/index.html

Chapman et al. (2018) usaram redes Bayesianas para prever o rendimento de plantações de dendezeiros, usando dados ambientais e dados de gerenciamento do plantio, tendo sido equivalente ou mais eficiente de RNAs. A rede foi desenvolvida a partir do método baseado em dados usando o pacote R bnlearn[6] aplicando o algoritmo heurístico *hill-climbing* para aprender a estrutura da rede. Em relação à RNA os resultados da rede Bayesiana desenvolvida foram equivalentes ou melhores.

Gandhi e Armstrong (2016) realizaram levantamentos bibliográfico da aplicação de técnicas de mineração de dados para tomada de decisão na agricultura e identificaram a utilização de redes Bayesianas em aplicações como predição do efeito de uso de fungicidas à produtividade do trigo, parâmetros genéticos de facilidade de parto em gado holandês, desenvolvimento de sistemas de apoio a tomada de decisão para o cultivo de malte sem uso de pesticidas, decisões associadas à seleção de sistemas de irrigação para fazendas leiteiras, controle de doenças, controle de proliferação de ervas daninhas entre outras.

O trabalho de Drury et al. (2017) complementa o levantamento apresentado nesta seção, visto que realizou um levantamento dos trabalhos existentes na literatura no período de 1984 a 2016, referente a aplicação de redes Bayesianas na agricultura. O levantamento analisou publicações em periódicos com revisão por pares e teses de doutorado distribuídos principalmente nas áreas de monitoramento automatizado, predição, identificação de causalidade, classificação e sistemas de suporte a decisão, classificando-os por método de construção, estrutura de aprendizado e método de avaliação das redes Bayesianas. Os autores concluíram que, apesar de serem adequadas, o uso de Redes Bayesianas na agricultura é uma estratégia raramente selecionada, apontando como provável principal motivo a curva de aprendizado mais íngreme do que a das técnicas de aprendizado de máquina mais tradicionais.

[6]https://cran.r-project.org/web/packages/bnlearn/index.html

4 O MODELO AGROBAYES

4.1 Organização e funcionalidades

O modelo apresentado neste trabalho, denominado AGROBAYES, faz uso tanto de redes Bayesianas estáticas quanto de redes Bayesianas dinâmicas para construir modelos capazes de prever o resultado da produção agrícola em diversos cenários de disponibilidade de dados. No pacote de software desenvolvido, o usuário pode apresentar um conjunto de dados organizados por áreas de produção, a partir do qual as redes são geradas, possibilitando visualizar as previsões de produção, a representação gráfica das redes e as métricas de desempenho dos modelos.

As áreas de produção são usadas para caracterizar a diversidade espacial existente em diferentes locais de uma mesma área. A diversidade de produção é causada pela heterogeneidade do solo, declive, proximidade de aquíferos, utilização ou não de irrigação, e outras variáveis que influem na produção, ainda que todas as áreas recebam as mesmas ações de manejo. Além da diversidade espacial, o tempo também é um fator significativo no processo produtivo, pois diferentes valores de uma mesma variável em tempo diferentes podem influenciar também de forma diferente o resultado final. Por exemplo, pouca chuva no início da germinação da planta afeta negativamente a produção de forma mais drástica que pouca chuva no período em que a planta já é adulta.

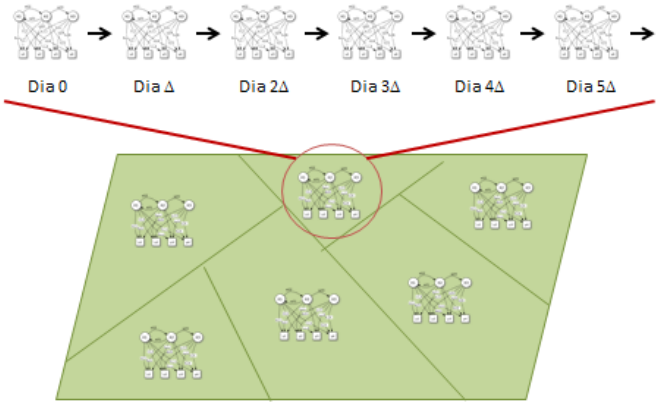
A Figura 6 ilustra essa ideia: pode-se identificar uma unidade produtiva, cujas subdivisões representam áreas em que a produção ocorre mais uniformemente em comparação com as áreas vizinhas. Com base nos dados gerados em cada fase fenológica das plantas são construídas redes estáticas, representadas nas figura pelas legendas dia 0, dia Δ, dia 2Δ ... 5Δ (assumindo aqui cinco fases fenológicas de relevância) e a interligação destas redes estáticas permite a criação de redes dinâmicas.

A construção do pacote AGROBAYES foi executada com o ferramental descrito na Seção 2.3. Adicionalmente, teve-se o cuidado de cumprir os requisitos de documentação das normas[1] necessárias para submissão de pacotes ao repositório padrão da linguagem R (https://cran.r-project.org/). O pacote completo está disponível no endereço https://github.com/GuiHalal/AgroBayes. O código completo do pacote pode ser encontrado no Apêndice C e a sua documentação, no formato do CRAN, encontra-se no Apêndice D.

O pacote AGROBAYES permite ao usuário gerar redes estáticas e dinâmicas a

[1] https://cran.r-project.org/web/packages/policies.html

Figura 6 – Caracterização espaço-temporal de uma área produtiva



Fonte: Autor (2021)

partir de um conjunto de *dataframes* apresentados em uma lista, em que cada elemento da lista representa uma área da plantação e cada *dataframe* representa os dados de uma fase fenológica da cultura. Os *dataframes* devem ser organizados de forma que cada linha contenha os dados de uma colheita sendo dispostas nas primeiras colunas as variáveis independentes e a variável dependente (resultado da produção), na última coluna.

A função *RunNetworks* é responsável pela criação das redes estáticas. O usuário deve apresentar os dados organizados em um conjunto finito de classes de uma área e duas matrizes contendo o conjunto de arcos que devem e não devem constar nas redes. A função retorna as métricas de desempenho das redes construídas na tarefa de previsão do resultado. Cada vez que a função é executada, são geradas quatro redes para cada fase fenológica da área testada. Os algoritmos utilizados no aprendizado de estrutura das redes são *Hill-Climbing* (HC) e *Max-Min Hill-Climbing* (MMHC) com e sem a predeterminação do conjunto de arcos que compõem a rede.

A função *createDbn* executa a criação das redes dinâmicas. O usuário deve apresentar os dados contínuos de uma área sendo retornadas as métricas (MAE, RMSE e acurácia) de desempenho das redes na previsão da produção. Duas redes são geradas para cada área, sendo o aprendizado da estrutura das redes realizado pelos métodos *dmmhc*[2] e *natPsoho*[3]. Cada vez que a função é executada, as redes são geradas e realizam a previsão da produção nos modos aproximado e exato, gerando portanto o comparativo de desempenho de quatro cenários. Os dados de entrada, nesse caso, não precisam ser

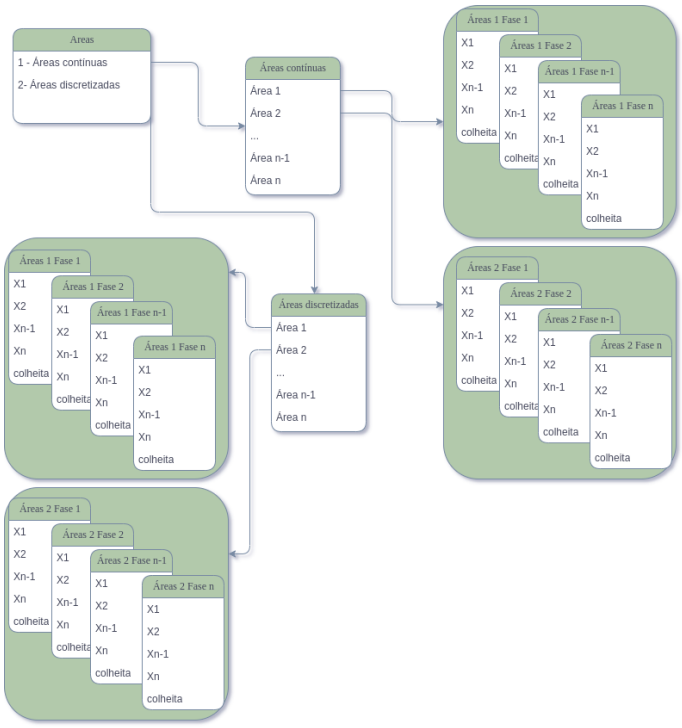
[2] *dynamic max-min hill climbing* (TRABELSI, 2013)
[3] *scalable particle swarm optimization algorithm for higher-order DBN* (QUESADA; BIELZA; LARRAÑAGA, 2021)

classificados, pois as redes dinâmicas trabalham com valores contínuos.

Com intuito de validar os modelos desenvolvidos no pacote, algumas funções de testes foram implementadas e também estão disponíveis aos usuários do pacote, permitindo a experimentação de várias estruturas de redes. A função *testRunDataGen* permite ao usuário gerar uma lista de tamanho dois, onde no primeiro índice estão disponibilizados os dados contínuos e no segundo os mesmos dados classificados. Cada um destes conjuntos de dados está estruturado como uma lista de *dataframes*, cada índice destas listas representa uma área da plantação e cada *dataframe* contém dados de uma fase fenológica. A função recebe como parâmetros o número de colheitas, o número de fases, o número de áreas, o número de variáveis e a quantidade de classes para a classificação dos dados.

A Figura 7 representa uma saída da função. Podemos identificar a lista (Áreas) com duas instâncias (1 - Áreas contínuas e 2 - Áreas discretizadas), cada índice da lista possui uma lista com um conjunto de *dataframes* (Área 1 Fase 1, Área 1 Fase 2, ..., Área 2 Fase 1, Área 2 Fase 2, ...).

Figura 7 – Saída *testRunDataGen*



Fonte: Autor (2023)

4.2 Geração de dados aleatórios

O pacote construído não pôde fazer uso de dados reais. Os dados disponíveis, que haviam sido usados em (FERREIRA; FERREIRA; PEREZ, 2020), constituíam dados de somente quatro (4) colheitas e somente com os valores finais de produção. Não havia possibilidade de treinar as redes correspondendo às diversas fases fenológicas. A área em questão, pertencente à EMBRAPA Pecuária Sul deixou de ser usada para plantar soja, sem perspectiva de novos dados. Além disso, ainda que houvesse a disponibilidade de dados em abundância, o parecer de um especialista seria necessário para o conhecimento da dependência entre as variáveis e a produção, limitando o uso de dados reais. Dessa forma, optou-se por usar dados gerados aleatoriamente, de forma controlada, para que as redes construídas pudessem ser analisadas.

As funções usadas para a geração de dados também estão disponíveis no pacote e podem ser usadas pelo usuário para testes, se desejado. A geração de dados permite definir variáveis e áreas de produção e também estipular como o resultado será produzido a partir dos dados gerados. Note-se que esse relacionamento precisa existir para que a produção seja função dos valores das variáveis de cada área e de cada fase fenológica. Note-se que cada área pode ter variáveis diferentes, dependendo dos dados de realidade disponíveis pois, na prática, cada área terá uma rede dinâmica para si, em que cada fase fenológica será representada por um nível da rede.

A partir dos parâmetros indicados pelo usuário, a função *testRunDataGen* gera conjuntos de dados cujas relações entre as variáveis independentes e a variável dependente são preestabelecidas. As regras que determinam as relação entre as variáveis formam um conjunto com sete tipos de áreas. No cenário padrão, em que três variáveis independentes são inicializadas, obtendo-se as áreas:

- A_1, onde o peso da produção varia linearmente com os valores das três variáveis nas duas primeiras fases fenológicas, conforme descrito na Equação (10):

$$P = X_{11} + X_{12} + X_{21} + X_{22} + X_{31} + X_{32} \tag{10}$$

  aqui e no que se segue, X_{ij} representa o valor da variável i na fase fenológica j.

- A_2, que varia com o quadrado de variável X_1, conforme descrito na Equação (11):

$$P = X_{11}^2 + X_{12}^2 + X_{13}^2 \tag{11}$$

- A_3, cujo peso de produção varia com o quadrado de X_3, conforme descrito na Equação (12):

$$P = X_{31}^2 + X_{32}^2 + X_{33}^2 \tag{12}$$

- A_4, onde a produção é inversamente proporcional à soma de X_1 com X_3, conforme descrito na Equação (13):

$$P = \frac{1}{(X_{11} + X_{31})} + \frac{1}{(X_{12} + X_{32})} + \frac{1}{(X_{13} + X_{33})} \tag{13}$$

- A_5, que decresce com uma ponderação dos valores de X_2, conforme descrito na Equação (14):

$$P = 1 \cdot X_{21} + 0,8 \cdot X_{22} + 0,6 \cdot X_{23} + 0,4 \cdot X_{24} + 0,2 \cdot X_{25} \tag{14}$$

- A_6, em que produção cresce com uma ponderação dos valores de X_1, conforme descrito na Equação (15):

$$P = 0,2 \cdot X_{11} + 0,4 \cdot X_{12} + 0,6 \cdot X_{13} + 0,8 \cdot X_{14} + 1 \cdot X_{15} \tag{15}$$

- Na A_7, em que produção depende de todas as variáveis de forma diferente em cada fase da produção. Conforme escrito na Equação (16):

$$\begin{aligned} P = {} & 50 \cdot X_{11} + 30 \cdot X_{21} + X_{31} + 40 \cdot X_{12} + 20 \cdot X_{22} + X_{32} + 30 \cdot X_{13} + 30 \cdot X_{23} + 5 \cdot X_{33} \\ & + 20 \cdot X_{14} + 40 \cdot X_{24} + 10 \cdot X_{34} + 10 \cdot X_{15} + 50 \cdot X_{25} + 20 \cdot X_{35} \end{aligned} \tag{16}$$

Nas Equações (10) a (16), P representa o valor da produção final, $X_1, X_2, \ldots, X_n$ identificam as variáveis e o segundo subíndice $1, 2, \ldots, n$ identificam a qual fase fenológica a variável X pertence.

Variáveis com diversos tipos de comportamento e toda a sorte de relações poderiam ser representadas no processo de geração de dados. Os exemplos de relações feitos acima têm a intenção de produzir os testes das redes construídas, para verificar até que ponto esse modelo proposto é pertinente ao problema em questão.

A função de geração de dados desenvolvida permite a definição de um conjunto de variáveis de qualquer tamanho. Nessa hipótese, a relação entre as variáveis e a produção é ligeiramente diferente do exposto no parágrafo anterior. Ainda que o usuário possa definir

um número de áreas maior do que sete, apenas sete tipos de áreas estão configuradas nas funções de geração de dados do pacote, portanto serão instanciadas mais de uma área do mesmo tipo. A função de geração de dados, contudo, pode ser substituída se o usuário assim o desejar. Na prática, contudo, ela não deve ser usada pelo usuário do pacote, que deverá ter seus próprios dados para alimentar as redes.

Uma funcionalidade que permite a geração de um conjunto de áreas com qualquer número variáveis também está disponível no pacote. Abaixo, descrevem-se as relações entre P e as variáveis geradas:

- A_1 onde o peso da produção varia linearmente com os valores de todas as variáveis nas duas primeiras fases fenológicas,
- A_2, que varia com o quadrado das variáveis ímpares,
- A_3 cujo peso de produção varia com o quadrado das variáveis pares,
- A_4 onde a produção é inversamente proporcional à soma das variáveis ímpares,
- A_5 que decresce com uma ponderação das variáveis ímpares e
- A_6 em que produção cresce com uma ponderação das variáveis pares.

As fórmulas podem ser inferidas a partir das fórmulas restritas às 3 variáveis deste problema, ou verificadas diretamente no código.

O conjunto de dados gerados passa por tratamentos antes de ser disponibilizado para o aprendizado das estruturas das redes. Primeiramente os dados são normalizados dentro de num intervalo entre 0 e 1, a partir dos valores máximos, mínimos e médios passados como parâmetro pelo usuário. A possibilidade de passar esses valores por parâmetros advém do fato de que diferentes culturas produzem resultados bastante diversos, ainda que o valor possa ser calculado por unidade de área. Para lidar com essas diferenças de escala, a normalização dos dados foi feita. O valor final do retorno deve ser multiplicado pelo valor máximo, para que o resultado possa fazer sentido para o usuário.

O procedimento de normalização é realizado utilizando-se as funções do pacote *caret*, apresentado na Seção 2.3. Além da normalização, para permitir a construção da redes estáticas, também realiza-se a classificação dos dados, distribuindo-os em três classes: baixo, médio e alto (B, M e A), ou cinco classes baixo, médio-baixo, médio, médio-alto e alto (B, MB, M, MA e A), conforme parâmetro definido pelo usuário.

As Figuras 8 a 11 apresentam exemplos dos *dataframes* gerados pelas funções do pacote, após o processo de normalização. As figuras ilustram a possibilidade de criar dados para qualquer quantidade de áreas de produção e qualquer número de variáveis.

Figura 8 – Primeiras linhas de *dataframe* gerado com 3 variáveis contínuas

Description: df [1,000 × 4]

	X_1 <dbl>	X_2 <dbl>	X_3 <dbl>	harvest <dbl>
1	0.95835857	0.322824844	0.3049212527	0.6155548
2	0.96155081	0.248148477	0.9990884675	0.4535724
3	0.98115122	0.184487146	0.6850122407	0.5953899
4	0.98993939	0.073453524	0.2079112421	0.4030747

Fonte: Autor (2023)

Figura 9 – Primeiras linhas de *dataframe* gerado com 3 variáveis discretizadas

Description: df [1,000 × 4]

X_1 <fctr>	X_2 <fctr>	X_3 <fctr>	harvest <fctr>
MH	MH	MH	M
M	ML	H	H
M	L	MH	MH
M	ML	H	H

Fonte: Autor (2023)

Figura 10 – Primeiras linhas de *dataframe* gerado com 10 variáveis contínuas

Description: df [10 × 11]

	X_1 <dbl>	X_2 <dbl>	X_3 <dbl>	X_4 <dbl>	X_5 <dbl>	X_6 <dbl>	X_7 <dbl>	X_8 <dbl>	X_9 <dbl>	X_10 <dbl>	harvest <dbl>
1	0.7279893	0.3146348	0.0000000	0.702749526	0.68698426	0.989680324	0.3153479	0.5556862	0.7265953	0.2322359	0.5007793
2	0.4182492	0.7228562	0.7486030	1.000000000	0.19790989	0.773744029	0.2537087	0.4127663	0.5407518	0.3822479	0.5028317
3	0.4646268	1.0000000	0.3363708	0.373906175	0.01932663	0.022496654	1.0000000	0.0000000	0.7290435	0.5240568	0.3276482
4	0.5939506	0.3697138	0.9252653	0.764122502	1.00000000	0.834926860	0.9793574	1.0000000	0.5414817	0.0000000	1.0000000
5	1.0000000	0.8632983	1.0000000	0.006397772	0.62291671	0.214119501	0.4617195	0.1429959	0.4384715	0.8040931	0.2694037

Fonte: Autor (2023)

Figura 11 – Primeiras linhas de *dataframe* gerado com 10 variáveis discretizadas

Description: df [10 × 11]

X_1 <fctr>	X_2 <fctr>	X_3 <fctr>	X_4 <fctr>	X_5 <fctr>	X_6 <fctr>	X_7 <fctr>	X_8 <fctr>	X_9 <fctr>	X_10 <fctr>	harvest <fctr>
H	M	L	M	H	H	L	M	M	L	H
M	H	H	H	L	H	L	M	L	L	H
M	H	M	M	L	M	H	L	H	M	M
M	M	H	H	H	H	H	H	M	L	H

Fonte: Autor (2023)

No Capítulo 5 são discutidos os resultados obtidos a partir das construções aqui relatadas.

5 DISCUSSÃO DOS RESULTADOS

5.1 Geração e tratamento de dados

Para a validação do modelo proposto é necessário que o conjunto de dados de teste seja gerado de forma controlada, permitindo que os valores das variáveis dependentes tenham sua relação com as variáveis independentes previamente conhecida. A função *testRunDataGen* anteriormente apresentada permite a realização desta tarefa. A geração de dados utilizadas na validação do sistema é detalhada a seguir.

Considerando a limitação apontada de disponibilidade de dados relativos à produção na área de estudo, optou-se pela geração de dados fictícios para a validação dos modelos probabilísticos desenvolvidos. A geração dos dados de produção foi realizada a partir definição dos seguintes parâmetros: três variáveis X_1, X_2 e X_3, ao longo de cinco fases fenológicas em um conjunto de sete áreas, uma de cada tipo (A_1, A_2, A_3, A_4, A_5, A_6 e A_7) conforme descrito na seção anterior. Ao todo, para representação desse sistema, são necessárias $5 \cdot 7 = 35$ redes estáticas para cada área e 7 redes dinâmicas (uma para cada área de produção).

Cada uma das variáveis instanciadas varia ao longo das fases fenológicas de forma distinta onde X_1 é uma variável que sempre cresce ao longo do tempo, podendo se fazer a analogia com, por exemplo, a precipitação acumulada; X_2 representa uma variável cujo valor oscila aleatoriamente ao longo do tempo, podendo ser comparada com a insolação média; e X_3 cujo valor é constante ao longo de todo o período, semelhante ao dados do solo, como por exemplo o teor de algum mineral. A ideia aqui foi simular o comportamento, ao longo do período produtivo, de variáveis consideradas, com base no levantamento bibliográfico realizado, relevantes para a produção agrícola.

Os dados foram gerados pela função *testRunDataGen* e serviram de base para as análises no restante deste capítulo.

5.2 Construção e treinamento das redes

5.2.1 Redes Bayesianas estáticas

A partir do conjunto de dados gerados foram construídas as redes Bayesianas estáticas e dinâmicas. As redes estáticas foram construídas individualmente e cada rede

representa o recorte equivalente a uma fase fenológica de uma área. Considerando 7 áreas, com 5 fases fenológicas cada e dois algoritmos de aprendizado de estrutura utilizados, cada um com duas variações de parâmetros, foram geradas 140 redes.

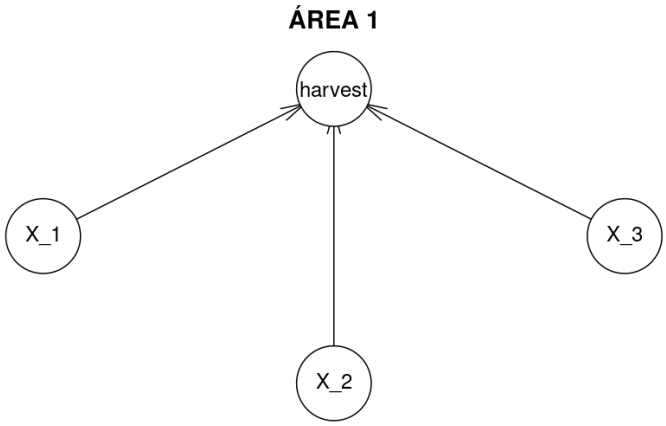
Os algoritmos de inferências de estrutura (construção de modelos) utilizados foram *Hill-Climbing* (HC) e *Max-Min Hill-Climbing* (mmhc) com e sem a predeterminação do conjunto de arcos que compõem a rede. No caso da não predeterminação de arcos, o algoritmo infere a estrutura a partir dos dados; caso contrário, a estrutura é passada pelo usuário e o algoritmo somente constrói as tabelas de probabilidades condicionais. Como é sabido, a partir das funções de geração de dados, quais são as variáveis de influência de cada área, somente arcos dessas variáveis são colocados na rede, diretamente ligados à variável que representa a produção. As redes construídas com restrições de arcos simulam a geração de redes usando a metodologia mista citada por Drury et al. (2017) onde a estrutura da rede é criada combinando o método automático (baseado em dados) e o modo manual (cuja construção é feita por especialista da área).

Hill-Climbing é um algoritmo de busca, com abordagem do tipo guloso, que explora a melhor configuração de rede a partir da adição, subtração e reversão de arcos do DAG e um sistema de pontuação. *Max-Min Hill-Climbing* é um algoritmo híbrido que combina HC com *Max-Min Parents and Children algorithm*.

As redes construídas com restrições receberam como parâmetros os conjuntos de arcos que deveriam compor a rede, de acordo com as relações definidas entre as variáveis independente e a variável dependente, conforme descrito na seção anterior nas Equações (10) a (16). As Figuras 12, 13 e 16 a 22 ilustram as redes estáticas esperadas e geradas a partir de um conjunto dados de 1.000 colheitas. Nas figuras citadas, as redes com título com o termo *raw* foram geradas sem a definição do conjunto de arcos, portanto foram construídas pelo aprendizado das probabilidades condicionais baseadas nos dados apresentados. Note-se que a topologia destas redes, a maior parte das vezes, não condiz com a estrutura esperada pelas regras de comportamento das variáveis conhecidas para cada área descritas na Seção 4.1.

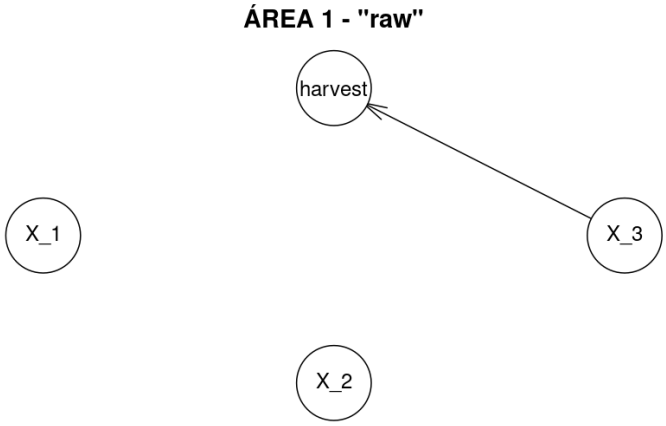
Na Figura 12 é possível ver a DAG esperada para A_1, com arcos partindo de X_1, X_2 e X_3 para *harvest*, conforme regra de comportamento das variáveis para a A_1 (Equação (10)). Todavia as redes inferidas por dados não representam o comportamento esperado, tendo sido geradas, para todas as fases fenológicas, conforme a Figura 13, com apenas um arco partindo de X_3 para *harvest*

Figura 12 – DAG - Área 1 - Esperada



Fonte: Autor (2023)

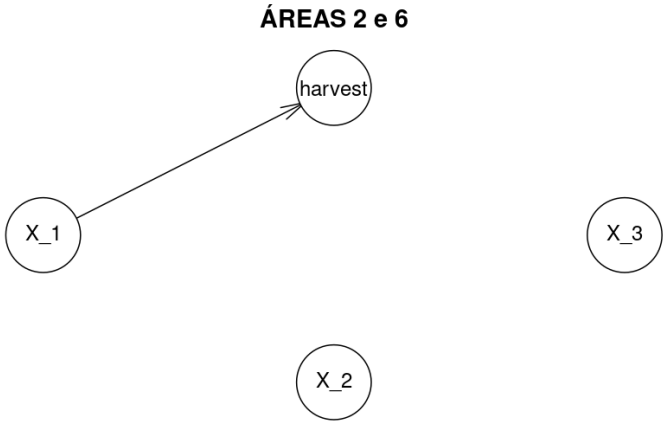
Figura 13 – DAG - Área 1 - todas fases - inferência



Fonte: Autor (2023)

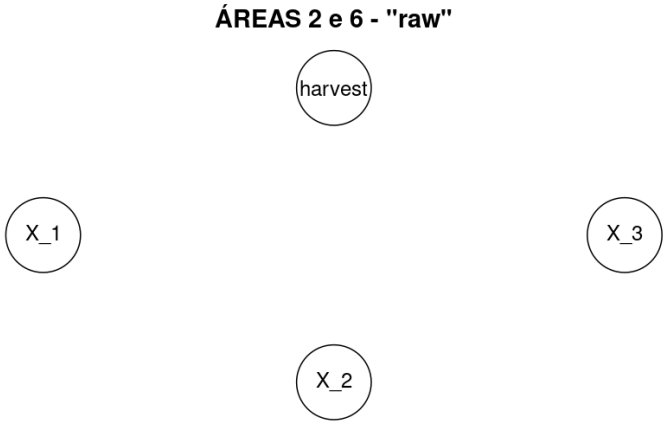
Na Figura 14 identificamos as DAG esperadas para A_2 e A_6, com um arco partindo de X_1 para *harvest*, conforme regra definidas (Equações (11) e (15)). Entretanto, para as fases 1 e 5 as redes inferidas por dados não geraram arcos, conforme a consta na Figura 15. Nas fases 2, 3 e 4 as redes foram inferidas corretamente.

Figura 14 – DAG - Áreas 2 e 6 - Esperada



Fonte: Autor (2023)

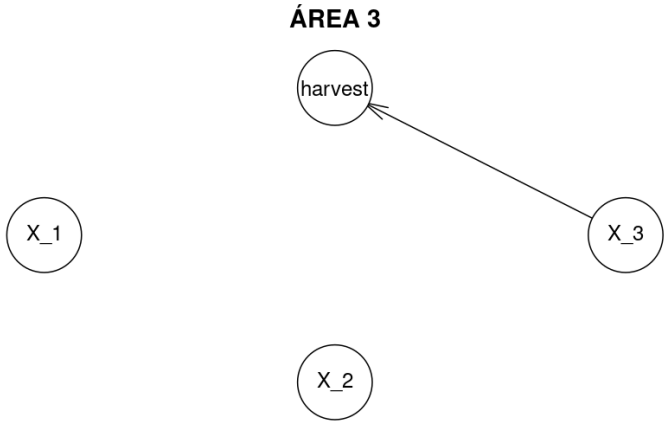
Figura 15 – DAG - Áreas 2 e 6 - fase 1 e 5 - inferência



Fonte: Autor (2023)

Para A_3 as redes geradas por inferência capturaram a estrutura esperada, com um arco partindo de X_3 para *harvest* (Equação (12)), gerando redes idênticas a apresentada na Figura 16.

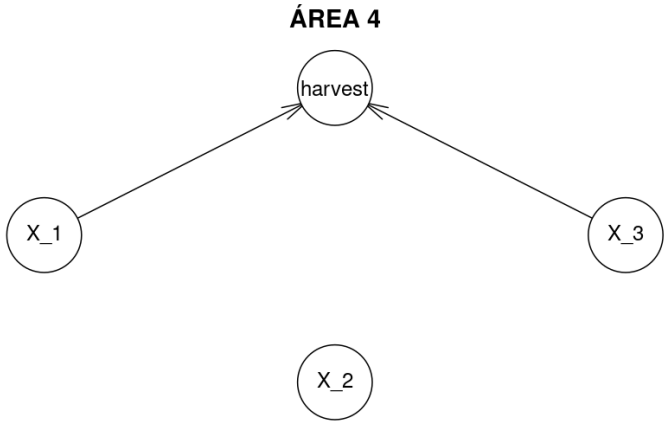
Figura 16 – DAG - Área 3 - Esperada



Fonte: Autor (2023)

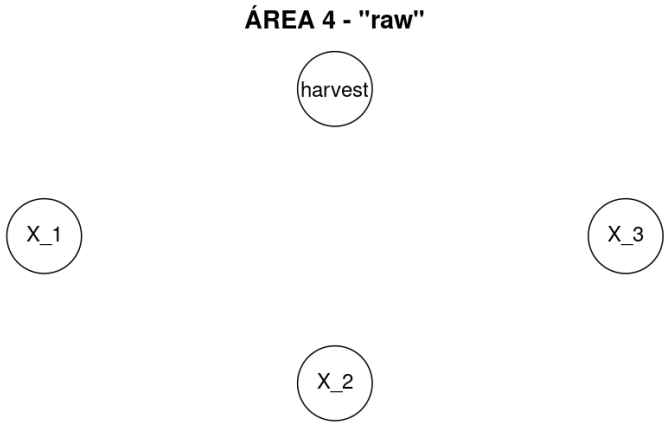
Para A_4, identificamos na Figura 17 a DAG esperada, com arcos partindo de X_1 e X_3 para *harvest*, conforme regra definidas para a A_4 (Equação (13)). Entretanto, as redes inferidas por dados não geraram arcos dirigidos para nenhuma das fases, conforme a consta na Figura 18.

Figura 17 – DAG - Área 4 - Esperada



Fonte: Autor (2023)

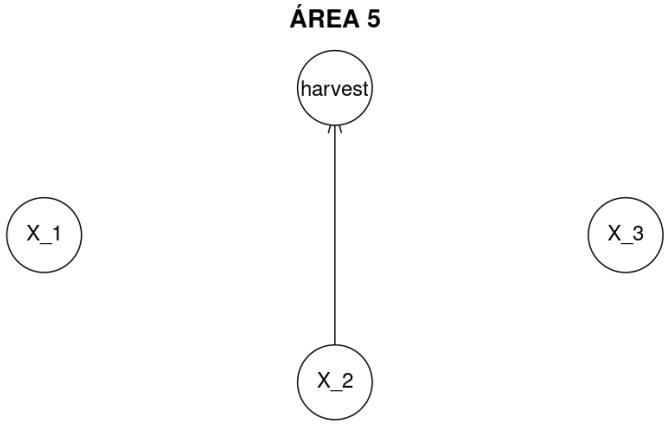
Figura 18 – DAG - Área 4 - todas fases - inferência



Fonte: Autor (2023)

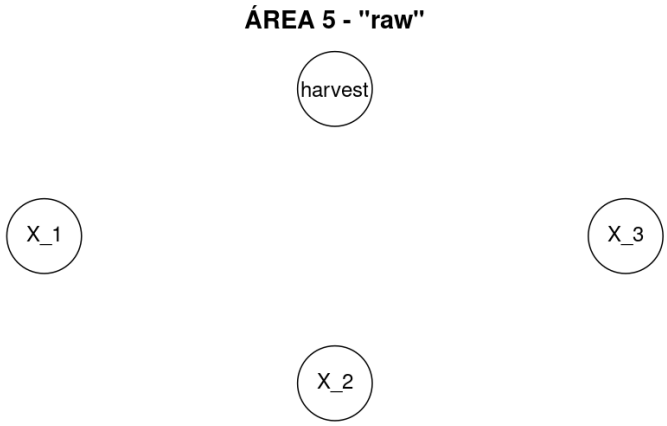
Conforme se verifica na Figura 19, para a Para A_5 a DAG esperada possui apenas um um arco partindo de X_2 para *harvest*, conforme regra definidas para a A_5 (Equação (14)). Entretanto, para as fases 4 e 5 as redes inferidas por dados não geraram arcos dirigidos, conforme a consta na Figura 20. Nas fases 1, 2 e 3 as redes foram inferidas corretamente.

Figura 19 – DAG - Área 5 - Esperada



Fonte: Autor (2023)

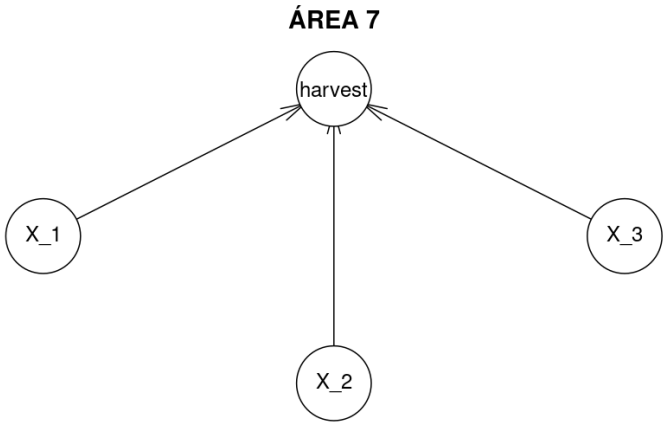
Figura 20 – DAG - Área 5 - fase 4 e 5 - inferência



Fonte: Autor (2023)

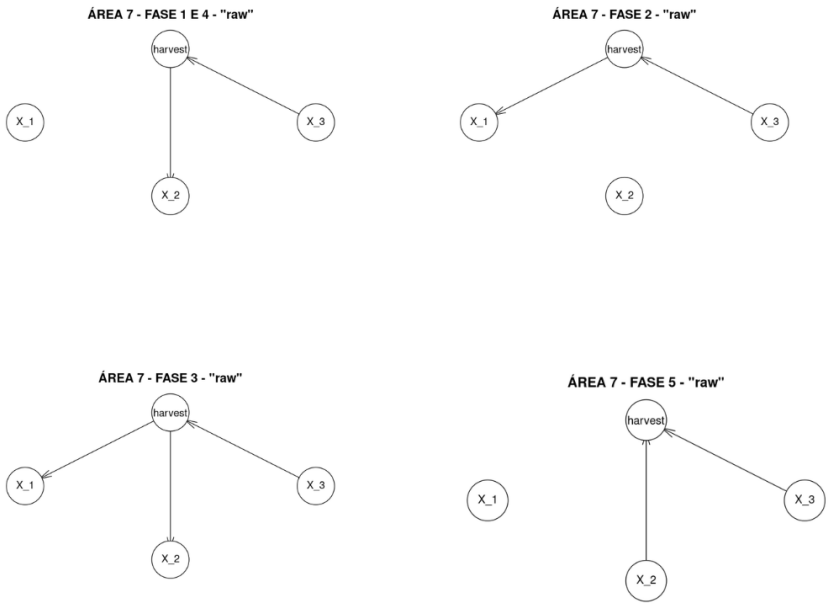
Na Figura 21 é possível ver a DAG esperada para A_7, com arcos partindo de X_1, X_2 e X_3 para *harvest*, conforme regra de comportamento das variáveis para a A_7 (Equação (16)). Todavia as redes inferidas por dados não representam o comportamento esperado, tendo sido geradas, nas fases 1 (um) e 4 (quatro), com apenas um arco partindo de X_3 para *harvest* e com um arco partindo de *harvest* para X_2. Na fase 2 (dois) vemos um arco partindo de X_3 para *harvest* e um arco partindo de *harvest* para X_1. Semelhantemente ao ocorrido nas fases 1 (um) e 2 (dois) temos, na fase 3 (três) arcos partindo X_3 para *harvest* e e dois arcos partindo de *harvest* para X_1 e X_2. Por fim na fase 5 (cinco) temos uma topologia mais próxima à esperada para A_7 com arcos partindo de X_2 e X_3 em direção a *harvest*. Todas DAG descritas podem ser visualizadas na Figura 22.

Figura 21 – DAG - Área 7 - Esperada



Fonte: Autor (2023)

Figura 22 – DAG - Área 7 - todas fases - inferência



Fonte: Autor (2023)

A diferença entre a estrutura esperada e a estrutura obtida é bastante problemática quando os relacionamentos entre as variáveis não são claros. Como todo o processo de inferência depende de probabilidades condicionais, a construção de probabilidades condicionais entre variáveis independentes não possui significado estatístico, o que quer dizer que a rede não vai inferir resultados confiáveis. A gravidade do problema pode ser minimizada quando as relações de causalidade em sistemas produtivos agropecuários estão bem definidas. Em todo o caso, não houve possibilidade de analisar as causas das diferenças entre produzido e esperado, no âmbito deste trabalho. Pode ter sido uma questão do problema ter poucas variáveis, de eventuais discrepâncias no processo de classificação dos dados ou das características dos algoritmos de inferência utilizados. Essa análise foi deixada para trabalhos futuros e a função de inferência de estrutura deve ser usada no pacote com restrições, advertidas na documentação.

Ao final da análise das redes geradas podemos dividir as redes do tipo *raw* 3 (três) categorias: As redes geradas corretamente, de acordo com as regras definidas na Seção 4.1; as redes geradas com inconsistências e as redes geradas sem arcos. As redes geradas com inconsistências no aprendizados, foram submetidas a testes com o intuito de servir como referência de base de desempenho para as redes cuja topologia está condizente com as regras, sendo esperado que sempre supere o desempenho das redes do tipo *raw*. O erro contudo, não será total, visto que em todos os casos o algoritmo de inferência capturou pelo menos uma variável de relevância para o resultado final. As redes geradas sem arcos não foram submetidas a testes por não serem confiáveis os resultados produzido.

5.2.2 Redes Bayesianas dinâmicas

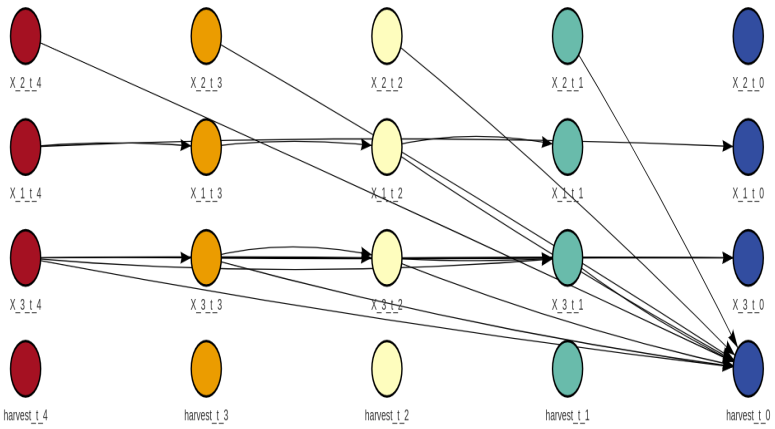
As redes dinâmicas foram construídas com as funções do pacote R *dbnR*, apresentado na Seção 2.3. A função *fold_dt*, permite o alargamento do conjunto de dados, adaptando às fatias de tempo desejadas para a rede. Normalmente, essa função simplesmente repete valores do conjunto de dados para completar as parcelas de tempo anteriores ao tempo atual t_0. No entanto, foi feita uma adaptação no resultado da função para que os dados referentes aos intervalos das fases fenológicas coincidissem com as divisões temporais geradas.

Após adequar o conjunto de dados ao formato de entrada das funções de aprendizado de rede, foram criadas duas redes dinâmicas, por meio dos algoritmos de

aprendizado de rede *dmmhc* e *natPsoho*, para cada uma das 7 áreas. As redes geradas, após treinadas, foram testadas na tarefa de previsão, utilizando-se a função *forecast_ts*, nas modalidades aproximado. Os resultados destes testes são apresentados na Seção 5.3 As redes geradas, a partir de um conjunto dados de 1.000 colheitas, estão ilustradas nas Figuras 23 a 36. Cada uma das figuras citadas representa uma rede dinâmica e nelas pode-se visualizar o fluxo temporal no sentido horizontal (da esquerda para direita), onde as redes locais (estáticas) são representadas pelos círculos da mesma cor, que retratam as variáveis da mesma fase fenológica e as diferentes instâncias das redes dinâmicas. Nas figuras podemos identificas o sufixo t_n, identificando uma contagem regressiva em direção a t_0, que representa o tempo atual, ou a fatia temporal em que se pretende fazer a previsão, t_1 a primeira fatia anterior ao tempo atual e assim por diante. Cabe esclarecer que esta regra de nomenclatura foi definida pelo autor do pacote *dbnR*.

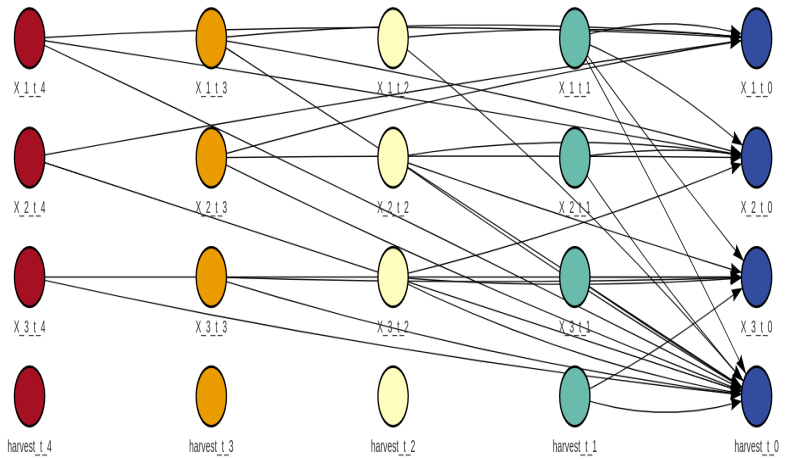
A Figura 23 pode ser utilizada como exemplo para facilitar o entendimento: na figura citada é possível verificar a existência de 5 (cinco) fases fenológicas, representadas pelos círculos com a mesma cor. Os círculos vermelhos representam a rede da primeira fase fenológica, amarelo-escuro a rede da segunda fase fenológica, amarelo-claro a da terceira, azul-claro a da quarta e azul-escuro representam a rede da última fase. Os arcos partem das redes mais à esquerda para as redes a direita indicando um fluxo de dependência dos dados, onde os nós mais à direita na figura configuram a situação das variáveis no instante da colheita sendo dependentes dos estados anteriores do sistema.

Figura 23 – DAG Rede Dinâmica - Área 1 - Dmmhc



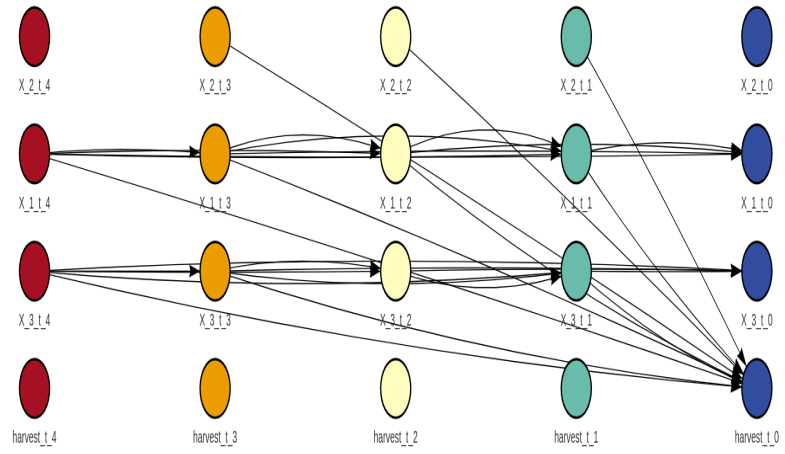
Fonte: Autor (2023)

Figura 24 – DAG Rede Dinâmica - Área 1 - natPsoho



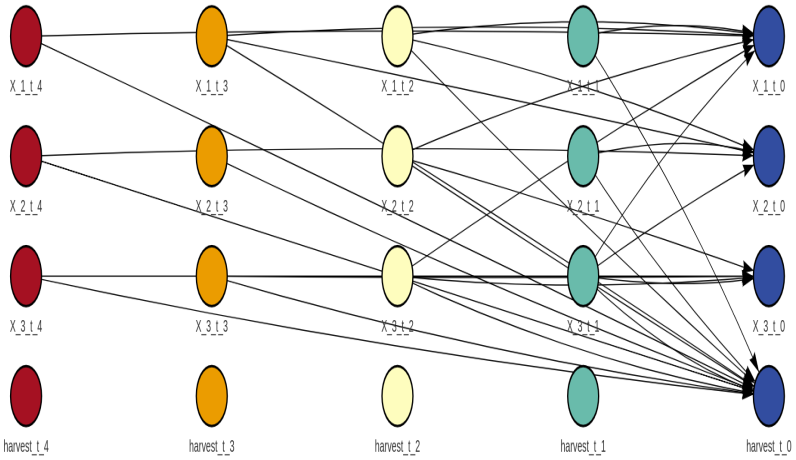
Fonte: Autor (2023)

Figura 25 – DAG Rede Dinâmica - Área 2 - Dmmhc



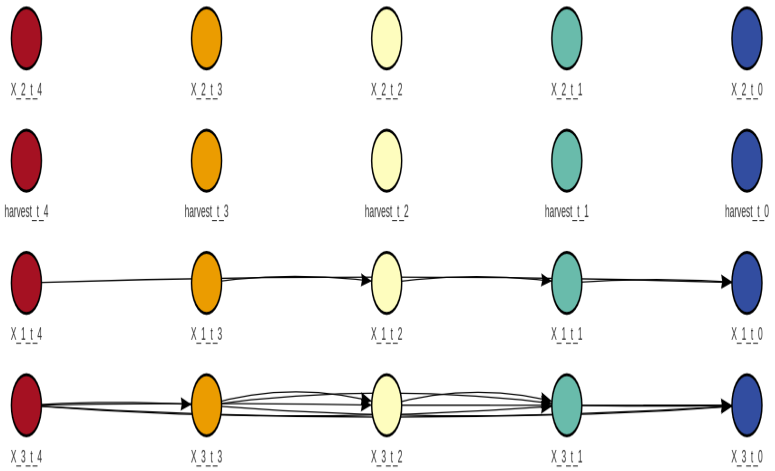
Fonte: Autor (2023)

Figura 26 – DAG Rede Dinâmica - Área 2 - natPsoho



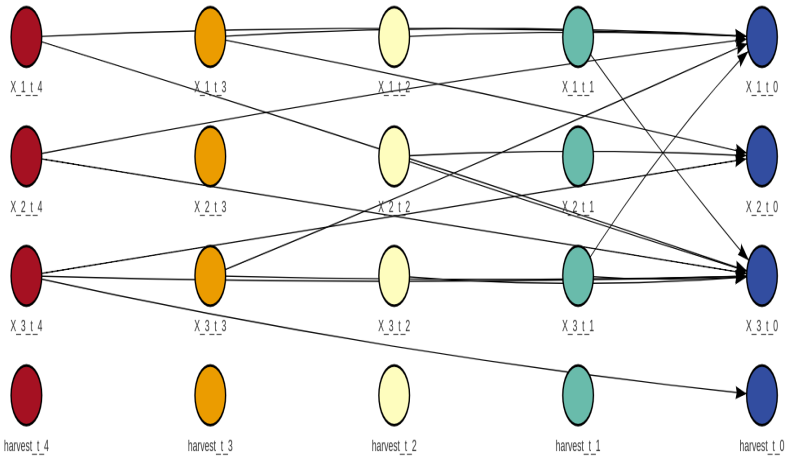
Fonte: Autor (2023)

Figura 27 – DAG Rede Dinâmica - Área 3 - Dmmhc



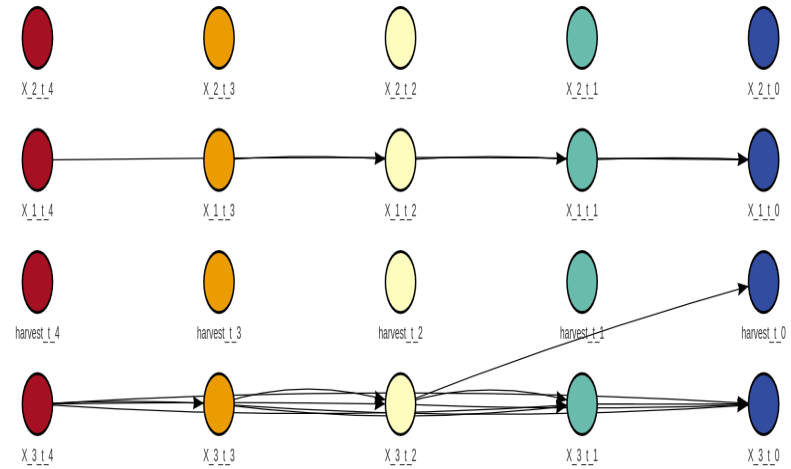
Fonte: Autor (2023)

Figura 28 – DAG Rede Dinâmica - Área 3 - natPsoho



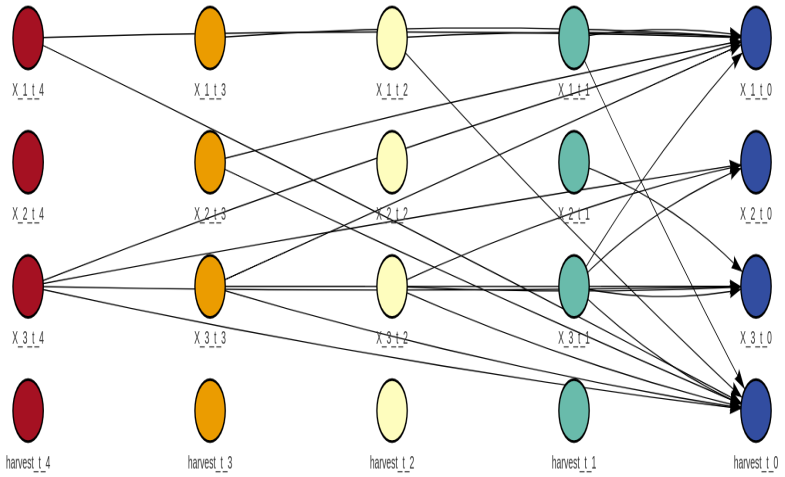
Fonte: Autor (2023)

Figura 29 – DAG Rede Dinâmica - Área 4 - Dmmhc



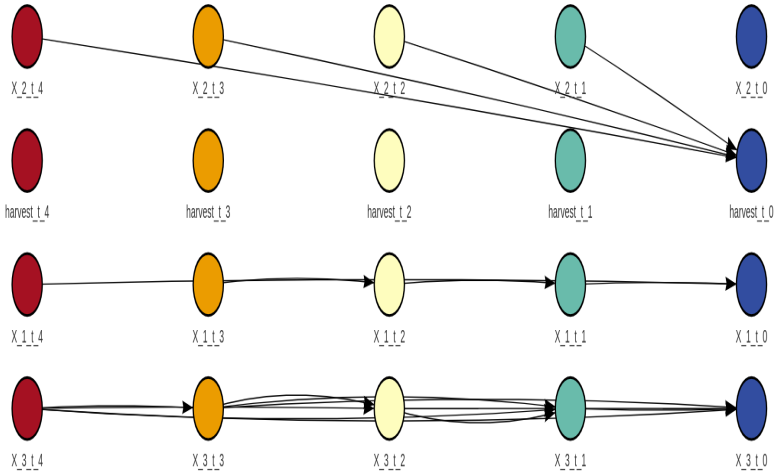
Fonte: Autor (2023)

Figura 30 – DAG Rede Dinâmica - Área 4 - natPsoho



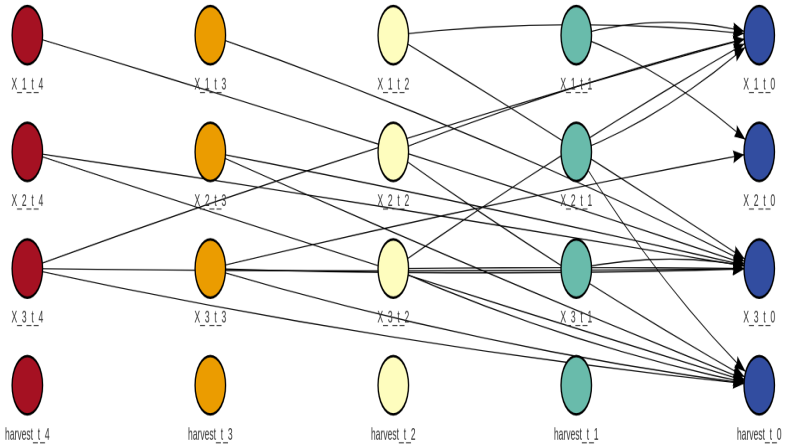
Fonte: Autor (2023)

Figura 31 – DAG Rede Dinâmica - Área 5 - Dmmhc



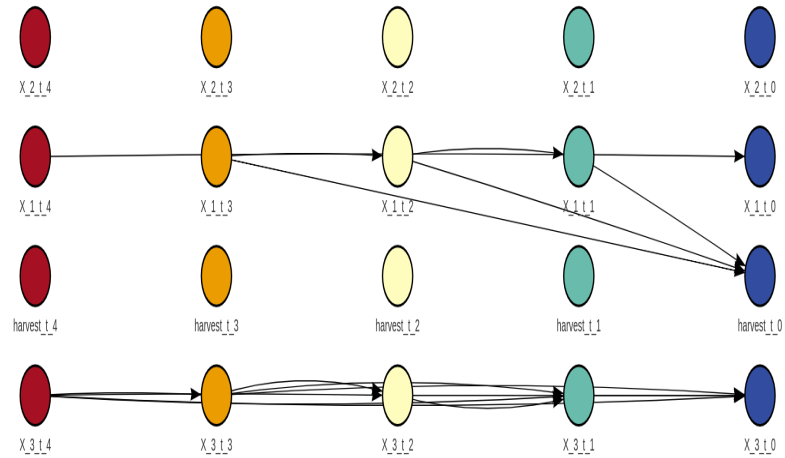
Fonte: Autor (2023)

Figura 32 – DAG Rede Dinâmica - Área 5 - natPsoho



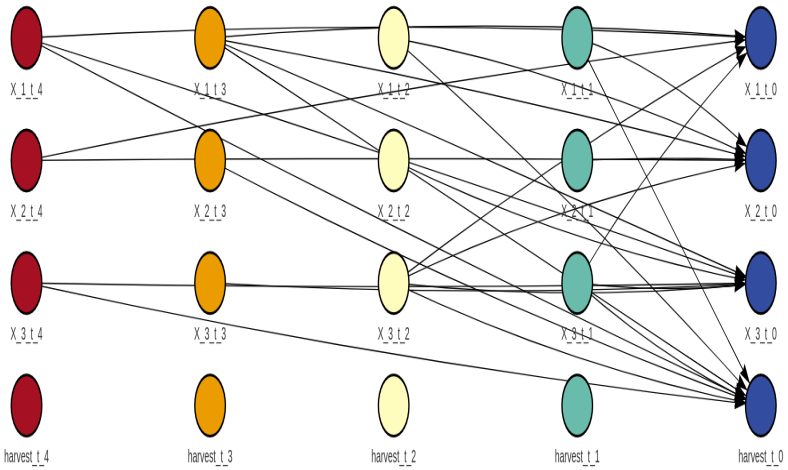
Fonte: Autor (2023)

Figura 33 – DAG Rede Dinâmica - Área 6 - Dmmhc



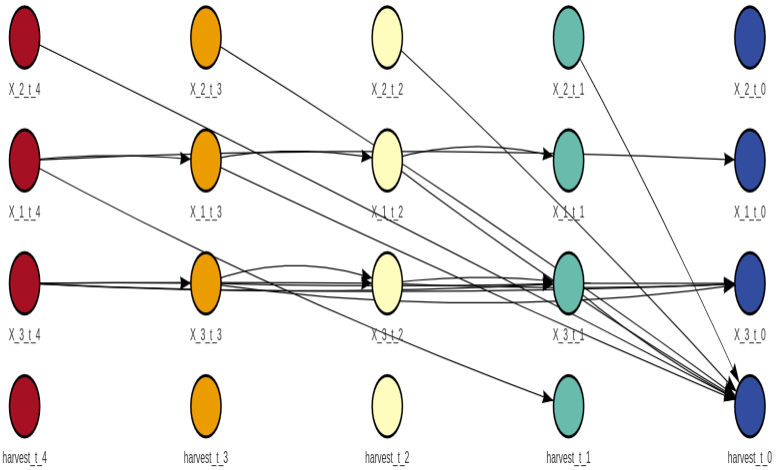
Fonte: Autor (2023)

Figura 34 – DAG Rede Dinâmica - Área 6 - natPsoho



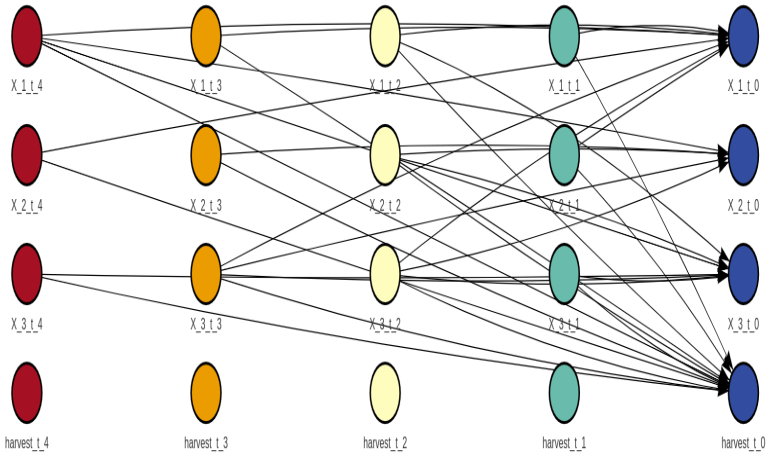
Fonte: Autor (2023)

Figura 35 – DAG Rede Dinâmica - Área 7 - Dmmhc



Fonte: Autor (2023)

Figura 36 – DAG Rede Dinâmica - Área 7 - natPsoho



Fonte: Autor (2023)

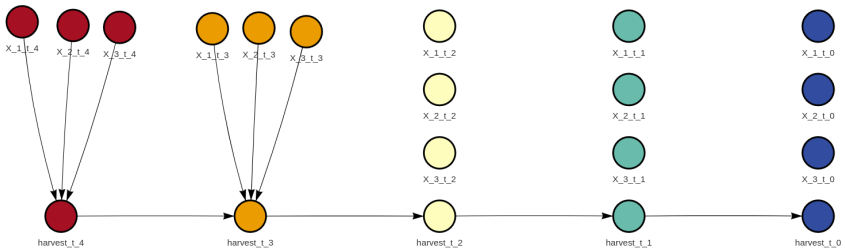
Da mesma forma que aconteceu no processo de aprendizado das redes estáticas, as redes dinâmicas também apresentaram inconsistências na topologia gerada, mas com maior gravidade, como se explica na sequência. Note-se que nenhum arco que expressa dependências entre as variáveis de uma mesma fase fenológica foi gerado; somente aparecem na figura os arcos temporais. Dessa forma, as regras definidas na Seção 4.1 parecem ter sido ignoradas. Os direcionamentos temporais também não estão de acordo com o esperado. As Figuras 25 e 26, por exemplo, permitem visualizar arcos partindo de todas as variáveis, em todos os níveis, em direção à colheita. Contudo, conforme a regra para áreas do tipo A_2, a colheita deveria depender apenas do quadrado da variável X_2 (Equação (11)). As redes representadas pelas Figuras 31 e 33 podem ser indicadas como exceção, visto que a topologia corresponde, com a ressalva da ausência dos arcos internos, à regra definida para as áreas do tipo A_5 e A_6, respectivamente.

Novamente, para que que exista a possibilidade de testar uma rede dinâmica, para verificar os seus resultados de maneira que faça sentido, optou-se pela construção manual das redes. Os arcos estáticos são os mesmos das construções de redes estáticas da Seção 5.2.1, de acordo com as funções de geração de dados definidas no pacote, com as dependências em relação a cada uma das áreas. As únicas ligações temporais existentes são entre as variáveis que representação a produção, em suas diferentes fases fenológicas. A explicação para essa escolha advém do fato de que as condições anteriores da expectativa de produção em um fase têm influência na fase seguinte. Por exemplo, se houve uma quantidade insuficiente de precipitação durante a primeira fase, que

corresponde ao plantio e germinação das plantas, a produção esperada será baixa, pois parte da plantação já está comprometida. Na fase seguinte, ainda que a quantidade de chuva seja adequada, o que aconteceu na fase anterior tem impacto. Dessa maneira, os arcos temporais representam a ideia de dependência dos estados anteriores, característicos dos processos estocásticos.

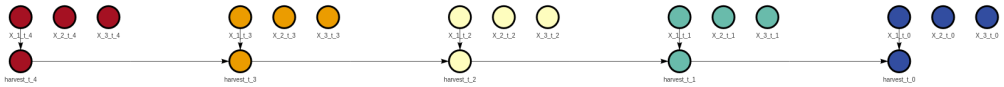
As Figuras 37 a 43 apresentam, respectivamente, as estrutura das redes dinâmicas construídas, para cada uma das sete áreas de produção.

Figura 37 – DAG Rede Dinâmica - Área 1 - construção manual



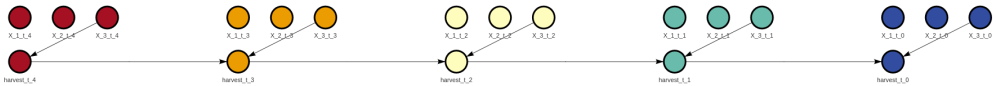
Fonte: Autor (2023)

Figura 38 – DAG Rede Dinâmica - Área 2 - construção manual



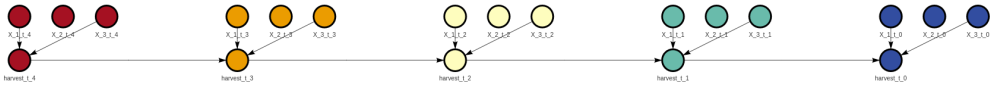
Fonte: Autor (2023)

Figura 39 – DAG Rede Dinâmica - Área 3 - construção manual



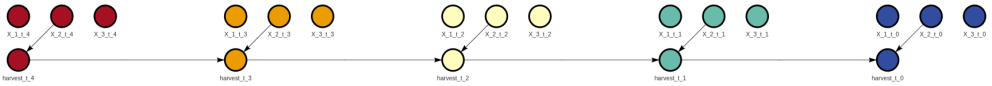
Fonte: Autor (2023)

Figura 40 – DAG Rede Dinâmica - Área 4 - construção manual



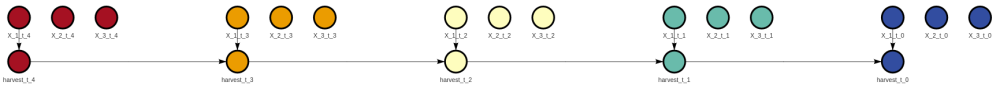
Fonte: Autor (2023)

Figura 41 – DAG Rede Dinâmica - Área 5 - construção manual



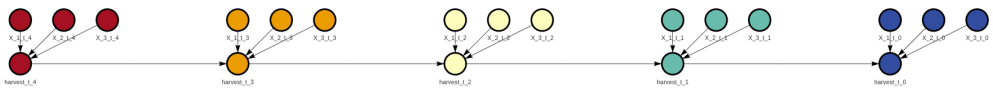
Fonte: Autor (2023)

Figura 42 – DAG Rede Dinâmica - Área 6 - construção manual



Fonte: Autor (2023)

Figura 43 – DAG Rede Dinâmica - Área 7 - construção manual



Fonte: Autor (2023)

5.3 Testes e discussão dos resultados

A validação dos modelos foi realizada com objetivo de: primeiro, identificar qual ou quais algoritmos de aprendizado de estrutura são mais eficazes na tarefa de construir redes capazes de prever o resultados das colheitas e, segundo, verificar o comportamento das redes em cenários de escassez e de abundância de dados. Conforme as discussões já

feitas nas Seções 5.2.1 e 5.2.2, os algoritmos de inferência, na maior parte das vezes, não foram capazes de capturar a estrutura esperada das redes. De toda forma, nesta seção, alguns resultados referentes às estruturas inferidas são apresentados e comparados aos resultados das redes cuja topologia foi definida manualmente.

Os seguintes cenários foram definidos para o teste das redes: 10, 100, 1.000, 5.000 e 10.000 colheitas. Evidentemente, cenários com conjuntos tão grande de dados não são, atualmente, factíveis com dados reais, entretanto o que se propõe é um exercício teórico a fim de identificar a sensibilidade do modelo proposto à disponibilidade de dados. O cenário com 10 colheitas representa uma situação de escassez de dados. Os dados utilizados nos testes foram gerados pela função *testRunDataGen*, tendo sido passado como parâmetros: 7 (áreas) áreas, 5 (cinco) fases, 3 (três) variáveis, 3 (três) classes e 10, 100, 1.000, 5.000 e 10.000 colheitas, conforme o cenário.

A construção de dois conjuntos de redes Bayesianas, estáticas e dinâmicas, visa a comparação do desempenho de cada modalidade na tarefa de inferência. No entanto, por conta das poucas opções de implementação disponíveis para construção de redes dinâmicas, as redes foram testadas sob circunstâncias diferentes. Ambos os conjuntos de redes (estáticas e dinâmicas) foram treinadas a partir dos mesmos dados. Porém, os dados usados nas redes estáticas foram previamente discretizados em classes enumeradas, enquanto os dados utilizados para as redes dinâmicas foram usados sem alteração (valores numéricos entre 0 e 1).

A divergência nos dados de entrada teve impacto nos resultado obtidos. Para as redes estáticas, as métricas derivadas da matriz de confusão foram geradas. As figuras disponíveis no Apêndice A resumem os valores de acurácia calculados para as redes estáticas em cada uma das áreas geradas, ao longo das fases fenológicas.

Para as redes dinâmicas, o erro médio absoluto (MAE) e a raiz do erro médio quadrático (RMSE) foram as métricas usadas para validação dos modelos, visto que a matriz de confusão usada para as redes estáticas não pode ser usada com valores contínuos. Entretanto, a fim de possibilitar a comparação dos resultados dos dois conjuntos de redes, foram calculados os valores de acurácia para as redes dinâmicas mediante a adoção de uma fase extra de classificação, onde os resultados obtidos pelas redes dinâmicas foram divididos em classes iguais as classes dos dados de aprendizado das redes estáticas. A comparação dos resultados encontrados nos diferentes modelos é necessária para justificar que se mantenha o desenvolvimento em paralelo dos modelos estáticos e dinâmicos no pacote, enquanto não se evolui a proposta de gerar as redes

dinâmicas a partir das redes estáticas inferidas. O desempenho equivalente encontrado em ambas as redes justifica a manutenção em paralelo de ambos modelos na versão atual do pacote.

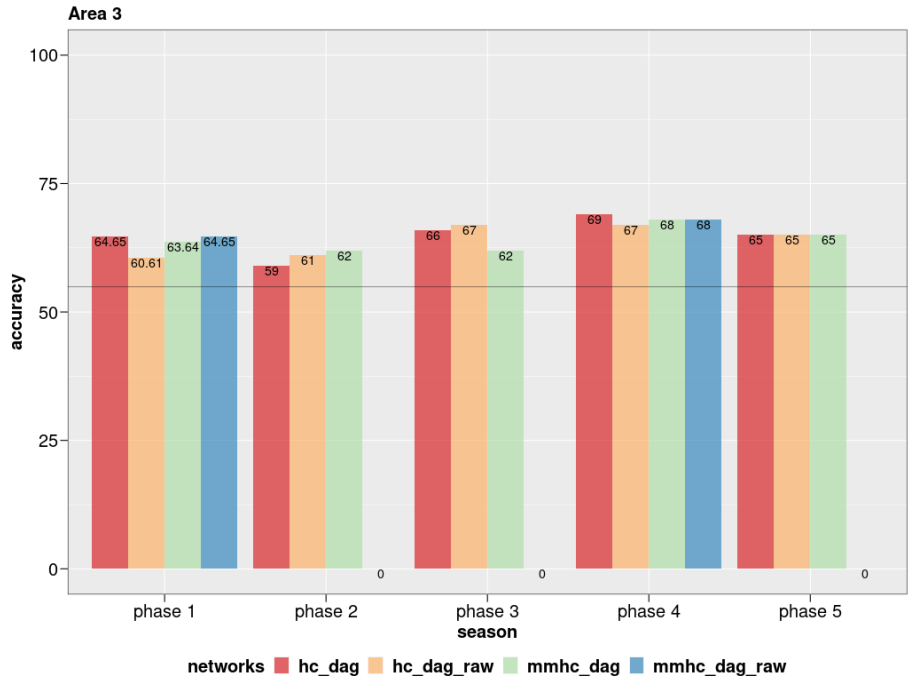
5.3.1 Resultados - Redes Bayesianas estáticas

Cada gráfico disponíveis no Apêndice A apresenta os valores da acurácia dos testes realizados com cada rede, de acordo com cada cenário de disponibilidade de dados. Cada figura representa os resultados de uma área em um cenário. Na Figura 44, por exemplo, podemos observar o desempenho das 4 redes geradas para a área A_2. As redes são identificadas por cores e nomeadas pelo algoritmo de aprendizado utilizado para a construção das mesmas: em vermelho e verde as redes geradas com topologia predefinida e em amarelo e azul as redes construídas baseadas somente em dados. Cada conjunto de barras representa o resultado das redes relativas a umas das 5 (cinco) fases fenológicas e cada barra representa o valor da acurácia resultante dos testes realizados com cada rede. Neste gráfico é possível identificar uma linha horizontal representando a acurácia média dos dados ali dispostos. Também cabe destacar que, nas fases 3 a 5 o algoritmo *mmhc* na modalidade *dag_raw* não possui acurácia exibida, isto é devido a inconsistência na geração das redes apontada nas Seções 5.2.1 e 5.2.2, portanto a eventual ausência de dados de acurácia nos gráficos constantes no Apêndice A se dá por este motivo.

Os gráficos das Figuras 54 a 60 representam o cenário de escassez de dados e indicam que os algoritmos de inferência são ineficazes na geração de redes em cenários de baixa disponibilidade de dados pois, com raras exceções, foram incapazes de gerar redes. Isto se demonstra pela ausência de dados de acurácia para as redes do tipo *_raw* nos gráficos citados. Em relação ao desempenho das redes, cabe ressalvar que, em ambos os cenários de disponibilidade de dados (com e sem escassez), a análise dos resultados faz mais sentido individualizando-se por área, visto que o comportamento das redes varia bastante de uma área para outra.

No cenário de escassez de dados as redes das A_1 e A_4 obtiveram os melhores desempenho, com acurácia média de 73% e 80% respectivamente com previsões eficazes já nas primeiras fases fenológicas. A A_2 obteve acurácia média de 65%. As demais áreas obtiveram um desempenho médio abaixo de 50%, cabendo destacar o efeito negativo da escassez de dados sobre as redes da A_7, onde a acurácia média ficou abaixo de 25%. A Tabela 6 resume os dados apresentados neste parágrafo. Conclui-se pela análise dos dados

Figura 44 – Acurácia - Redes estáticas - A3 - 100 colheitas



Fonte: Autor (2023)

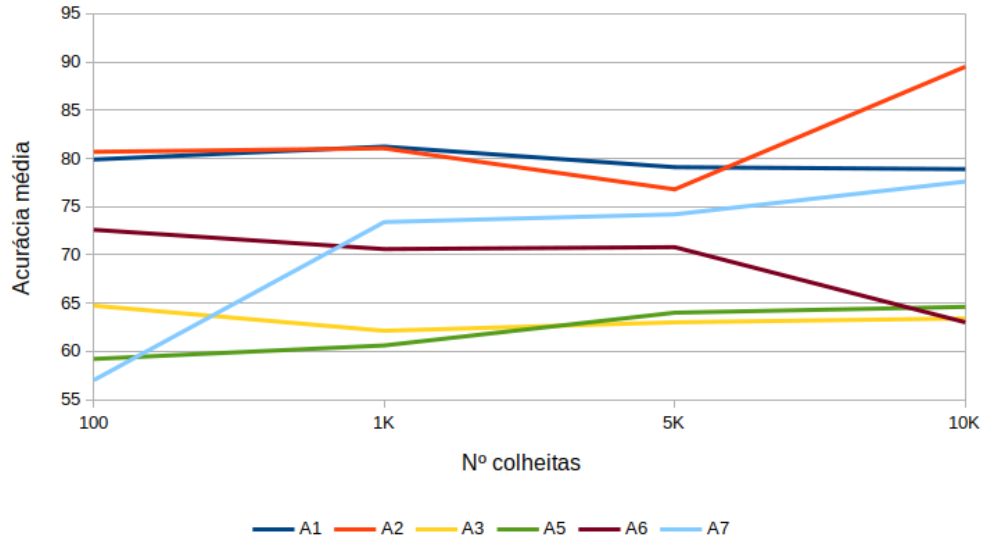
constantes nesta tabela e nos gráficos das Figuras 54 a 60 que os modelos gerados para cenários de baixa disponibilidade de dados não são confiáveis, visto que são inconstantes e apresentam uma grande variação entre fases e de área para área.

Tabela 6 – Resumo métricas - Redes estáticas - cenário: 10 colheitas

Área	Inferência		Manual	
	Média	Desv. Padrão	Média	Desv. Padrão
Área 1	65%	36.00%	59%	32.00%
Área 2	6%	18.00%	65%	15.00%
Área 3	24%	26.60%	47%	11.59%
Área 4	0%	0.00%	80%	12.84%
Área 5	3.00%	9.48%	30.00%	18.85%
Área 6	21.00%	37.80%	60.00%	18.22%
Área 7	10.00%	13.00%	25.00%	40.00%

Os resultados nos cenários com maior disponibilidade de dados, tanto de inferência da topologia das redes, quanto de desempenho na previsão dos resultados melhoram. A sensibilidade à disponibilidade de dados pode ser avaliada pela evolução da acurácia entre os cenários com 100 e 10.000 colheitas. O gráfico da Figura 45 facilita a análise, nele pode-se identificar que as acurácias tendem a aumentam juntamente com a disponibilidade de dados, para todas as áreas, crescendo principalmente no intervalo entre 100 e 1.000 colheitas, estabilizando-se, ainda que com leve crescimento, entre 1.000 e 5.000 e apresentando ligeiro decréscimo entre 5.000 e 10.000.

Figura 45 – Evolução acurácia redes tipo *HC_dag* 100 a 10.000 colheitas



Fonte: Autor (2023)

As observações apontadas no parágrafo anterior indicam que o cenário com 1.000 colheitas é o mais adequado para análise dos resultados de desempenho das redes estáticas. Novamente, individualizar as análises por área é mais indicado. As redes das A_1 e A_2 obtiveram acurácia média de 81% com desvio padrão de 0,87% e 2,1% respectivamente, tanto na modalidade *raw* como na modalidade pré definida. Nas redes da A_3 a acurácia das redes com determinação de arcos obtiveram acurácia média de 63%, com desvio padrão 0,94%, enquanto as redes do tipo *raw* obtiveram 59% de acurácia média com desvio padrão de 4,6%. Neste cenário A_4 apresenta uma acurácia média de quase 100%. Na A_5 a acurácia média foi de 61% em ambos tipos de redes, com desvio padrão de 0,76% para as comuns e 0,66% para as do tipo *raw*. Na A_6 a acurácia média foi de 74%, com desvio padrão de 4,9% em ambos tipos de redes. Nas redes da A_7 a acurácia das redes com determinação de arcos obtiveram acurácia média de 70%, com desvio padrão 1,4%, enquanto as redes do tipo *raw* obtiveram 69% de acurácia média com desvio padrão de 2%. A Tabela 7 apresenta um resumo do que foi apresentado neste parágrafo.

5.3.2 Resultados - Redes Bayesianas Dinâmicas

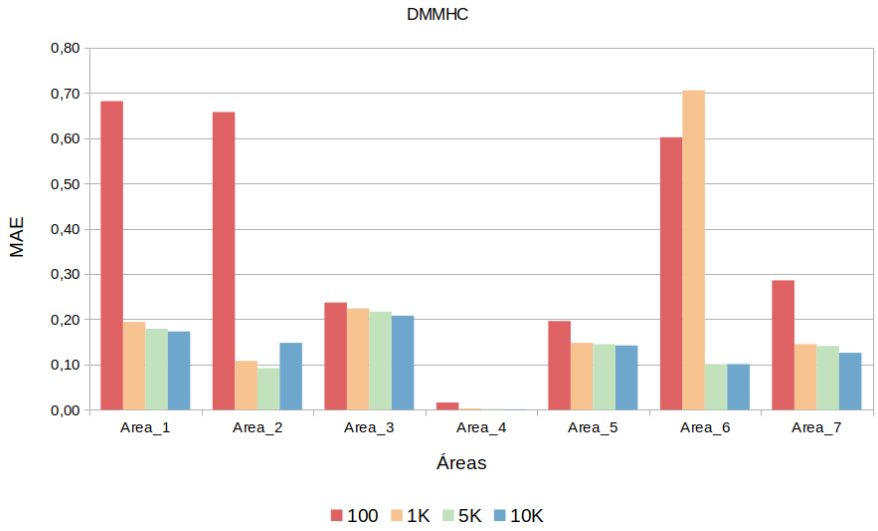
As figuras constantes do Apêndice B apresentadas as métricas geradas pelos testes das redes dinâmicas. Os algoritmos de aprendizados de estrutura das redes dinâmicas

Tabela 7 – Resumo métricas - Redes estáticas - cenário: 1.000 colheitas

Área	Inferência		Manual	
	Média	Desv. Padrão	Média	Desv. Padrão
Área 1	81%	0.87%	81%	0.87%
Área 2	81%	2.10%	81%	2.10%
Área 3	58%	4.60%	63%	0.94%
Área 4	99%	0.00%	99%	0.00%
Área 5	61.50%	0.66%	61.30%	0.76%
Área 6	74.60%	4.90%	74.60%	4.90%
Área 7	69.80%	1.98%	70.50%	1.44%

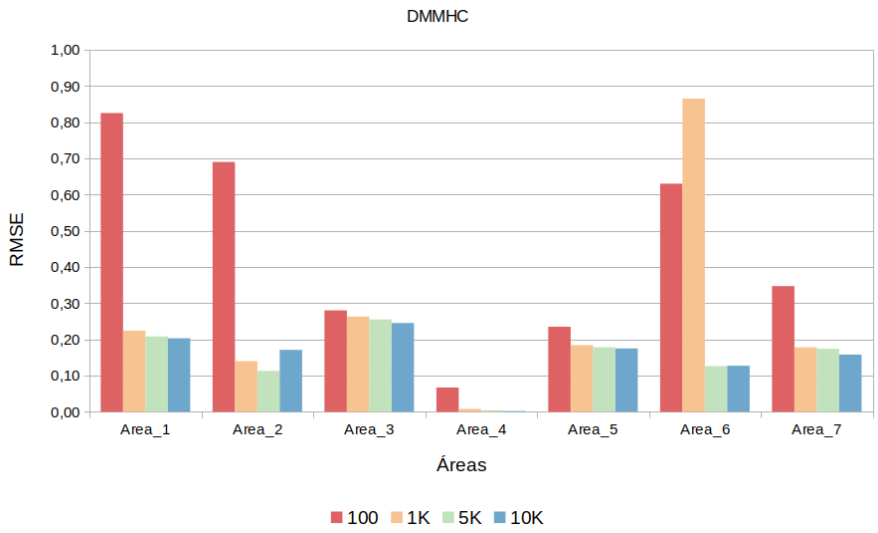
apresentaram limitações para construir redes em cenários de escassez de dados, desta forma não são apresentados resultados para os cenários com dez colheitas. Os gráficos das Figuras 46 e 47 explicitam esta dependência por dados, podem ser observados nestes gráficos que o erro médio absoluto e a raiz do erro quadrático médio, das redes geradas pelo algoritmo *dmmhc*, decrescem conforme a disponibilidade de dados aumenta.

Figura 46 – DMMHC: Evolução MAE - 100 a 10.000 colheitas



Fonte: Autor (2023)

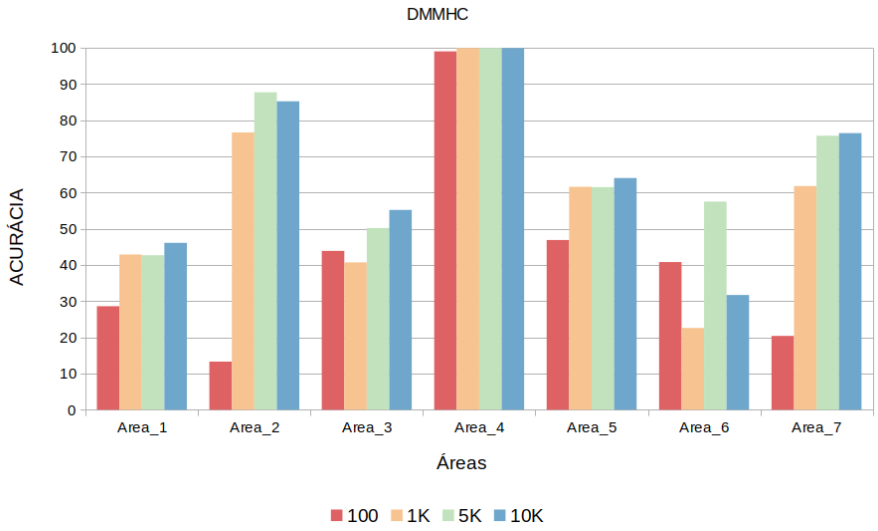
Figura 47 – DMMHC: Evolução RMSE - 100 a 10.000 colheitas



Fonte: Autor (2023)

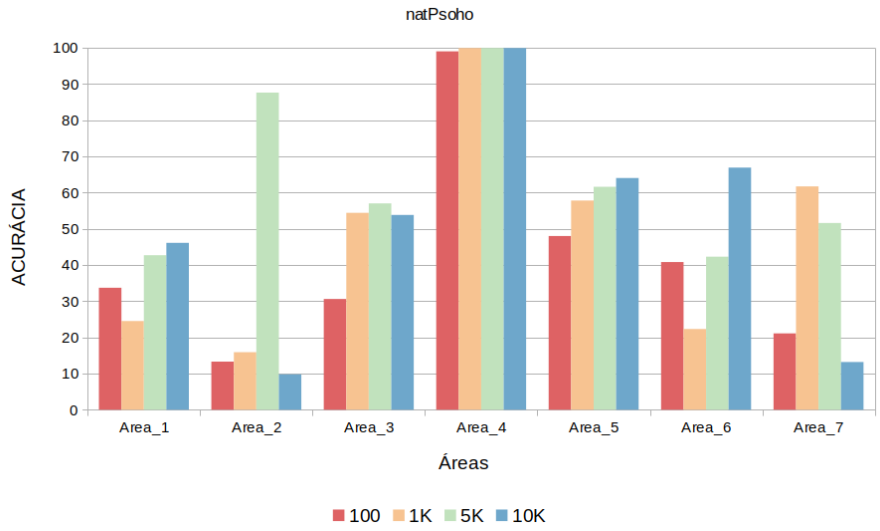
A acurácia das redes dinâmicas inferidas a partir de dados é apresentada nas Figuras 48 e 49 e das redes geradas manualmente na Figura 50. Os gráficos apresentam conjuntos de dados agrupados por área, onde cada barra representa a acurácia das redes medidas nos cenários de 100 colheitas (vermelho), 1.000 colheitas (amarelo), 5.000 colheitas (azul claro) e 10.000 colheitas (azul escuro). Analisando estes gráficos é possível verificar que os resultados obtidos foram equivalentes aos das redes estáticas apresentados na Seção 5.3.1.

Figura 48 – DMMHC: Acurácia 100 a 10.000 colheitas



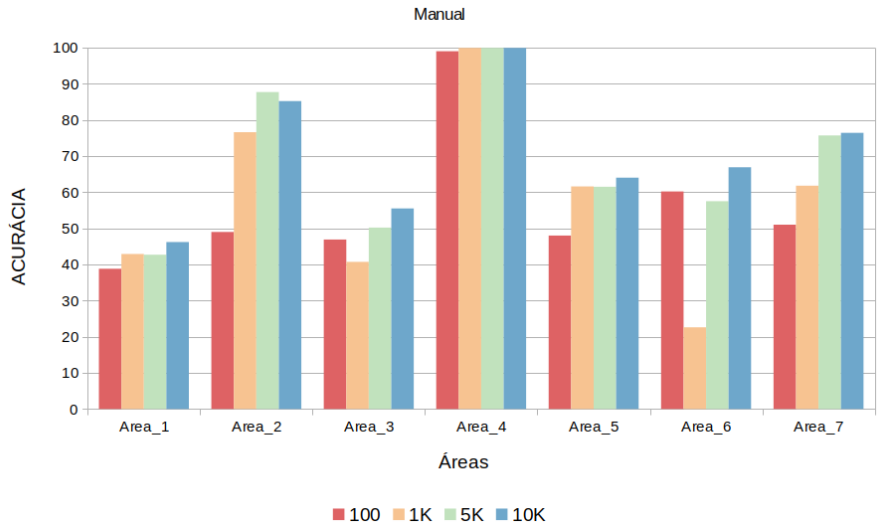
Fonte: Autor (2023)

Figura 49 – natPsoho: Acurácia 100 a 10.000 colheitas



Fonte: Autor (2023)

Figura 50 – DAG construção Manual: Acurácia 100 a 10.000 colheitas

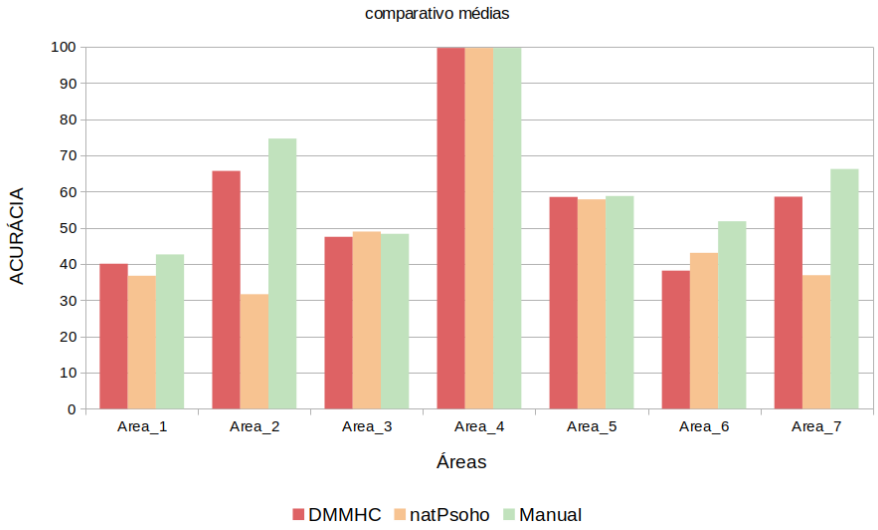


Fonte: Autor (2023)

O gráfico disposto na Figura 51 apresenta as médias das acurácias calculadas para as redes Bayesianas dinâmicas ao longo dos cenários de disponibilidade de dados, cada conjunto representa uma área em que as barras vermelhas apresentam os dados das redes construídas pelo algoritmos *Dmmhc*, as borras amarelas as redes construídas pelo *natPsoho* e em azul claro os dados das redes construídas a partir do conjunto de arcos informados manualmente, a partir das relações definidas na Seção 4.1. As informações do

gráfico permitem afirmar que as redes manuais possuem desempenho superior às demais redes em quase todas as áreas, excetuando-se apenas na A_3.

Figura 51 – Comparativo Redes dinâmicas: Acurácia média 100 a 10.000 colheitas



Fonte: Autor (2023)

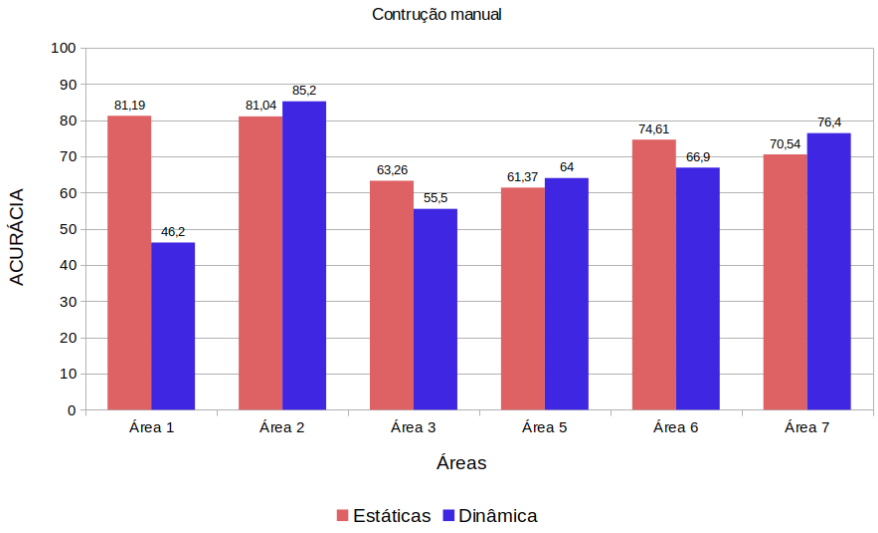
5.3.3 Comparação de resultados

Conforme observado na Seção 5.3.1 as acurácia das redes estáticas cresce quando a quantidades de colheitas cresce, estagnando entre 1.000 e 5.000 colheitas, decrescendo ligeiramente entre 5.000 e 10.000. Por outro lado, nas redes Bayesianas dinâmicas não se observou este comportamento, crescendo o desempenho juntamente com a quantidade de dados até o cenário com 10.000 colheitas. Desta forma entende-se que o comparativo de desempenho entre as redes estáticas e dinâmicas deve ser feito com os melhores cenários de cada conjunto de redes. Assim, a seguir, serão contrastados os resultados da redes estáticas no cenário com 1.000 colheitas com o das redes dinâmicas no cenário de 10.000 colheitas.

Sendo a única métrica calculada para os dois conjuntos de redes, a acurácia será utilizado a título de comparação. As Figuras 52 e 53 apresentam as médias dos desempenhos das redes estáticas (em vermelho) dinâmicas (em azul) por área, ficando evidenciado que, em se tratando das redes construídas com os arcos definidos manualmente, as redes estáticas obtiveram desempenho equilibrado, sendo que nas A_1, A_3 e A_5 as redes estáticas foram desempenharam melhor e nas A_2, A_4 e A_5 as redes dinâmicas

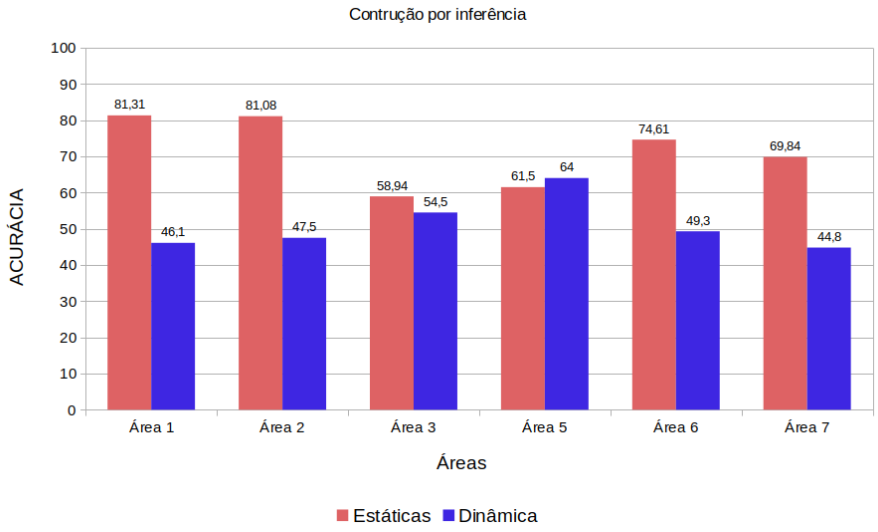
foram superiores. No gráfico da Figura 53 as redes dinâmicas representam o média do desempenho das redes construídas pelos algoritmos *dmmhc* e *natPsoho*. Neste gráfico fica claro que as redes estáticas foram melhores em quase todas as áreas, com exceção apenas na A_5. A área A_4 foi omitida para facilitar a visualização, considerando que todas as redes obtiveram acurácia de 99,9%.

Figura 52 – Acurácia: Redes dinâmicas vs Estáticas - Construção Manual



Fonte: Autor (2023)

Figura 53 – Acurácia: Redes dinâmicas vs Estáticas - Construção por inferência



Fonte: Autor (2023)

Em relação aos trabalhos correlatos, identificamos apenas dois trabalhos cujos

resultados podem ser comparados com os obtidos pelos modelos criados pelo pacote AGROBAYES: o trabalho desenvolvido por Chapman et al. (2018), utilizando redes estáticas para prever a função de produtividade futura na plantação de dendezeiros, obteve acurácias entre 75% e 95% e o trabalho de Gandhi, Armstrong e Petkar (2016) utilizando redes estáticas para previsão de plantações de arroz alcançou entre 84% e 97% de acurácia. Já as redes estáticas geradas no AGROBAYES obtiveram um desempenho médio de 75,3% para o caso das redes inferidas por dados e 76% para as redes com arcos informados manualmente, no cenário de 1.000 colheitas.

Os melhores resultados obtidos pelas redes geradas neste trabalho (desconsiderando-se A_4) foram de 84% para A_1 (cenário com 100 colheitas), 83% para A_2 (cenários com 1.000 e 5.000 colheitas) e 89,8% (cenário com 10.000 colheitas).

Ainda que os trabalhos de referência não tenham utilizado redes dinâmicas é importante registrar que a acurácia média performada pelas redes dinâmicas foi de 51% (ou 58% considerado A_4) para as redes inferidas por dados e 65,7% (ou 70,5% considerado A_4) para as redes manuais nos cenários com 10.000 colheitas.

6 CONSIDERAÇÕES FINAIS

6.1 Conclusões

Esta monografia foi desenvolvida com o objetivo de relatar os achados do levantamento bibliográfico realizado e apresentar o processo de desenvolvimento do pacote R AGROBAYES.

A partir da revisão de escopo da literatura, foram encontradas diversas abordagens sobre as técnicas de previsão de resultados de colheita, sendo mais frequentemente aplicadas as técnicas de aprendizado de máquina, seguida de trabalhos focados em técnicas de regressão. Quanto às variáveis usadas nas estimativas, houve uma distribuição com maior tendência a dados meteorológicos (temperatura, precipitação etc.), seguidos por dados de características de solo e, por fim, existem alguns trabalhos que focam em abordagem com dados climatológicos. Quanto à obtenção destes dados, as fontes citadas variam desde a coleta *in loco* por meio do emprego de redes de sensores a acesso a bancos de dados públicos, havendo menções também à utilização de imagens de sensoriamento remoto abrangendo as várias faixas do espectro eletromagnético.

A revisão de escopo também buscou aplicações de modelos de inferência probabilística no setor da produção agrícola. Dos trabalhos que utilizam redes Bayesianas encontrados na literatura, apenas quatro foram aplicados para previsão de resultado de colheita, sendo que nenhum deles aplicou redes dinâmicas para esta tarefa e apenas dois dispunham de resultados compatíveis com os gerados neste trabalhos e, por isso foram usados para comparação.

O pacote R desenvolvido pretendia disponibilizar ferramenta para inferência de redes Bayesianas, capazes de prever resultado de produção agrícola, a partir de um conjunto elástico de dados, ou seja, sem um número predeterminado de variáveis, em que se possa inclusive acrescentar novas variáveis ao modelo após a sua construção. No entanto, os modelos criados pelos algoritmos de inferência nem sempre refletiram a estrutura esperada, visto que não estabeleceram corretamente quais eram todas as variáveis que influenciavam no resultado da colheita, em cada uma das áreas de teste. Os algoritmos de inferência de modelos dinâmicos nem ao menos estabeleceram as dependências estáticas, somente dependências temporais. As dependências temporais também não refletiram o esperado em relação aos valores das variáveis. Com isso, a documentação do pacote AGROBAYES alerta ao usuário que as redes geradas pelas

funções de inferência devem ser confrontadas com a topologia de rede que melhor reflita as relações causais conhecidas do sistema solo-planta-atmosfera. Também serão incluídas dicas para a construção manual das redes-modelo.

As redes criadas a partir de conjuntos de arcos pré definidos apresentaram desempenho superior em relação as redes inferidas por dados, ainda assim com desempenho aquém do desejável para um modelo destinado para uso em casos reais. A dependência à existência de conjuntos grandes de dados também é uma fragilidade do modelo apresentado neste trabalho. Apenas em cenários com maior disponibilidade de dados os modelos foram capazes de realizar inferências com acurácia superior a 60% para todas as fases/áreas, no caso das redes estáticas e em apenas algumas áreas no caso das redes dinâmicas

Ainda que as redes inferidas com base em dados, não tenham sido, na maior parte das vezes, capazes de prever a relação existente entre a variável dependente e as variáveis independentes e que o desemprenho das redes dependa da existência de um conjunto grande de dados, o trabalho desenvolvido introduz um novo método para tratar de um problema que é simultaneamente importante e difícil. A potencial aplicação do método proposto com dados reais e a melhoria do processo de inferência das redes dependerão do desenvolvimento de trabalhos futuros.

Por fim, julga-se que houve impacto positivo na formação do discente autor em razão da realização das tarefas necessárias para atingimento dos objetivos elencados na Seção 1.2, como por exemplo a realização de revisão de escopo da literatura, codificação e a elaboração da documentação do pacote R AGROBAYES.

6.2 Trabalhos futuros

Considerando se tratar de abordagem inovadora, no que se refere à aplicação de redes Bayesianas aplicada à agricultura de precisão, entende-se que o desdobramento deste trabalho poderá desenvolver aplicações relevantes para a área. Desta forma, como trabalhos futuros indica-se a realização de um estudo mais aprofundado nas funções de aprendizado de redes baseadas em dados constantes nos pacotes *bnlear* e *dbnR* para possibilitar o desenvolvimento de funcionalidade que permita a criação das redes dinâmicas a partir das redes estáticas aprendidas individualmente, com objetivo de aumentar a eficiência das redes dinâmicas.

Objetivando facilitar o acesso ao usuário final aos modelos gerados pelo pacote

AGROBAYES, indica-se também desenvolver uma função de tratamento dos dados de entrada e a construção de interface gráfica.

Por fim, outro desdobramento possível seria explorar a capacidade de inferência abdutiva das redes Bayesianas, a fim de extrair as causar da variabilidade produtiva.

REFERÊNCIAS

AHMAD, I. et al. Remote sensing-based framework to predict and assess the interannual variability of maize yields in pakistan using landsat imagery. **Computers and Electronics in Agriculture**, v. 178, p. 105732, 2020. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169920310450.

APARECIDO, L. E. de O. et al. Agrometeorological models for forecasting coffee yield. **Agronomy Journal**, v. 109, n. 1, p. 249–258, 2017. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.2134/agronj2016.03.0166.

ARFA, N. B. et al. Agricultural policies and structural change in french dairy farms: A nonstationary markov model. **Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie**, v. 63, n. 1, p. 19–42, 2015. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1111/cjag.12036.

ARMSTRONG, J. S. Introduction. In: ____. **Principles of Forecasting: A Handbook for Researchers and Practitioners**. Boston, MA: Springer US, 2001. p. 1–12. ISBN 978-0-306-47630-3. Disponível em: https://doi.org/10.1007/978-0-306-47630-3_1.

BELLPRAT, O. et al. Towards reliable extreme weather and climate event attribution. **Nature Communications**, v. 10, n. 1, p. 1732, Apr 2019. ISSN 2041-1723. Disponível em: https://doi.org/10.1038/s41467-019-09729-2.

BOCQUET-APPEL, J.-P. When the world's population took off: The springboard of the neolithic demographic transition. **Science**, American Association for the Advancement of Science, v. 333, n. 6042, p. 560–561, 2011. ISSN 0036-8075. Disponível em: https://science.sciencemag.org/content/333/6042/560.

BROOKER, P. Experts, bayesian belief networks, rare events and aviation risk estimates. **Safety Science**, v. 49, n. 8, p. 1142–1155, 2011. ISSN 0925-7535. Disponível em: https://www.sciencedirect.com/science/article/pii/S0925753511000762.

BUSSAB, W.; DE, O.; MORETTIN, P. A. Estatística básica 6 ed. In: **Saraiva. 540 páginas**. São Paulo: [s.n.], 2010.

BYAKATONDA, J. et al. Influence of climate variability and length of rainy season on crop yields in semiarid botswana. **Agricultural and Forest Meteorology**, v. 248, p. 130–144, 2018. ISSN 0168-1923. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168192317303118.

CHANDRAPRABHA, M.; DHANARAJ, R. K. Soil based prediction for crop yield using predictive analytics. In: **2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)**. [S.l.: s.n.], 2021. p. 265–270.

CHAPMAN, R. et al. Using bayesian networks to predict future yield functions with data from commercial oil palm plantations: A proof of concept analysis. **Computers and Electronics in Agriculture**, v. 151, p. 338–348, 2018. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169917312905.

CHEN, Y. et al. Nationwide crop yield estimation based on photosynthesis and meteorological stress indices. **Agricultural and Forest Meteorology**, v. 284, p. 107872, 2020. ISSN 0168-1923. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168192319304885.

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

CORNET, D. et al. Bayesian network modeling of early growth stages explains yam interplant yield variability and allows for agronomic improvements in west africa. **European Journal of Agronomy**, v. 75, p. 80–88, 2016. ISSN 1161-0301. Disponível em: https://www.sciencedirect.com/science/article/pii/S1161030116300090.

DAGUM, P.; GALPER, A.; HORVITZ, E. Dynamic network models for forecasting. In: **Proceedings of the Eighth International Conference on Uncertainty in Artificial Intelligence**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1992. (UAI'92), p. 41–48. ISBN 1558602585.

DRURY, B. et al. A survey of the applications of bayesian networks in agriculture. **Engineering Applications of Artificial Intelligence**, v. 65, p. 29–42, 2017. ISSN 0952-1976. Disponível em: https://www.sciencedirect.com/science/article/pii/S0952197617301513.

ELAVARASAN, D. et al. Forecasting yield by integrating agrarian factors and machine learning models: A survey. **Computers and Electronics in Agriculture**, v. 155, p. 257–282, 2018. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169918311529.

FAROOQUE, A. A. et al. Forecasting potato tuber yield using a soil electromagnetic induction method. **European Journal of Soil Science**, v. 71, n. 5, p. 880–897, 2020. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1111/ejss.12923.

FATHI, M. T. et al. The relevant data mining algorithm for predicting the quality of production of olive in granada region influenced by the climate change. In: **Proceedings of the Mediterranean Symposium on Smart City Application**. New York, NY, USA: Association for Computing Machinery, 2017. (SCAMS '17). ISBN 9781450352116. Disponível em: https://doi.org/10.1145/3175628.3175649.

FERREIRA, A. P. L. Modelos de markov. Material didático em desenvolvimento. 2021.

FERREIRA, J. S. A. **Predição da variabilidade espacial da produtividade agrícola com modelos ocultos de Markov**. 90 p. Dissertação (Mestrado) — Universidade Federal do Pampa, Bagé, Dec 2019.

FERREIRA, J. S. A.; FERREIRA, A.; PEREZ, N. A hidden markov chain approach to crop yield forecasting. **IADIS International Journal on Computer Science and Information Systems**, v. 15, p. 148–160, 2020.

FOLBERTH, C. et al. Spatio-temporal downscaling of gridded crop model yield estimates based on machine learning. **Agricultural and Forest Meteorology**, v. 264, p. 1–15, 2019. ISSN 0168-1923. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168192318303162.

FOX, E. B. et al. An hdp-hmm for systems with state persistence. In: **Proceedings of the 25th International Conference on Machine Learning**. New York, NY, USA: Association for Computing Machinery, 2008. (ICML '08), p. 312–319. ISBN 9781605582054. Disponível em: https://doi.org/10.1145/1390156.1390196.

GANDHI, N.; ARMSTRONG, L. J. A review of the application of data mining techniques for decision making in agriculture. In: **2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)**. [S.l.: s.n.], 2016. p. 1–6.

GANDHI, N.; ARMSTRONG, L. J.; PETKAR, O. Predictingrice crop yield using bayesian networks. In: **2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)**. [S.l.: s.n.], 2016. p. 795–799.

Gandhi, N. et al. Rice crop yield prediction in india using support vector machines. In: **2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)**. [S.l.: s.n.], 2016. p. 1–5.

Gandhi, N.; Petkar, O.; Armstrong, L. J. Rice crop yield prediction using artificial neural networks. In: **2016 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR)**. [S.l.: s.n.], 2016. p. 105–110.

GASO, D. V.; BERGER, A. G.; CIGANDA, V. S. Predicting wheat grain yield and spatial variability at field scale using a simple regression or a crop model in conjunction with landsat images. **Computers and Electronics in Agriculture**, v. 159, p. 75–83, 2019. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169918302072.

GHAMGHAMI, M. et al. A parametric empirical bayes (peb) approach for estimating maize progress percentage at field scale. **Agricultural and Forest Meteorology**, v. 281, p. 107829, 2020. ISSN 0168-1923. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168192319304459.

GOUTHAMI, R.; BALAJI, H. Improved estimating crop yield for paddy using bayesian networks. In: **2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)**. [S.l.: s.n.], 2017. p. 1–5. ISSN 2473-943X.

GUAN, K. et al. The shared and unique values of optical, fluorescence, thermal and microwave satellite data for estimating large-scale crop yields. **Remote Sensing of Environment**, v. 199, p. 333–349, 2017. ISSN 0034-4257. Disponível em: https://www.sciencedirect.com/science/article/pii/S0034425717303024.

HAGHVERDI, A.; WASHINGTON-ALLEN, R. A.; LEIB, B. G. Prediction of cotton lint yield from phenology of crop indices using artificial neural networks. **Computers and Electronics in Agriculture**, v. 152, p. 186–197, 2018. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169918307166.

HARARI, Y. N. **Sapiens: uma breve história da humanidade**. São Paulo: L&PM, 2017. 443 p.

HARTEMINK, A. J. **Principled computational methods for the validation discovery of genetic regulatory networks**. Tese (Doutorado) — Massachusetts Institute of Technology, 2001. Disponível em: http://hdl.handle.net/1721.1/8699.

HELFER, G. A. et al. A model for productivity and soil fertility prediction oriented to ubiquitous agriculture. In: **Proceedings of the 25th Brazillian Symposium on Multimedia and the Web**. New York, NY, USA: Association for Computing Machinery, 2019. (WebMedia '19), p. 489–492. ISBN 9781450367639. Disponível em: https://doi.org/10.1145/3323503.3360637.

HIE, B. et al. Learning the language of viral evolution and escape. **Science**, American Association for the Advancement of Science, v. 371, n. 6526, p. 284–288, 2021. ISSN 0036-8075. Disponível em: https://science.sciencemag.org/content/371/6526/284.

HOLZMAN, M. E. et al. Early assessment of crop yield from remotely sensed water stress and solar radiation data. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 145, p. 297–308, 2018. ISSN 0924-2716. SI: Latin America Issue. Disponível em: https://www.sciencedirect.com/science/article/pii/S0924271618300790.

Holzman, M. E.; Rivas, R. E. Early maize yield forecasting from remotely sensed temperature/vegetation index measurements. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 9, n. 1, p. 507–519, Jan 2016. ISSN 2151-1535.

HUANG, Z.; SKLAR, E.; PARSONS, S. Design of automatic strawberry harvest robot suitable in complex environments. In: **Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction**. New York, NY, USA: Association for Computing Machinery, 2020. (HRI '20), p. 567–569. ISBN 9781450370578. Disponível em: https://doi.org/10.1145/3371382.3377443.

HUNTINGTON, T. et al. Machine learning to predict biomass sorghum yields under future climate scenarios. **Biofuels, Bioproducts and Biorefining**, v. 14, n. 3, p. 566–577, 2020. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/bbb.2087.

IIZUMI, T.; RAMANKUTTY, N. Changes in yield variability of major crops for 1981–2010 explained by climate change. **Environmental Research Letters**, IOP Publishing, v. 11, n. 3, p. 034003, feb 2016. Disponível em: https://doi.org/10.1088/1748-9326/11/3/034003.

JHA, P. K. et al. Using daily data from seasonal forecasts in dynamic crop models for yield prediction: A case study for rice in nepal's terai. **Agricultural and Forest Meteorology**, v. 265, p. 349–358, 2019. ISSN 0168-1923. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168192318303836.

JORJANI, E.; Chehreh Chelgani, S.; MESROGHLI, S. Application of artificial neural networks to predict chemical desulfurization of tabas coal. **Fuel**, v. 87, n. 12, p. 2727 – 2734, 2008. ISSN 0016-2361. Disponível em: http://www.sciencedirect.com/science/article/pii/S0016236108000409.

KADIGI, I. L. et al. Forecasting yields, prices and net returns for main cereal crops in tanzania as probability distributions: A multivariate empirical (mve) approach. **Agricultural Systems**, v. 180, p. 102693, 2020. ISSN 0308-521X. Disponível em: https://www.sciencedirect.com/science/article/pii/S0308521X18308850.

KAMIR, E.; WALDNER, F.; HOCHMAN, Z. Estimating wheat yields in australia using climate records, satellite image time series and machine learning methods. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 160, p. 124–135, 2020. ISSN 0924-2716. Disponível em: https://www.sciencedirect.com/science/article/pii/S092427161930262X.

KANG, Y.; ÖZDOğAN, M. Field-level crop yield mapping with landsat using a hierarchical data assimilation approach. **Remote Sensing of Environment**, v. 228, p. 144–163, 2019. ISSN 0034-4257. Disponível em: https://www.sciencedirect.com/science/article/pii/S0034425719301427.

KHAKZAD, N.; LANDUCCI, G.; RENIERS, G. Application of dynamic bayesian network to performance assessment of fire protection systems during domino effects. **Reliability Engineering & System Safety**, v. 167, p. 232–247, 2017. ISSN 0951-8320. Special Section: Applications of Probabilistic Graphical Models in Dependability, Diagnosis and Prognosis. Disponível em: https://www.sciencedirect.com/science/article/pii/S0951832016305828.

KHAN, J. et al. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. **Nature Medicine**, v. 7, n. 6, p. 673–679, Jun 2001. ISSN 1546-170X. Disponível em: https://doi.org/10.1038/89044.

KHANAL, S. et al. Integration of high resolution remotely sensed data and machine learning techniques for spatial prediction of soil properties and corn yield. **Computers and Electronics in Agriculture**, v. 153, p. 213–225, 2018. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169918300334.

KOCIAN, A. et al. Dynamic bayesian network for crop growth prediction in greenhouses. **Computers and Electronics in Agriculture**, v. 169, p. 105167, 2020. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169919321131.

KUNGU, E. **Difference Between Forecasting and prediction**. [S.l.]: Difference Between, 2018. http://www.differencebetween.net/science/difference-between-forecasting-and-prediction/. Accessed: 2023-2-2.

LEHMANN, J. et al. Potential for early forecast of moroccan wheat yields based on climatic drivers. **Geophysical Research Letters**, v. 47, n. 12, p. e2020GL087516, 2020. E2020GL087516 2020GL087516. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1029/2020GL087516.

LEROUX, L. et al. Maize yield estimation in west africa from crop process-induced combinations of multi-domain remote sensing indices. **European Journal of Agronomy**, v. 108, p. 11–26, 2019. ISSN 1161-0301. Disponível em: https://www.sciencedirect.com/science/article/pii/S116103011830354X.

Li, N. et al. Triggered measurements in markov processes for entropy-constrained state estimation with application to precision agriculture. In: **2020 59th IEEE Conference on Decision and Control (CDC)**. [S.l.: s.n.], 2020. p. 3611–3616. ISSN 2576-2370.

Liman Harou, I. et al. Crop modelling in data-poor environments – a knowledge-informed probabilistic approach to appreciate risks and uncertainties in flood-based farming systems. **Agricultural Systems**, v. 187, p. 103014, 2021. ISSN 0308-521X. Disponível em: https://www.sciencedirect.com/science/article/pii/S0308521X20308751.

LóPEZ, D. Q.; CASTILLA, G. V. **Multivariate forecast of Ecuadorian financial indexes using Gaussian DBNs with an extension of bnlearn**. 2019.

MAPA. **Agricultura de Precisão**. Brasília, DF, 2011. 36 p.

Mladenova, I. E. et al. Intercomparison of soil moisture, evaporative stress, and vegetation indices for estimating corn and soybean yields over the u.s. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 10, n. 4, p. 1328–1343, April 2017. ISSN 2151-1535.

MOR, B.; GARHWAL, S.; KUMAR, A. A systematic review of hidden markov models and their applications. **Archives of Computational Methods in Engineering**, v. 28, n. 3, p. 1429–1448, May 2021. ISSN 1886-1784. Disponível em: https://doi.org/10.1007/s11831-020-09422-4.

MORAES, J. R. da Silva Cabral de et al. Agrometeorological models to forecast açaí (euterpe oleracea mart.) yield in the eastern amazon. **Journal of the Science of Food and Agriculture**, v. 100, n. 4, p. 1558–1569, 2020. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/jsfa.10164.

MUELLER-WARRANT, G. W. et al. Spatial methods for deriving crop rotation history. **International Journal of Applied Earth Observation and Geoinformation**, v. 60, p. 22–37, 2017. ISSN 0303-2434. Disponível em: https://www.sciencedirect.com/science/article/pii/S0303243417300740.

MUKHERJEE, A.; Nag Biswas, S. Artificial neural networks in prediction of mechanical behavior of concrete at high temperature. **Nuclear Engineering and Design**, v. 178, n. 1, p. 1 – 11, 1997. ISSN 0029-5493. Disponível em: http://www.sciencedirect.com/science/article/pii/S0029549397001520.

NADERPOUR, M.; LU, J.; ZHANG, G. A fuzzy dynamic bayesian network-based situation assessment approach. In: IEEE. **2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)**. [S.l.], 2013. p. 1–8.

NEIVA, F.; SILVA, R. **Revisão Sistemática da Literatura em Ciência da Computação - Um Guia Prático**. Juiz de Fora, 2016.

NGUYEN-HUY, T. et al. Modeling the joint influence of multiple synoptic-scale, climate mode indices on australian wheat yield using a vine copula-based approach. **European Journal of Agronomy**, v. 98, p. 65–81, 2018. ISSN 1161-0301. Disponível em: https://www.sciencedirect.com/science/article/pii/S1161030118301096.

OTTO, F. E. L. et al. Attributing high-impact extreme events across timescales—a case study of four different types of events. **Climatic Change**, v. 149, n. 3, p. 399–412, Aug 2018. ISSN 1573-1480. Disponível em: https://doi.org/10.1007/s10584-018-2258-3.

PARK, J. et al. A layered features analysis in smart farm environments. In: **Proceedings of the International Conference on Big Data and Internet of Thing**. New York, NY, USA: Association for Computing Machinery, 2017. (BDIOT2017), p. 169–173. ISBN 9781450354301. Disponível em: https://doi.org/10.1145/3175684.3175720.

Patil, S. S.; Thorat, S. A. Early detection of grapes diseases using machine learning and iot. In: **2016 Second International Conference on Cognitive Computing and Information Processing (CCIP)**. [S.l.: s.n.], 2016. p. 1–5.

Pawara, S. et al. Early detection of pomegranate disease using machine learning and internet of things. In: **2018 3rd International Conference for Convergence in Technology (I2CT)**. [S.l.: s.n.], 2018. p. 1–4.

PENG, B. et al. Benefits of seasonal climate prediction and satellite data for forecasting u.s. maize yield. **Geophysical Research Letters**, v. 45, n. 18, p. 9662–9671, 2018. 2018GL079291. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1029/2018GL079291.

POPLI, S.; JHA, R. K.; JAIN, S. Adaptive small cell position algorithm (aspa) for green farming using nb-iot. **Journal of Network and Computer Applications**, v. 173, p. 102841, 2021. ISSN 1084-8045. Disponível em: https://www.sciencedirect.com/science/article/pii/S108480452030309X.

PRICE, C. S.; MOODLEY, D.; PILLAY, A. W. Dynamic bayesian decision network to represent growers' adaptive pre-harvest burning decisions in a sugarcane supply chain. In: **Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists**. New York, NY, USA: Association for Computing Machinery, 2018. (SAICSIT '18), p. 89–98. ISBN 9781450366472. Disponível em: httpsdoi.org10.11453278681.3278693.

QUESADA, D.; BIELZA, C.; LARRAÑAGA, P. Structure learning of high-order dynamic bayesian networks via particle swarm optimization with order invariant encoding. In: GONZÁLEZ, H. S. et al. (Ed.). **Hybrid Artificial Intelligent Systems**. Cham: Springer International Publishing, 2021. p. 158–171. ISBN 978-3-030-86271-8.

ROBERTON, S.; LOBSEY, C.; BENNETT, J. A bayesian approach toward the use of qualitative information to inform on-farm decision making: The example of soil compaction. **Geoderma**, v. 382, p. 114705, 2021. ISSN 0016-7061. Disponível em: https://www.sciencedirect.com/science/article/pii/S0016706119329465.

SCHAUBERGER, B.; GORNOTT, C.; WECHSUNG, F. Global evaluation of a semiempirical model for yield anomalies and application to within-season yield forecasting. **Global Change Biology**, v. 23, n. 11, p. 4750–4764, 2017. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1111/gcb.13738.

SCHNEIDER, U. A. et al. Impacts of population growth, economic development, and technical change on global food production and consumption. **Agricultural Systems**, v. 104, n. 2, p. 204 – 215, 2011. ISSN 0308-521X. Methods and tools for integrated

assessment of sustainability of agricultural systems and land use. Disponível em: http://www.sciencedirect.com/science/article/pii/S0308521X10001575.

SCUTARI, M. Learning bayesian networks with the bnlearn R package. **Journal of Statistical Software**, v. 35, n. 3, p. 1–22, 2010.

SHAMSHAD, A. et al. First and second order markov chain models for synthetic generation of wind speed time series. **Energy**, v. 30, n. 5, p. 693–708, 2005. ISSN 0360-5442. Disponível em: https://www.sciencedirect.com/science/article/pii/S0360544204002609.

SHU, K. Prediction of soybean yield using self-normalizing neural networks. In: **Proceedings of the 2020 5th International Conference on Machine Learning Technologies**. New York, NY, USA: Association for Computing Machinery, 2020. (ICMLT 2020), p. 51–55. ISBN 9781450377645. Disponível em: https://doi.org/10.1145/3409073.3409092.

SOUZA, K. X. S. de et al. Agricultura digital: definições e tecnologias. **Embrapa Informática Agropecuária-Capítulo em livro científico (ALICE)**, In: MASSRUHÁ, SMFS; LEITE, MA de A.; OLIVEIRA, SR de M.; MEIRA, CAA . . . , 2020.

SRIKAMDEE, S.; RIMCHAROEN, S.; LEELATHAKUL, N. Sugarcane yield and quality forecasting models: Adaptive es vs. deep learning. In: **Proceedings of the 2nd International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence**. New York, NY, USA: Association for Computing Machinery, 2018. (ISMSI '18), p. 6–11. ISBN 9781450364126. Disponível em: https://doi.org/10.1145/3206185.3206190.

TORRES, M.; HOWITT, R.; RODRIGUES, L. Analyzing rainfall effects on agricultural income: Why timing matters. **EconomiA**, v. 20, n. 1, p. 1 – 14, 2019. ISSN 1517-7580. Disponível em: http://www.sciencedirect.com/science/article/pii/S1517758018301310.

TRABELSI, G. **New structure learning algorithms and evaluation methods for large dynamic Bayesian networks**. Tese (Theses) — Université de Nantes ; Ecole Nationale d'Ingénieurs de Sfax, dez. 2013. Disponível em: https://theses.hal.science/tel-00996061.

UNO, Y. et al. Artificial neural networks to predict corn yield from compact airborne spectrographic imager data. **Computers and Electronics in Agriculture**, v. 47, n. 2, p. 149 – 161, 2005. ISSN 0168-1699. Disponível em: http://www.sciencedirect.com/science/article/pii/S0168169904001577.

Uwizera, D.; McSharry, P. Forecasting and monitoring maize production using satellite imagery in rwanda. In: **2017 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR)**. [S.l.: s.n.], 2017. p. 51–56.

van Klompenburg, T.; KASSAHUN, A.; CATAL, C. Crop yield prediction using machine learning: A systematic literature review. **Computers and Electronics in Agriculture**, v. 177, p. 105709, 2020. ISSN 0168-1699. Disponível em: https://www.sciencedirect.com/science/article/pii/S0168169920302301.

VITERBI, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. **IEEE Transactions on Information Theory**, v. 13, n. 2, p. 260–269, 1967.

WANG, M.-M. et al. Identification of the most limiting factor for rice yield using soil data collected before planting and during the reproductive stage. **Land Degradation & Development**, v. 29, n. 8, p. 2310–2320, 2018. LDD-17-0226.R2. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/ldr.3026.

WHITE, J. et al. Improving crop yield forecasts with satellite-based soil moisture estimates: An example for township level canola yield forecasts over the canadian prairies. **International Journal of Applied Earth Observation and Geoinformation**, v. 89, p. 102092, 2020. ISSN 0303-2434. Disponível em: https://www.sciencedirect.com/science/article/pii/S0303243419313534.

YANG, S. et al. Visualization analysis of markov chain based on citespace. In: **Proceedings of the 2019 11th International Conference on Information Management and Engineering**. New York, NY, USA: Association for Computing Machinery, 2019. (ICIME 2019), p. 15–19. ISBN 9781450372343. Disponível em: https://doi.org/10.1145/3373744.3373750.

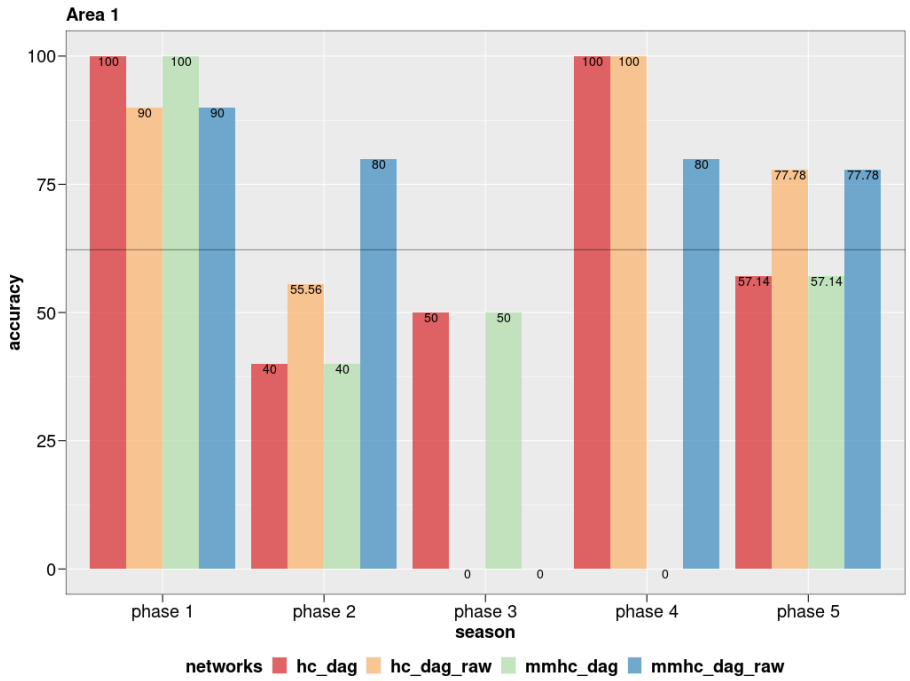
Yashaswini, L. S. et al. Smart automated irrigation system with disease prediction. In: **2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)**. [S.l.: s.n.], 2017. p. 422–427.

ZAEEN, A. A. et al. In-season potato yield prediction with active optical sensors. **Agrosystems, Geosciences & Environment**, v. 3, n. 1, p. e20024, 2020. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/agg2.20024.

ZHANG, N. et al. Winter wheat yield prediction using normalized difference vegetative index and agro-climatic parameters in oklahoma. **Agronomy Journal**, v. 109, n. 6, p. 2700–2713, 2017. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.2134/agronj2017.03.0133.

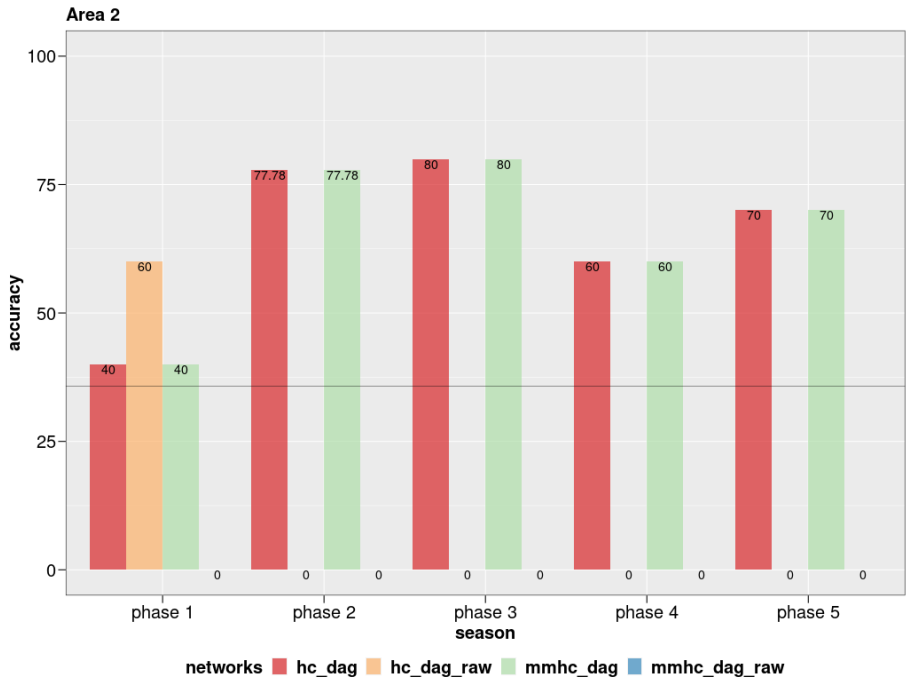
APÊNDICE A – GRÁFICOS – REDES ESTÁTICAS

Figura 54 – Acurácia - Redes estáticas - A1 - 10 colheitas



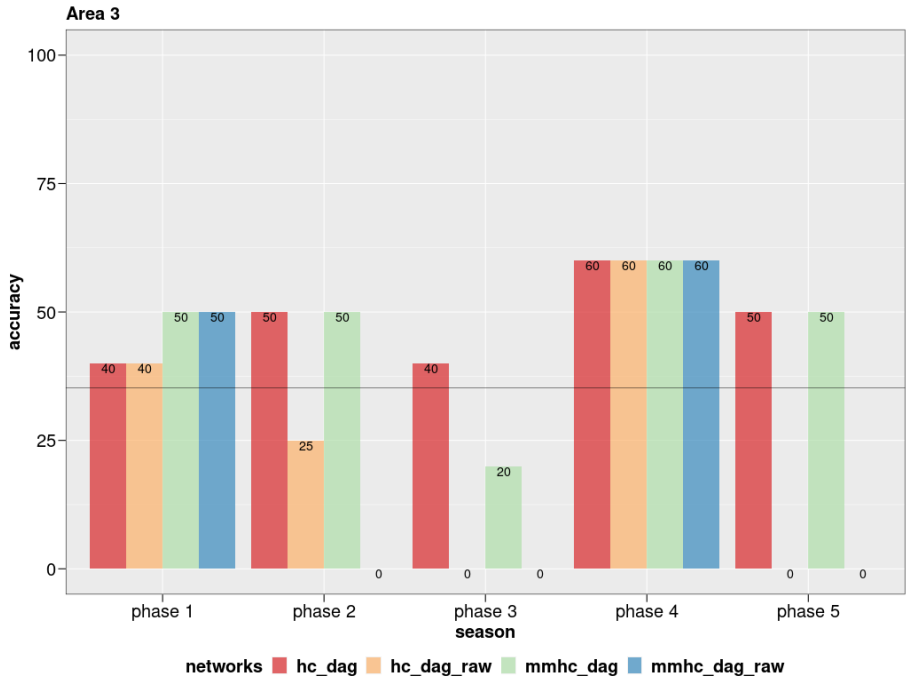
Fonte: Autor (2023)

Figura 55 – Acurácia - Redes estáticas - A2 - 10 colheitas



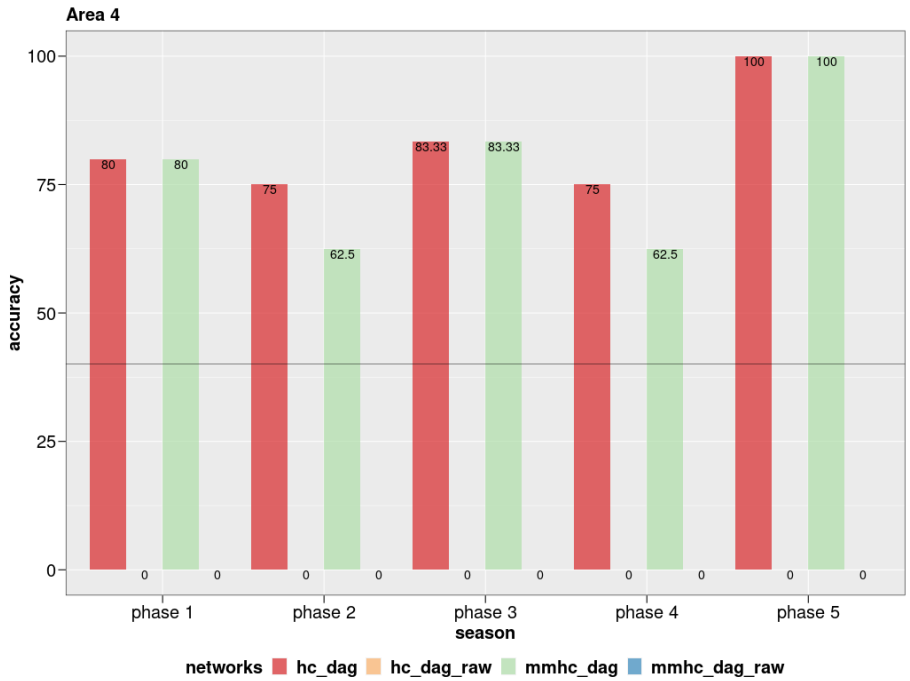
Fonte: Autor (2023)

Figura 56 – Acurácia - Redes estáticas - A3 - 10 colheitas



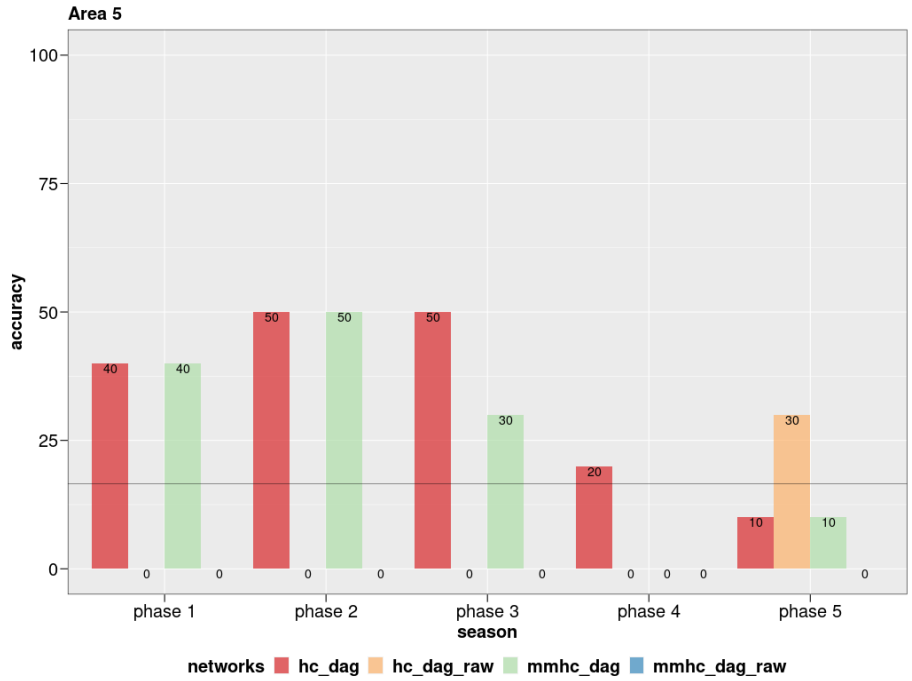
Fonte: Autor (2023)

Figura 57 – Acurácia - Redes estáticas - A4 - 10 colheitas



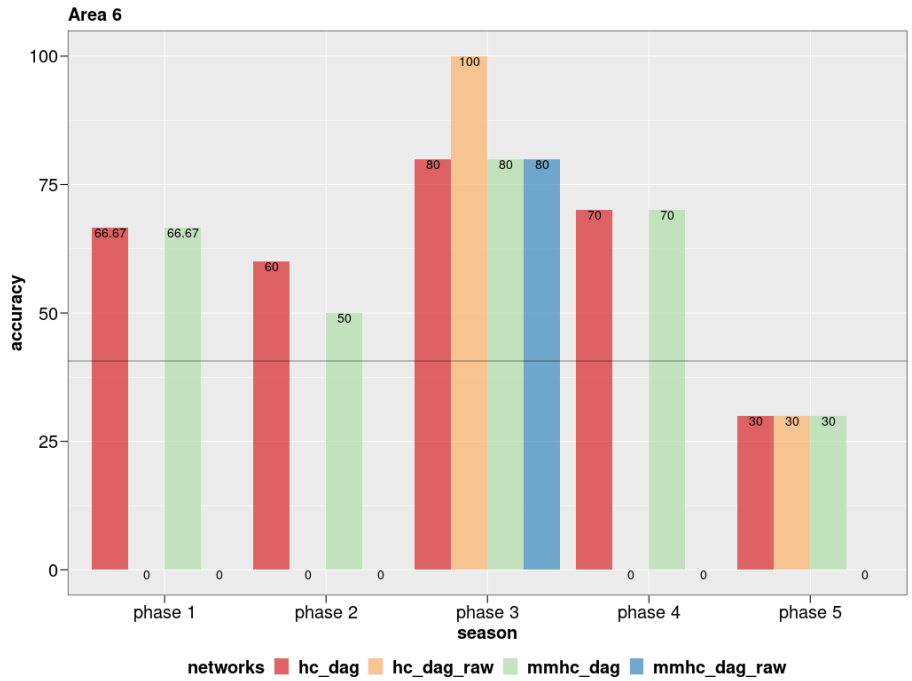
Fonte: Autor (2023)

Figura 58 – Acurácia - Redes estáticas - A5 - 10 colheitas



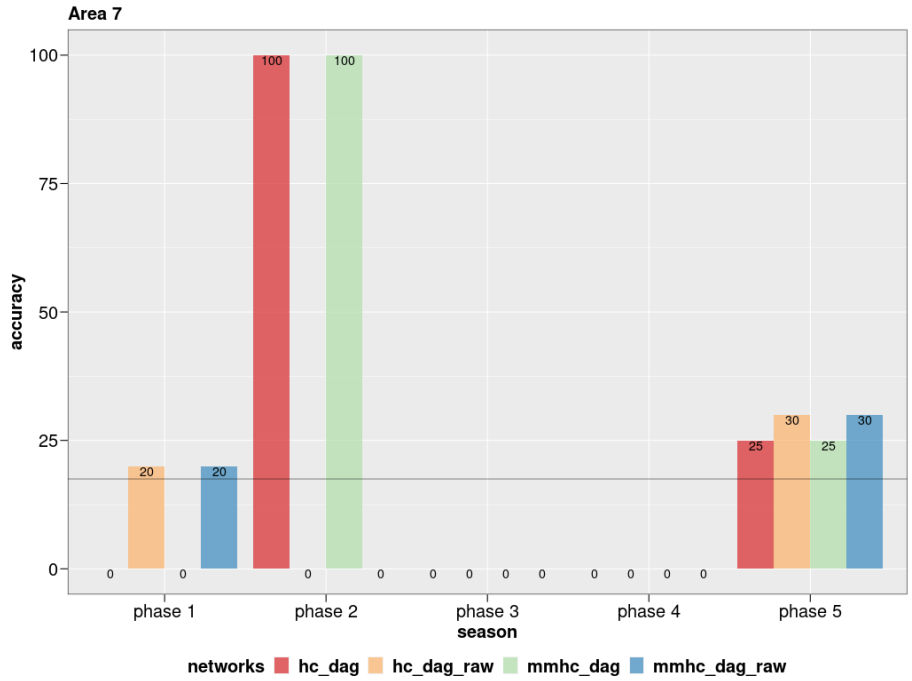
Fonte: Autor (2023)

Figura 59 – Acurácia - Redes estáticas - A6 - 10 colheitas



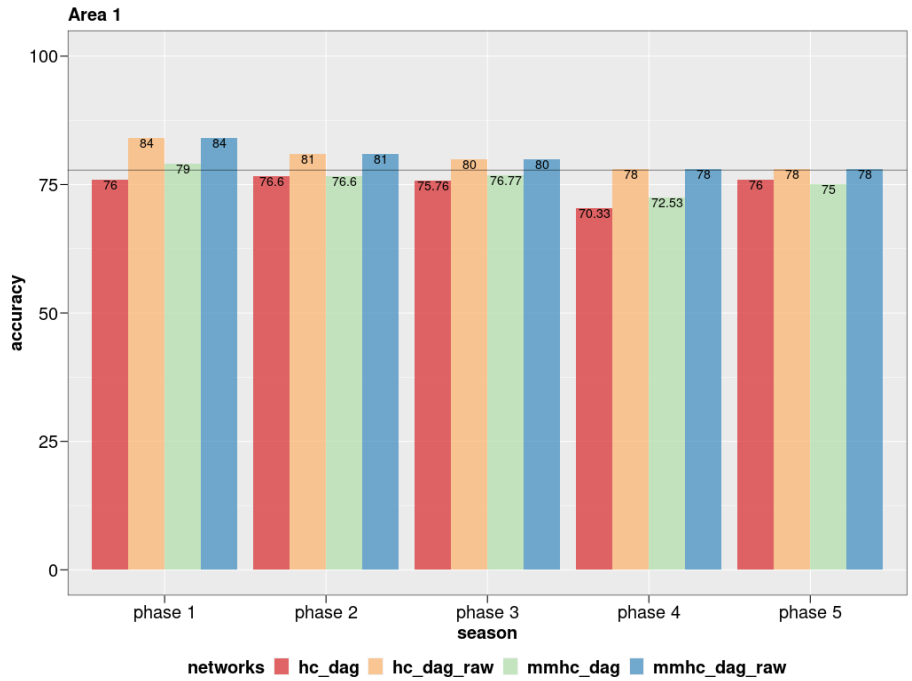
Fonte: Autor (2023)

Figura 60 – Acurácia - Redes estáticas - A7 - 10 colheitas



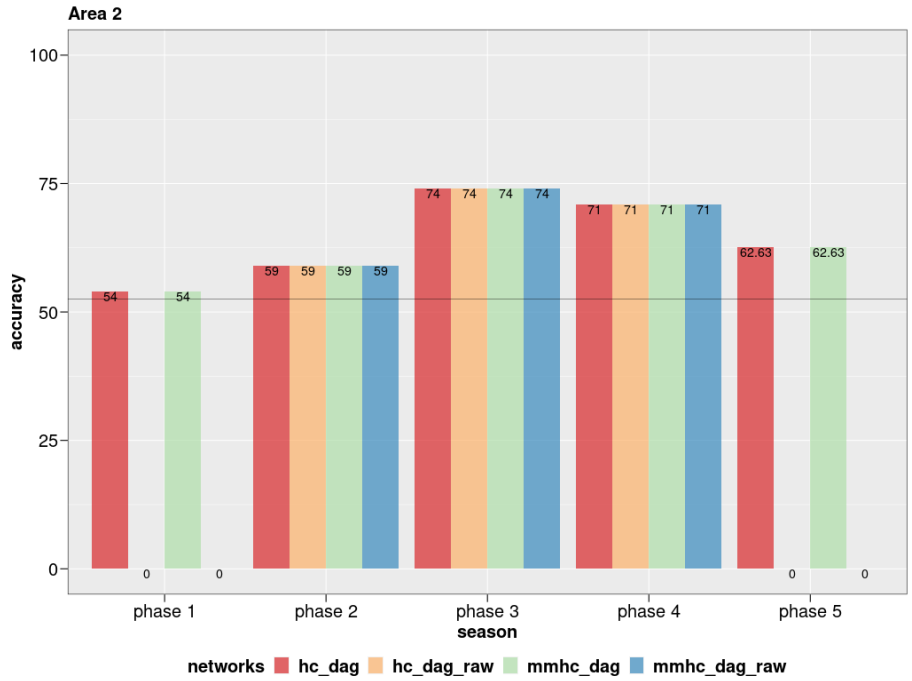
Fonte: Autor (2023)

Figura 61 – Acurácia - Redes estáticas - A1 - 100 colheitas



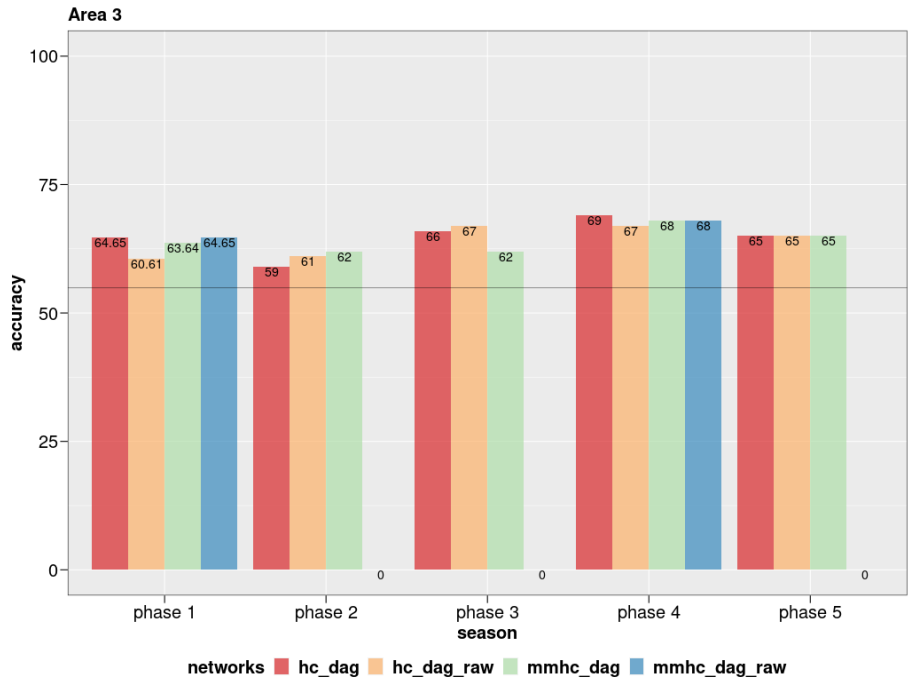
Fonte: Autor (2023)

Figura 62 – Acurácia - Redes estáticas - A2 - 100 colheitas



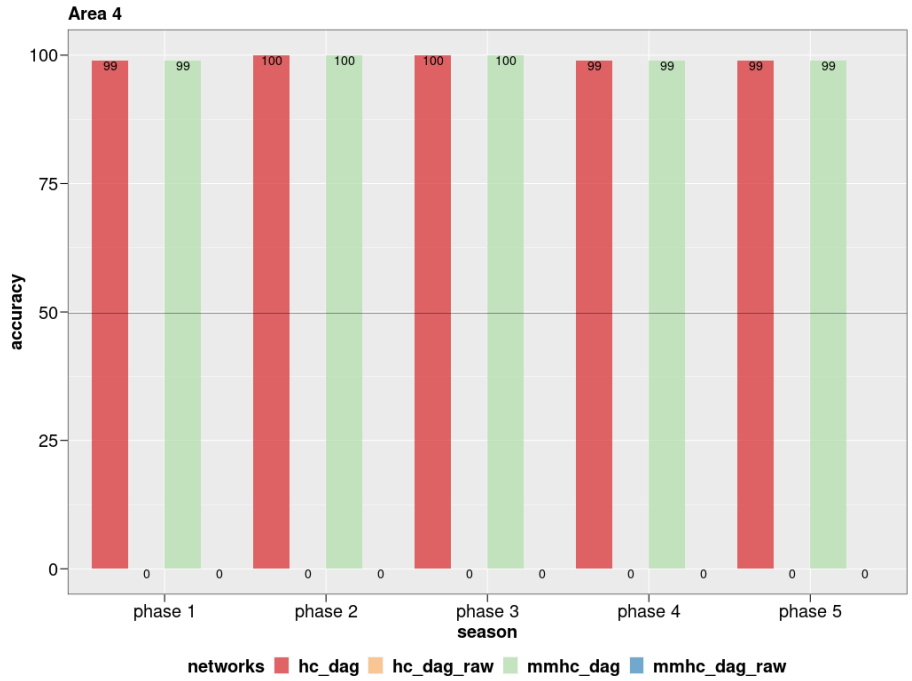
Fonte: Autor (2023)

Figura 63 – Acurácia - Redes estáticas - A3 - 100 colheitas



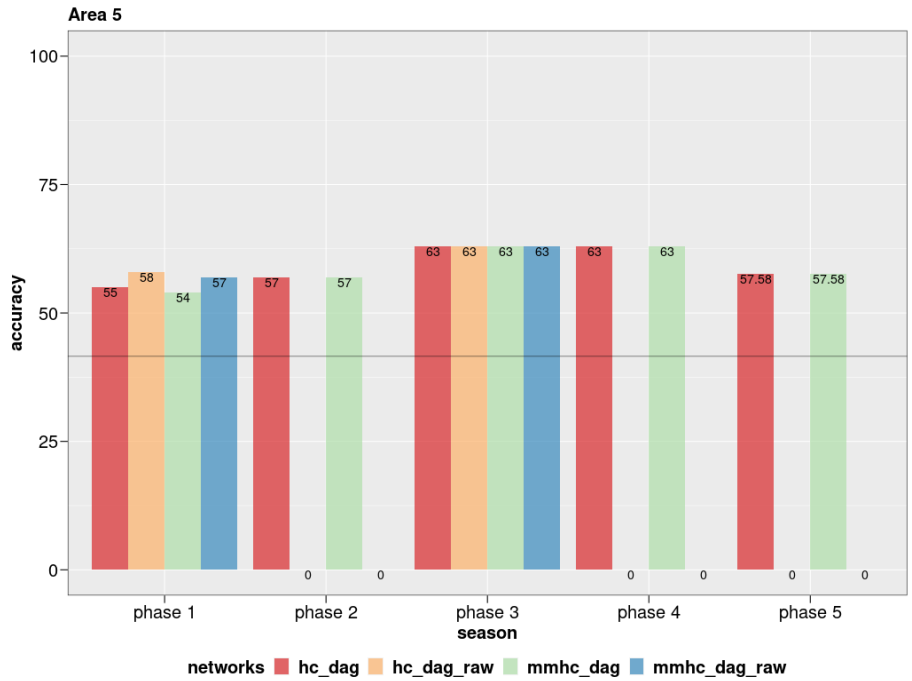
Fonte: Autor (2023)

Figura 64 – Acurácia - Redes estáticas - A4 - 100 colheitas



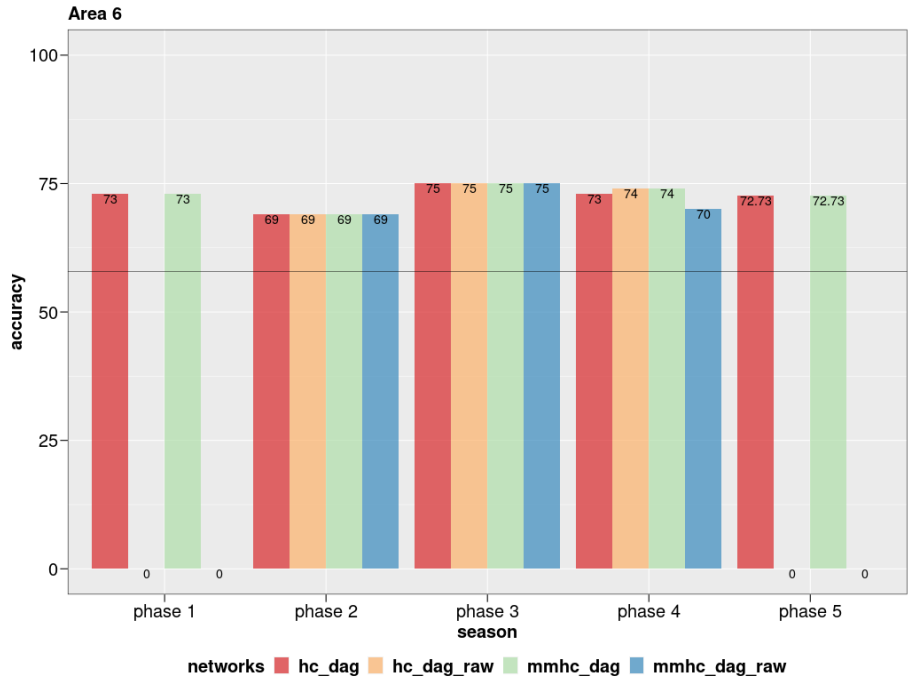
Fonte: Autor (2023)

Figura 65 – Acurácia - Redes estáticas - A5 - 100 colheitas



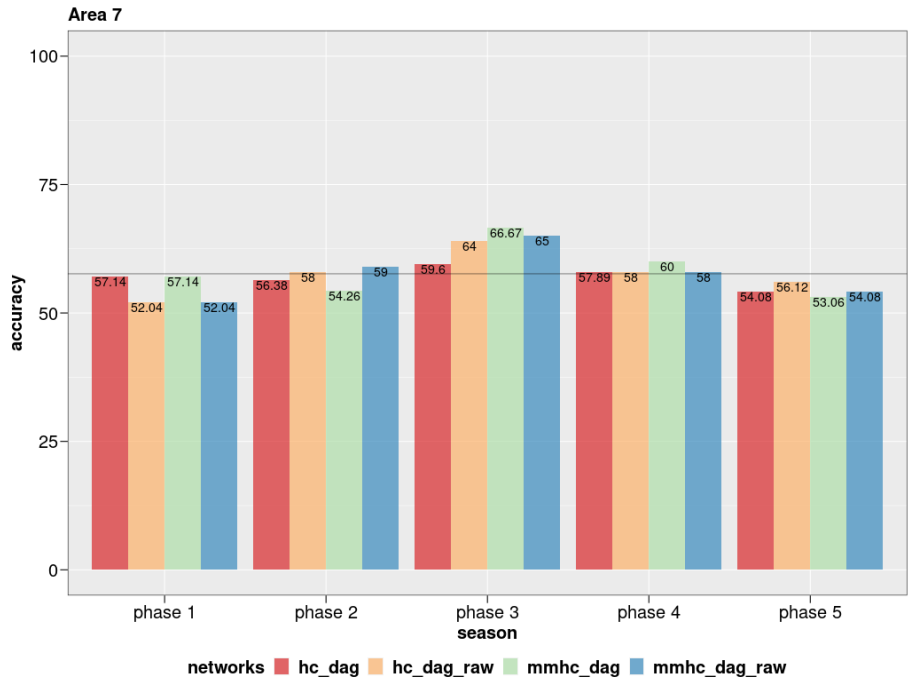
Fonte: Autor (2023)

Figura 66 – Acurácia - Redes estáticas - A6 - 100 colheitas



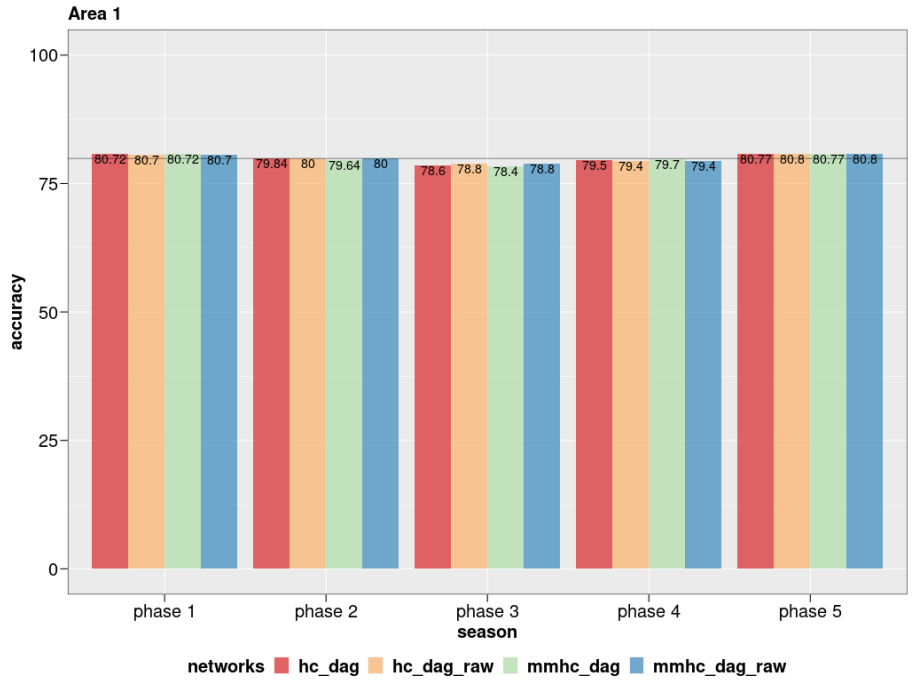
Fonte: Autor (2023)

Figura 67 – Acurácia - Redes estáticas - A7 - 100 colheitas



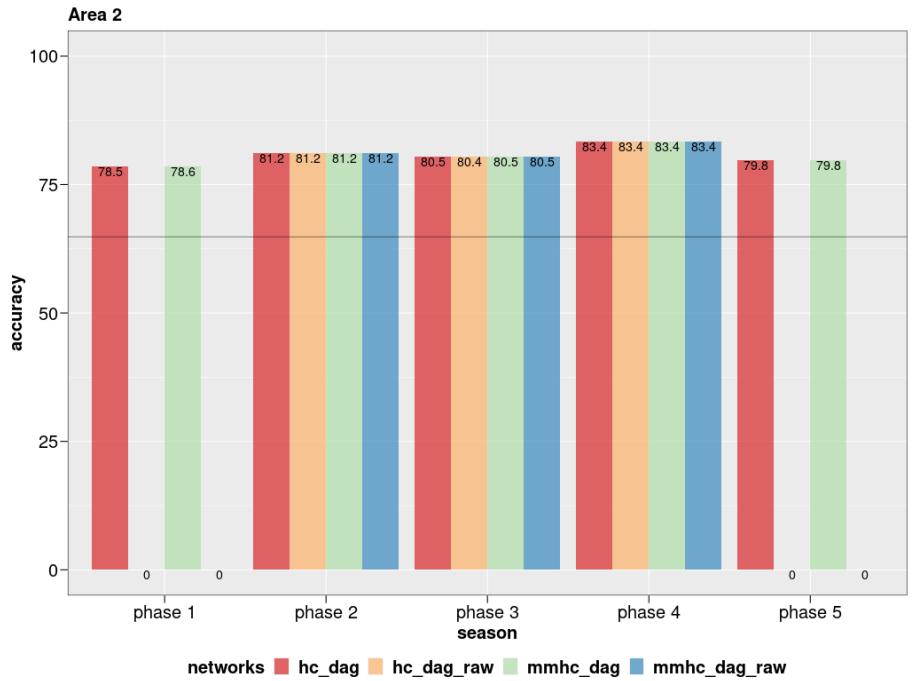
Fonte: Autor (2023)

Figura 68 – Acurácia - Redes estáticas - A1 - 1.000 colheitas



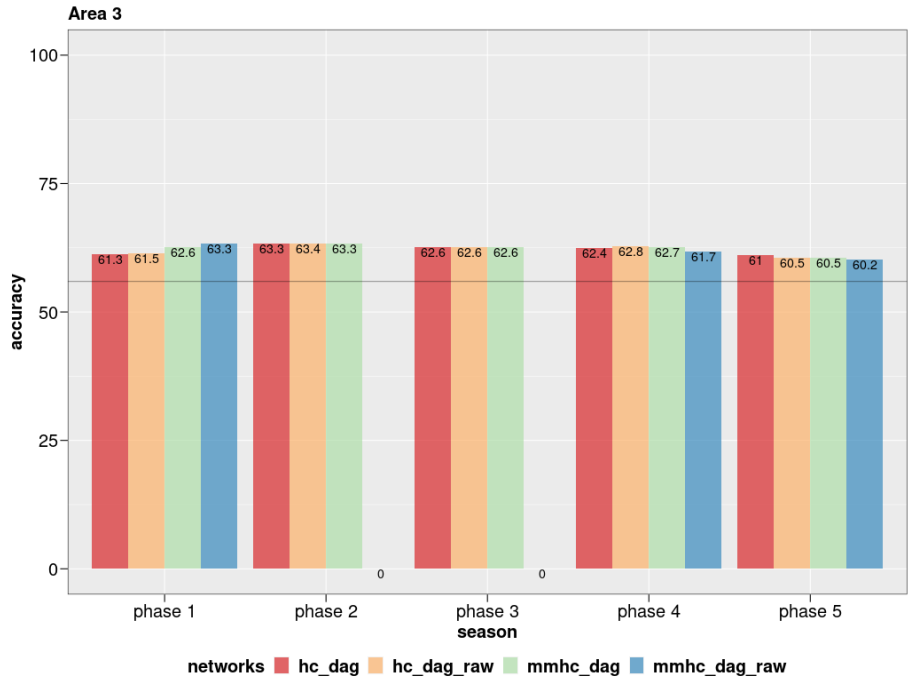
Fonte: Autor (2023)

Figura 69 – Acurácia - Redes estáticas - A2 - 1.000 colheitas



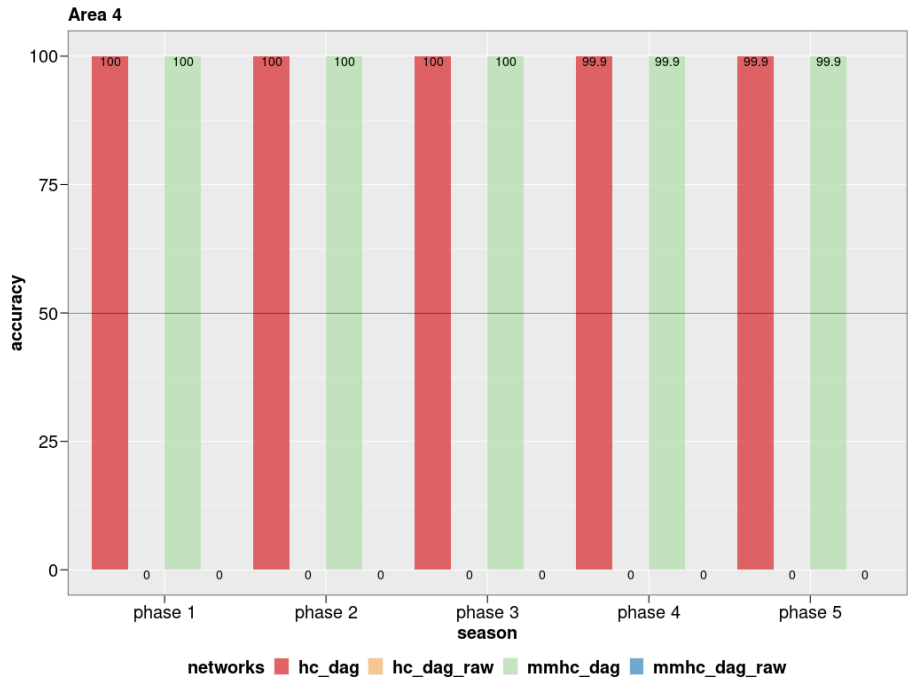
Fonte: Autor (2023)

Figura 70 – Acurácia - Redes estáticas - A3 - 1.000 colheitas



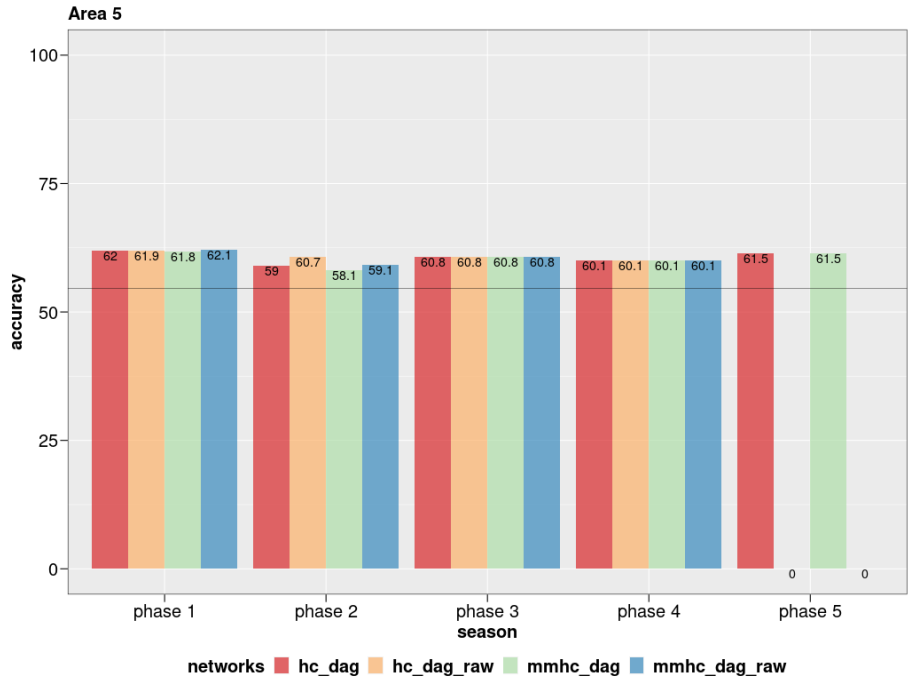
Fonte: Autor (2023)

Figura 71 – Acurácia - Redes estáticas - A4 - 1.000 colheitas



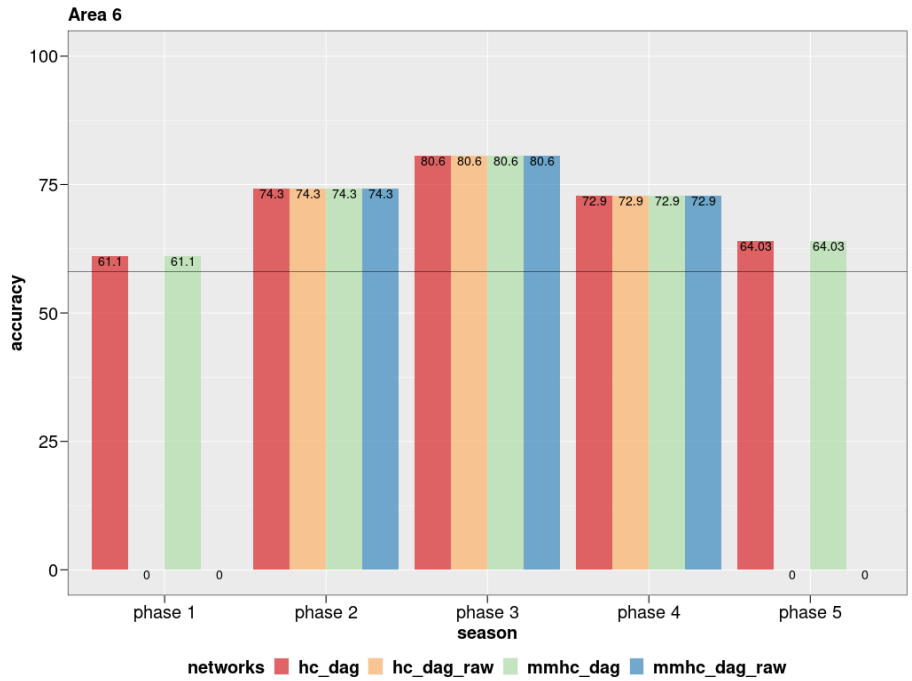
Fonte: Autor (2023)

Figura 72 – Acurácia - Redes estáticas - A5 - 1.000 colheitas



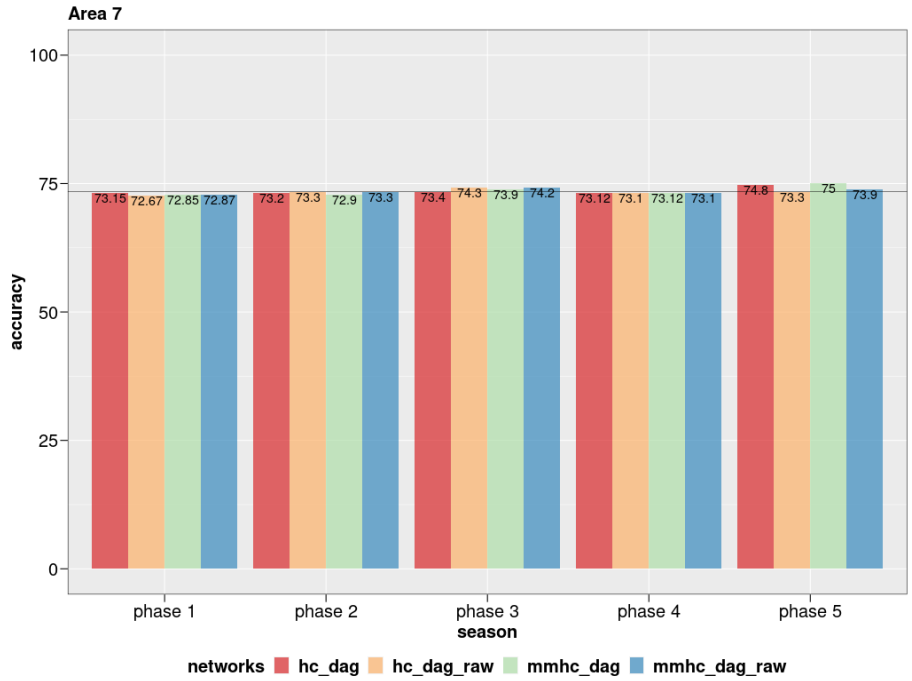
Fonte: Autor (2023)

Figura 73 – Acurácia - Redes estáticas - A6 - 1.000 colheitas



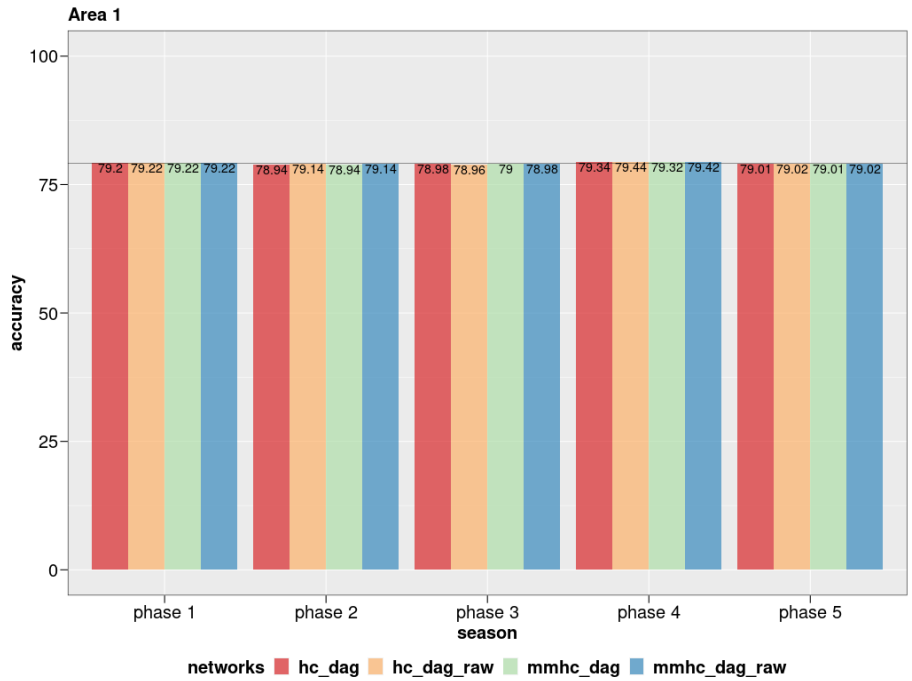
Fonte: Autor (2023)

Figura 74 – Acurácia - Redes estáticas - A7 - 1.000 colheitas



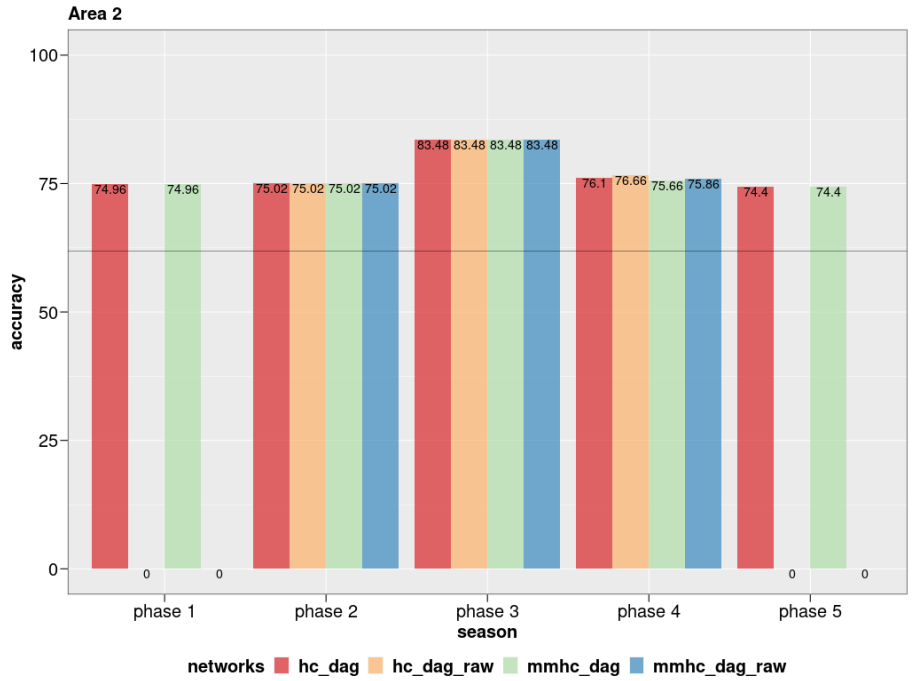
Fonte: Autor (2023)

Figura 75 – Acurácia - Redes estáticas - A1 - 5.000 colheitas



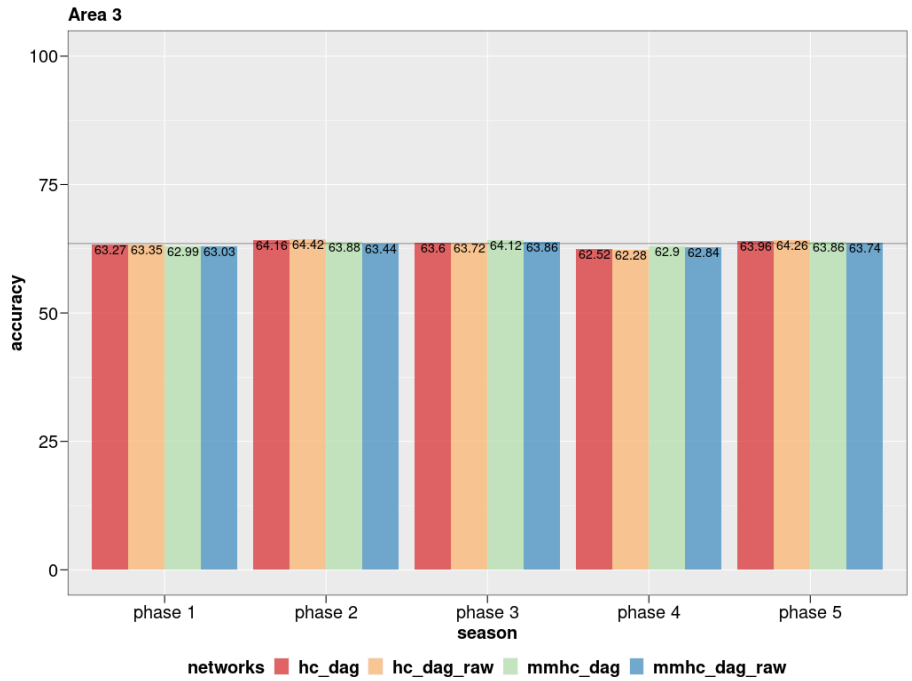
Fonte: Autor (2023)

Figura 76 – Acurácia - Redes estáticas - A2 - 5.000 colheitas



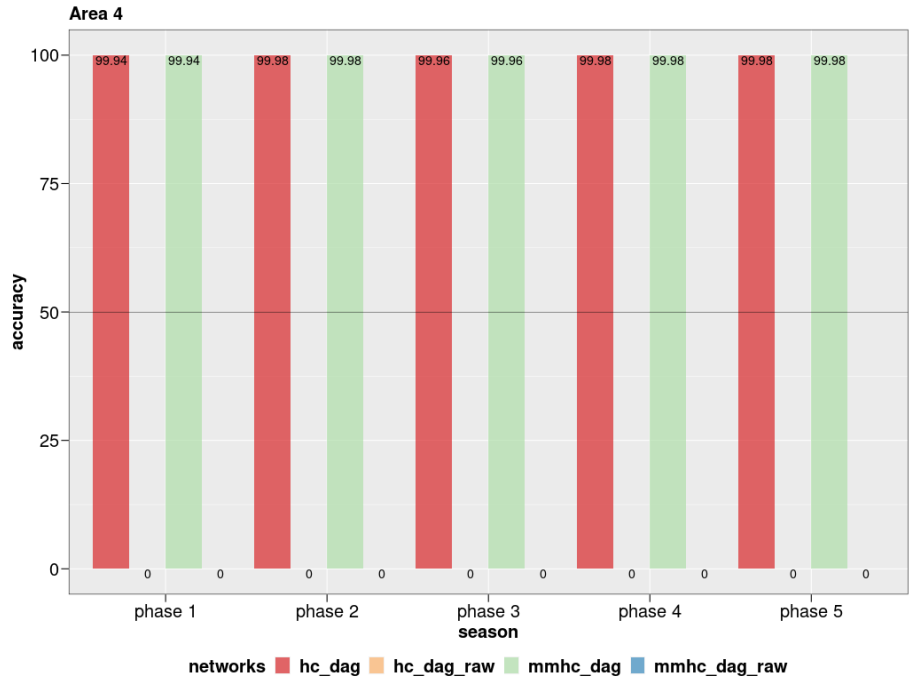
Fonte: Autor (2023)

Figura 77 – Acurácia - Redes estáticas - A3 - 5.000 colheitas



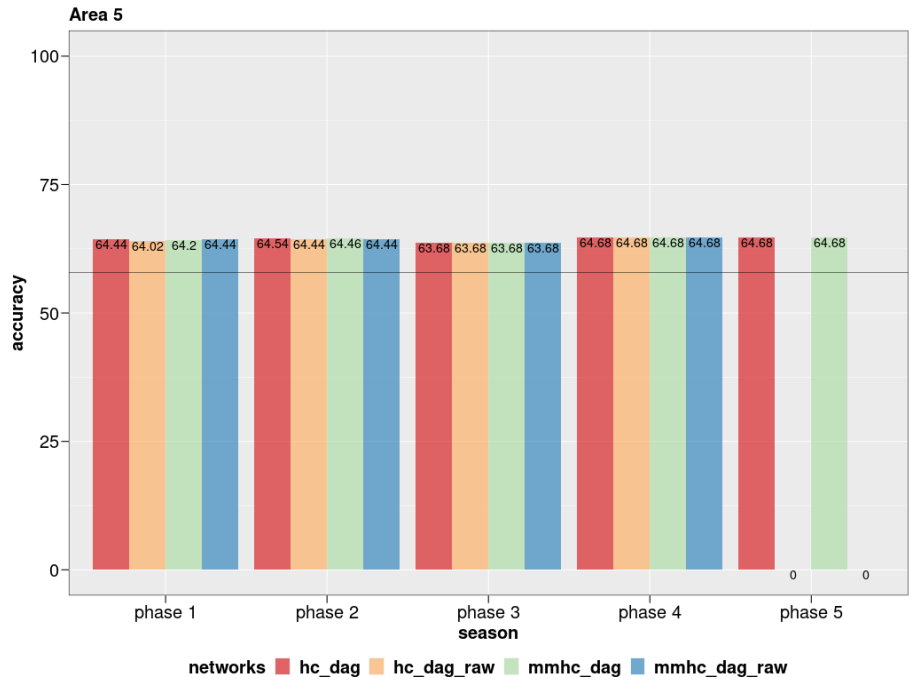
Fonte: Autor (2023)

Figura 78 – Acurácia - Redes estáticas - A4 - 5.000 colheitas



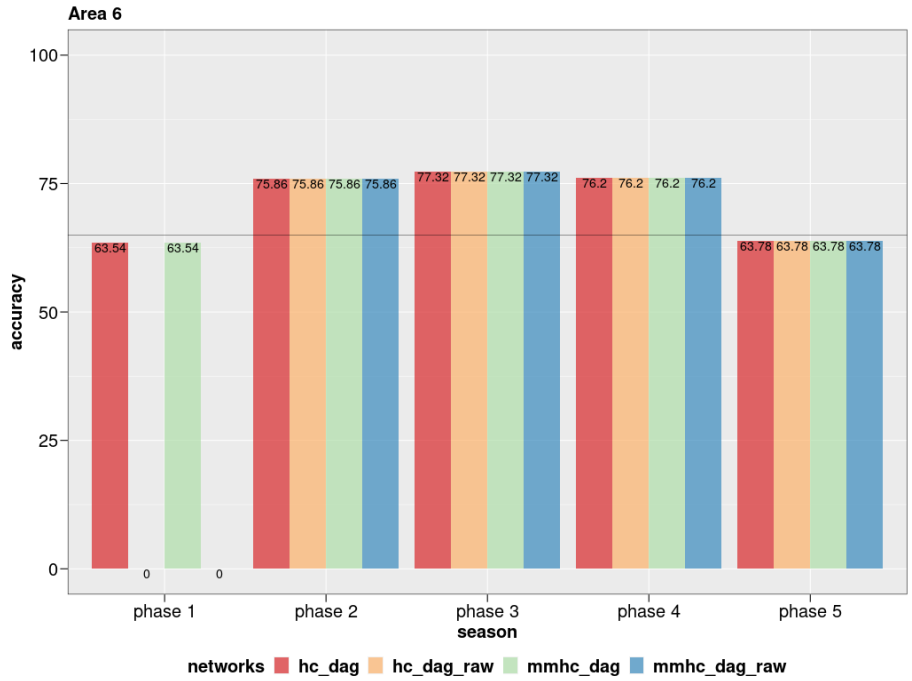
Fonte: Autor (2023)

Figura 79 – Acurácia - Redes estáticas - A5 - 5.000 colheitas



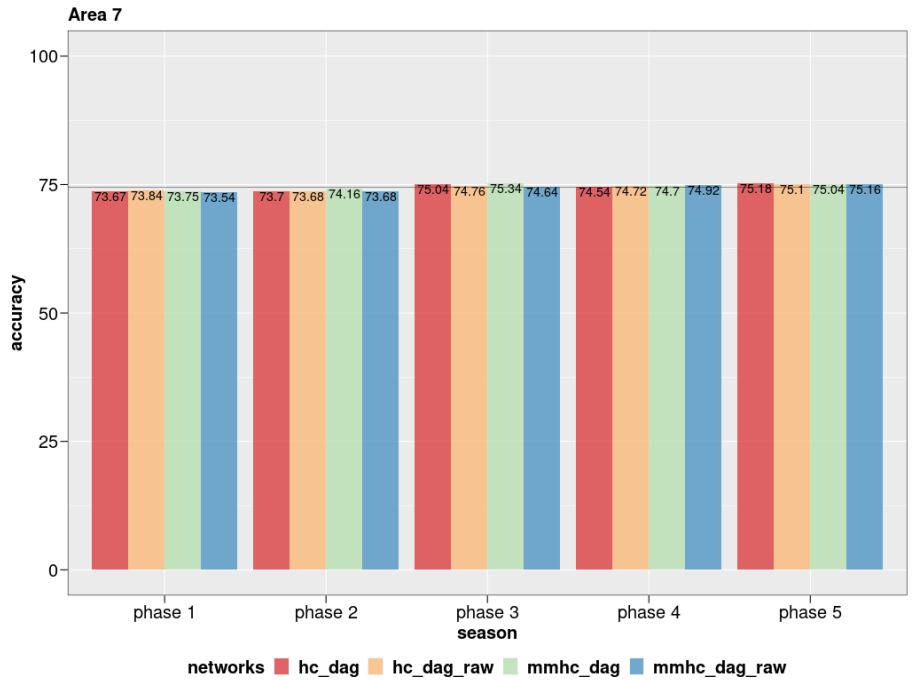
Fonte: Autor (2023)

Figura 80 – Acurácia - Redes estáticas - A6 - 5.000 colheitas



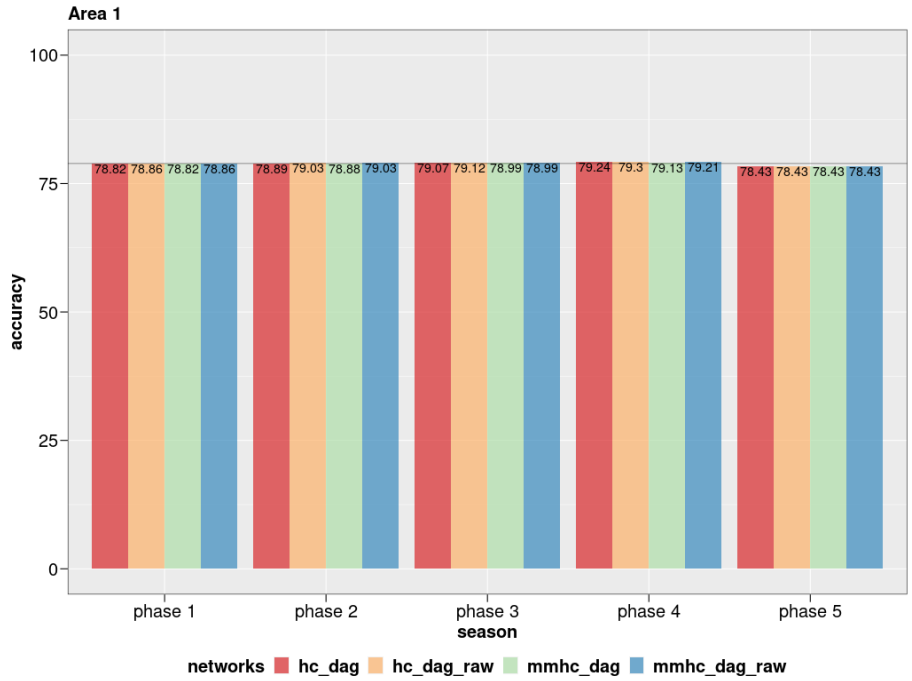
Fonte: Autor (2023)

Figura 81 – Acurácia - Redes estáticas - A7 - 5.000 colheitas



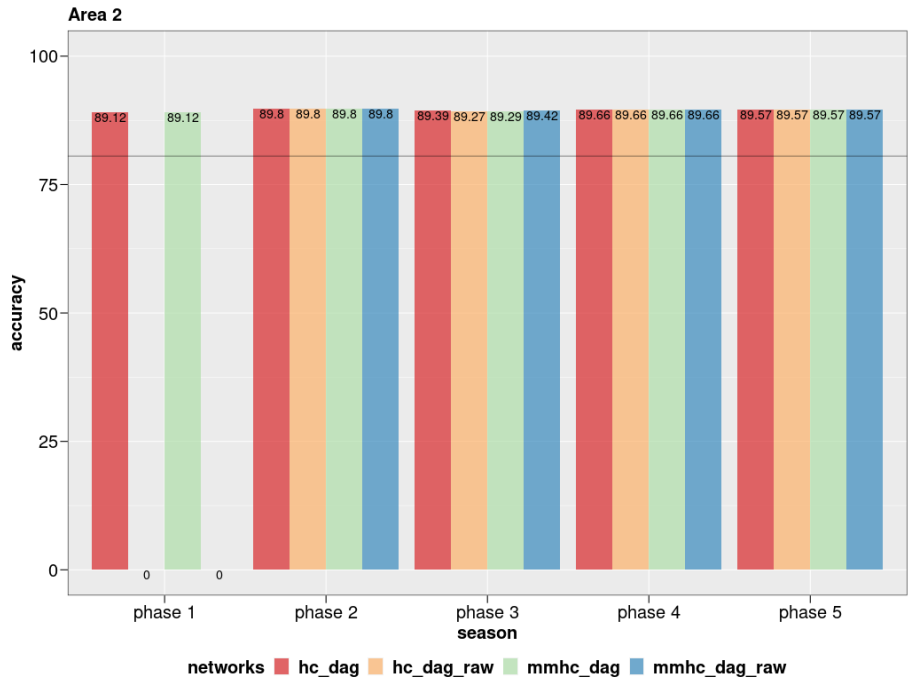
Fonte: Autor (2023)

Figura 82 – Acurácia - Redes estáticas - A1 - 10.000 colheitas



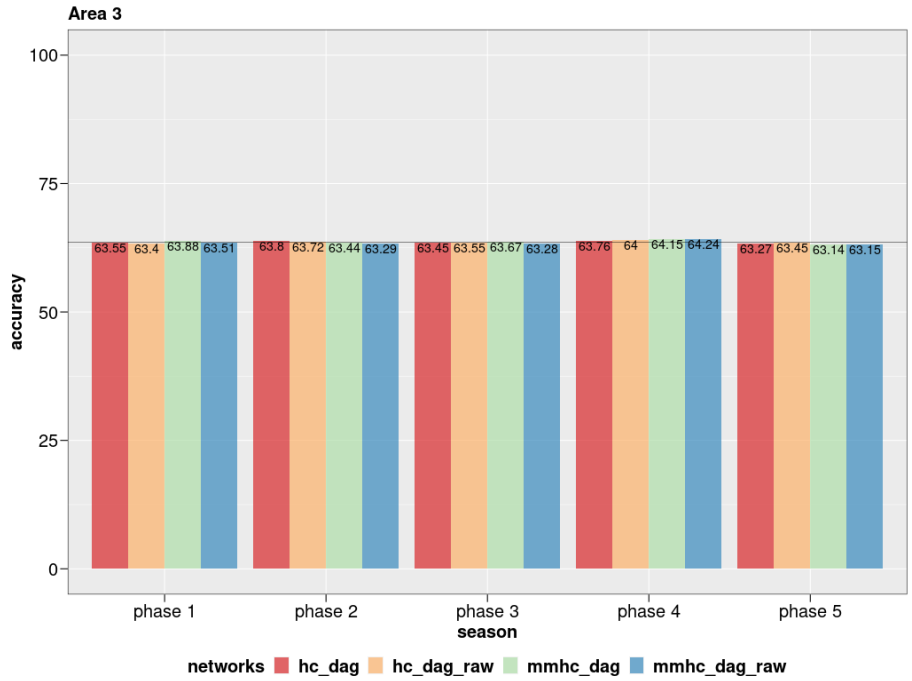
Fonte: Autor (2023)

Figura 83 – Acurácia - Redes estáticas - A2 - 10.000 colheitas



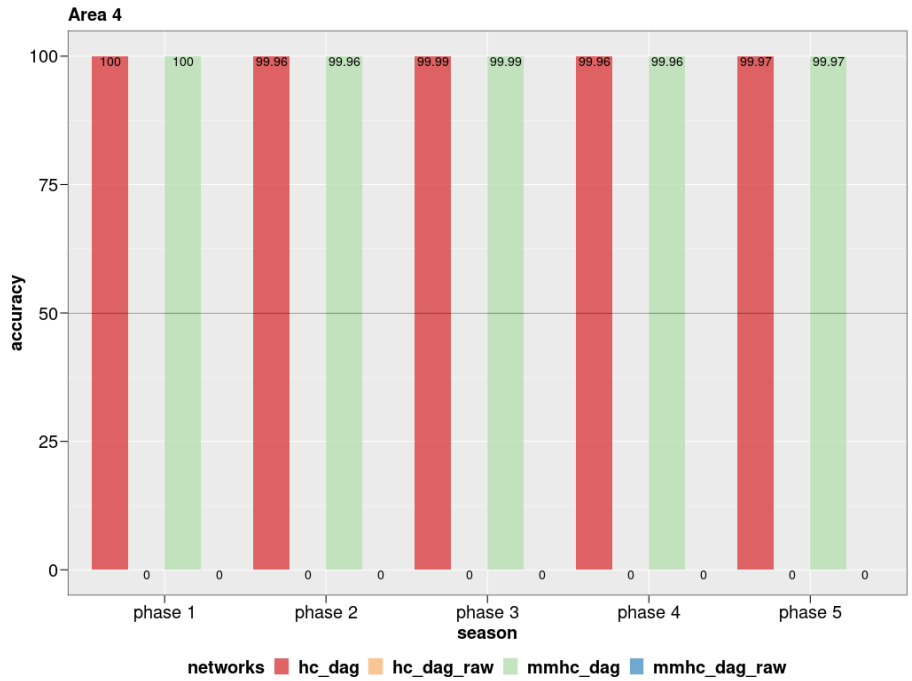
Fonte: Autor (2023)

Figura 84 – Acurácia - Redes estáticas - A3 - 10.000 colheitas



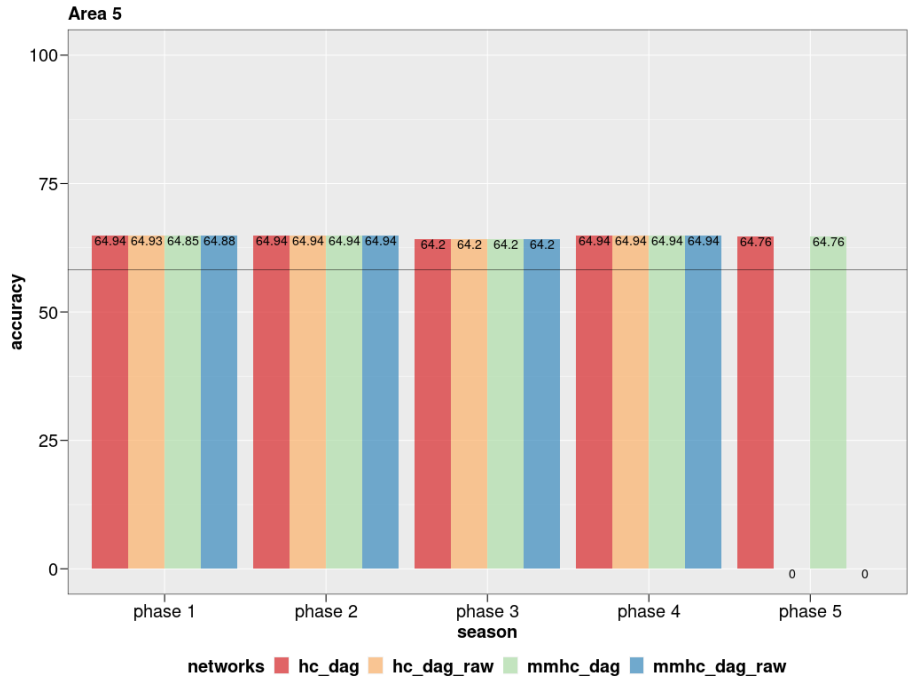
Fonte: Autor (2023)

Figura 85 – Acurácia - Redes estáticas - A4 - 10.000 colheitas



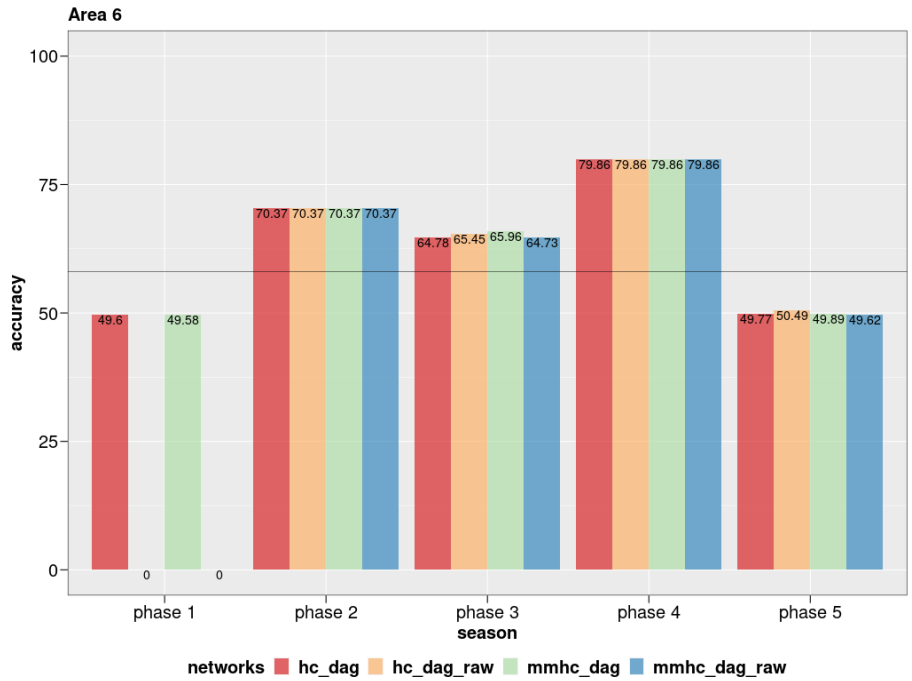
Fonte: Autor (2023)

Figura 86 – Acurácia - Redes estáticas - A5 - 10.000 colheitas



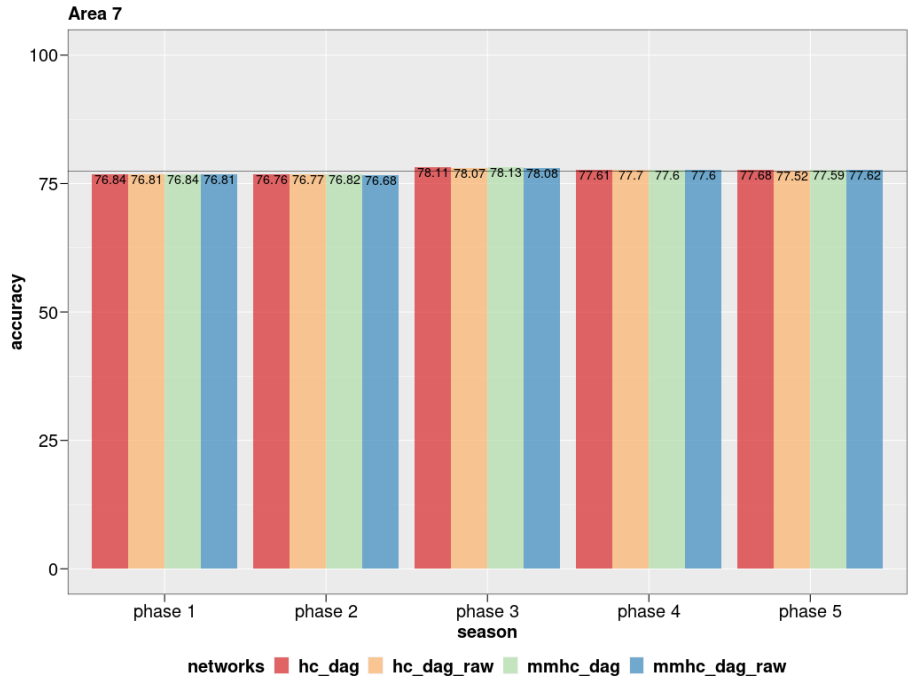
Fonte: Autor (2023)

Figura 87 – Acurácia - Redes estáticas - A6 - 10.000 colheitas



Fonte: Autor (2023)

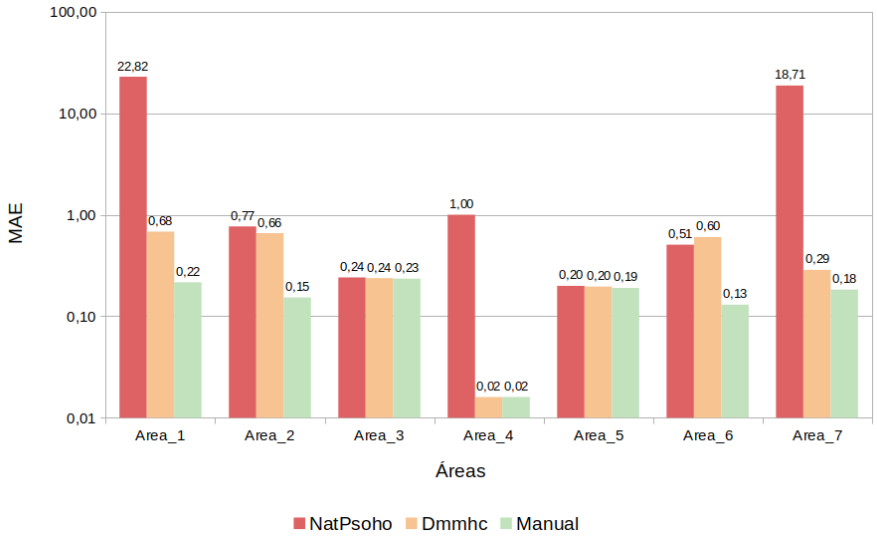
Figura 88 – Acurácia - Redes estáticas - A7 - 10.000 colheitas



Fonte: Autor (2023)

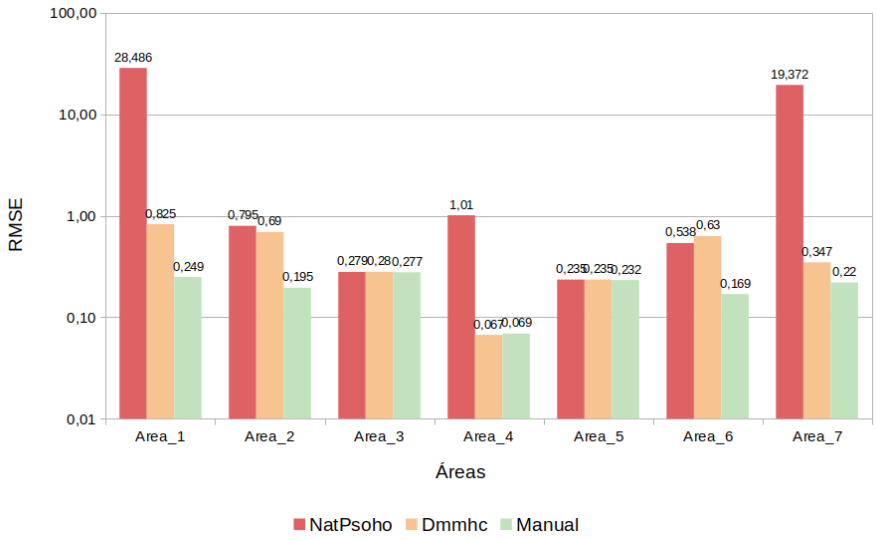
APÊNDICE B – GRÁFICOS – REDES DINÂMICAS

Figura 89 – Redes dinâmicas: MAE - 100 colheitas



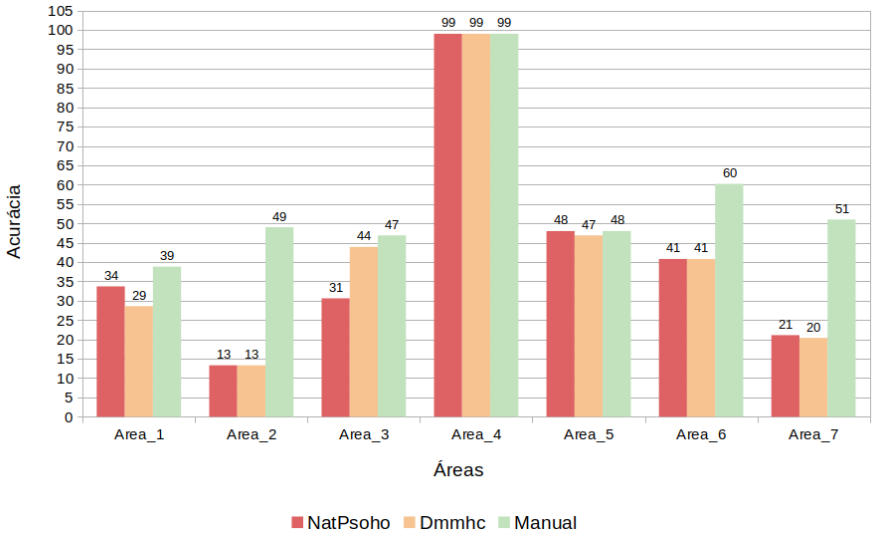
Fonte: Autor (2023)

Figura 90 – Redes dinâmicas: RMSE - 100 colheitas



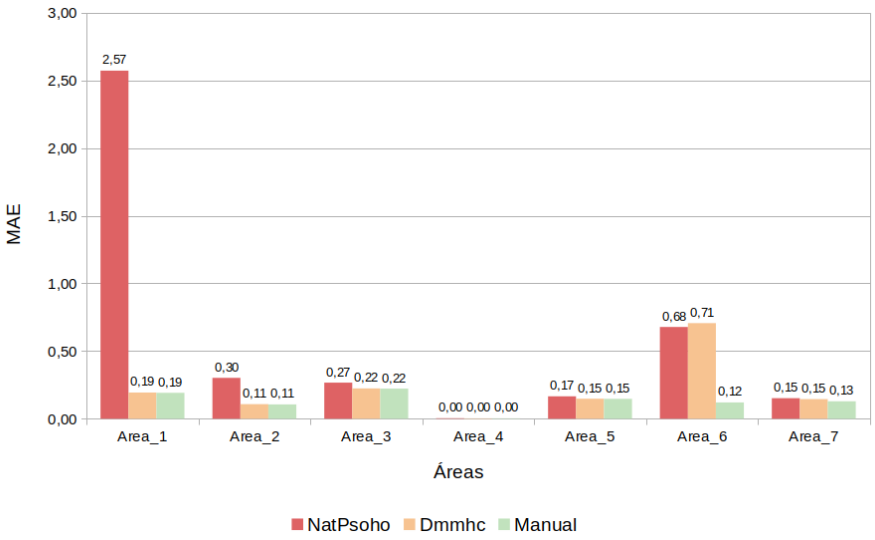
Fonte: Autor (2023)

Figura 91 – Redes dinâmicas: Acurácia - 100 colheitas



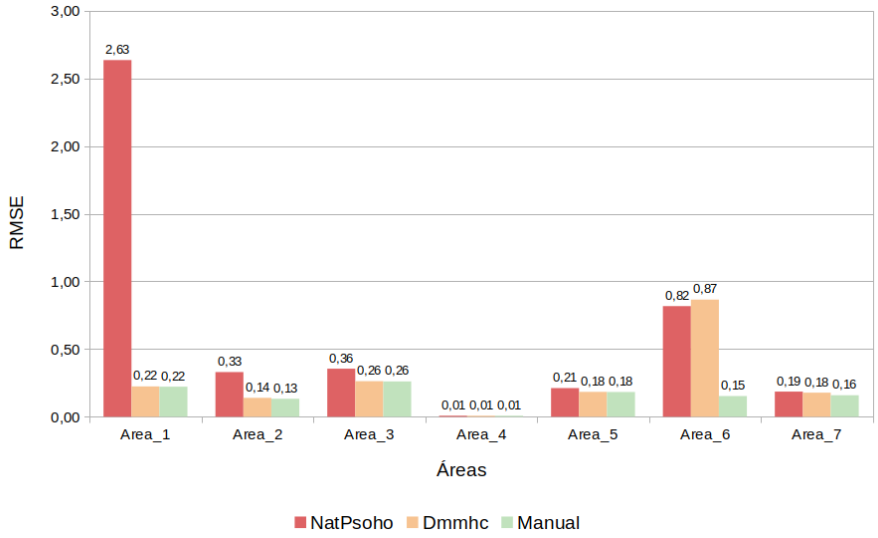
Fonte: Autor (2023)

Figura 92 – Redes dinâmicas: MAE - 1.000 colheitas



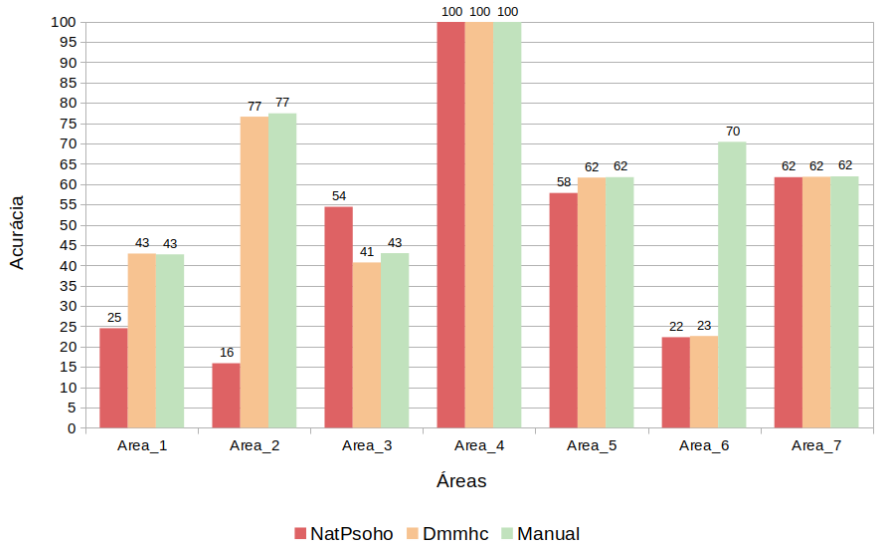
Fonte: Autor (2023)

Figura 93 – Redes dinâmicas: RMSE - 1.000 colheitas



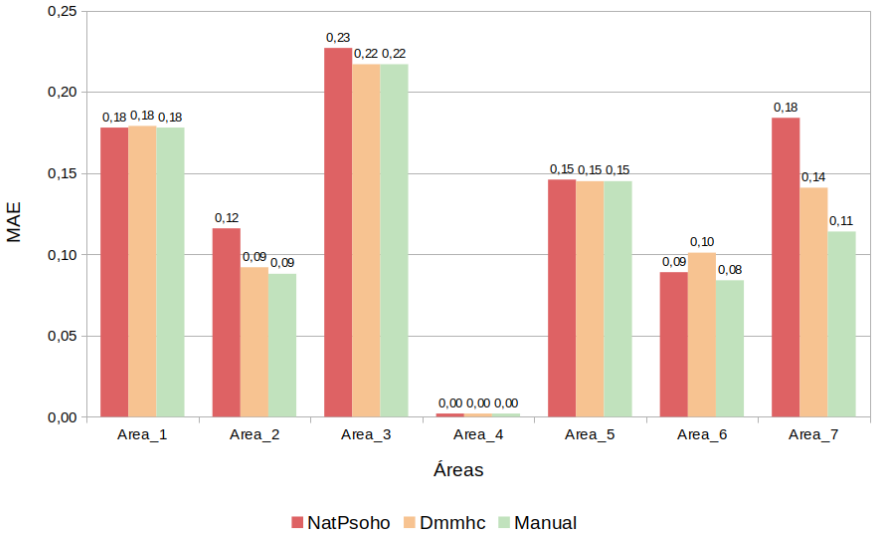
Fonte: Autor (2023)

Figura 94 – Redes dinâmicas: Acurácia - 1.000 colheitas



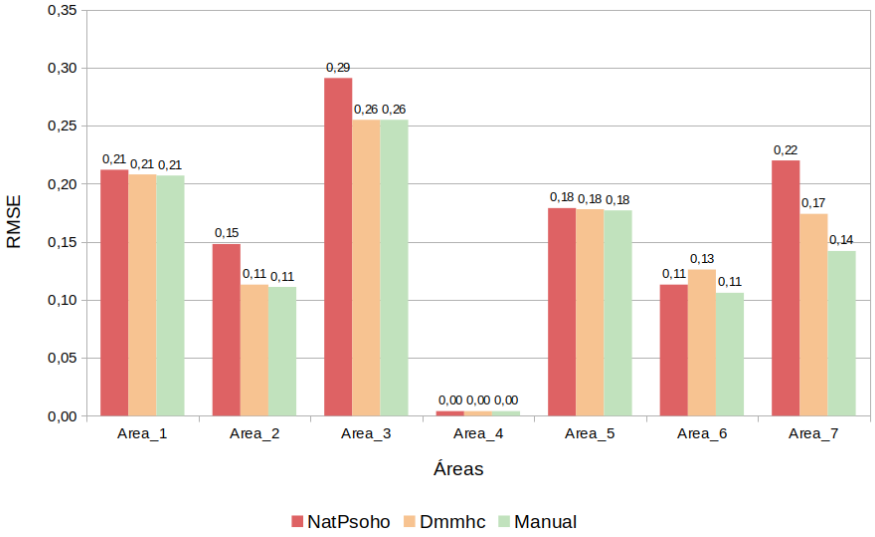
Fonte: Autor (2023)

Figura 95 – Redes dinâmicas: MAE - 5.000 colheitas



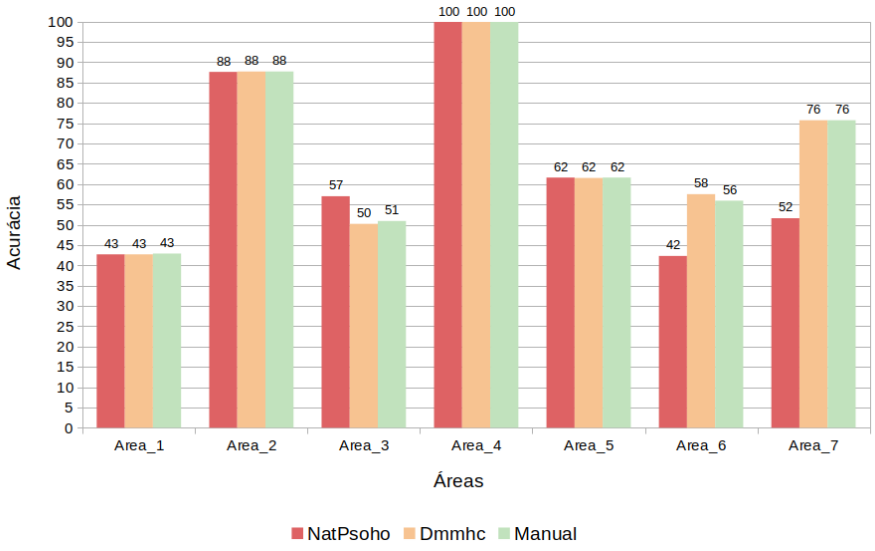
Fonte: Autor (2023)

Figura 96 – Redes dinâmicas: RMSE - 5.000 colheitas



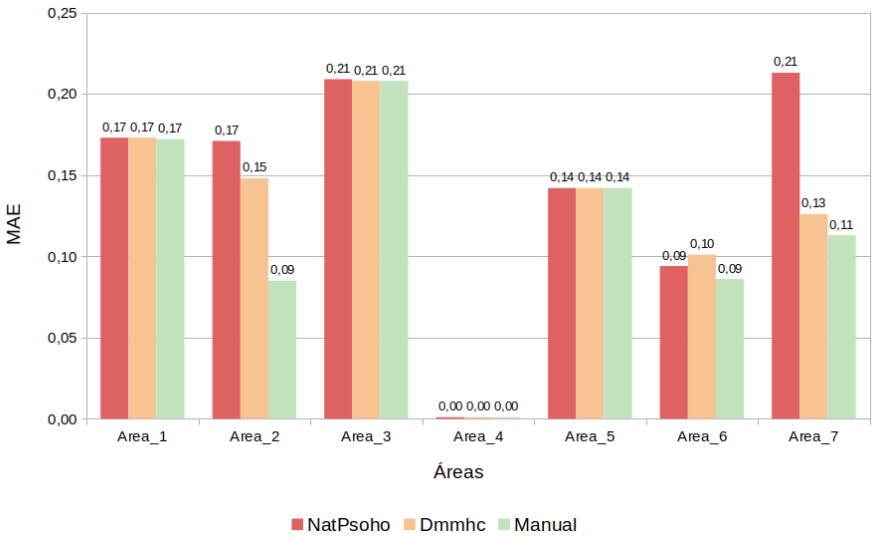
Fonte: Autor (2023)

Figura 97 – Redes dinâmicas: Acurácia - 5.000 colheitas



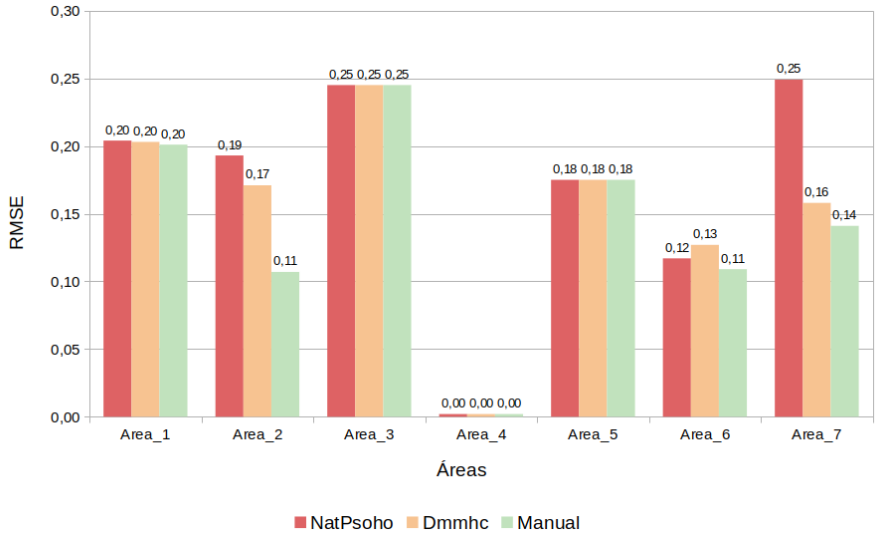
Fonte: Autor (2023)

Figura 98 – Redes dinâmicas: MAE - 10.000 colheitas



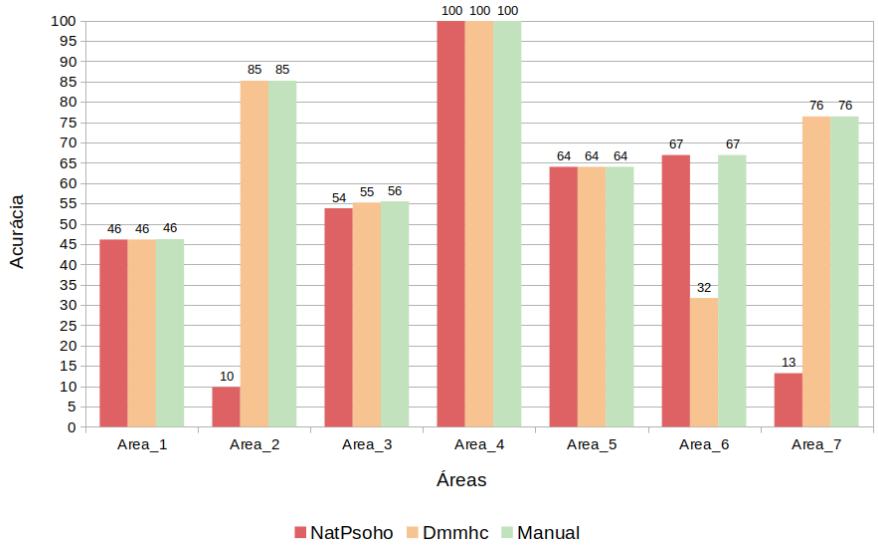
Fonte: Autor (2023)

Figura 99 – Redes dinâmicas: RMSE - 10.000 colheitas



Fonte: Autor (2023)

Figura 100 – Redes dinâmicas: Acurácia - 10.000 colheitas



Fonte: Autor (2023)

APÊNDICE C – CÓDIGO FONTE – PACOTE AGROBAYES

C.1 Código fonte – geração dos dados – data_gen.R

```
testRunDataGen <- function(nHarvests, nphases, nAreas, nVars, nClass, ...){
  #set.seed(101)
  file_path = "/data"
  if(nClass == 5){
    class_names <- c("L", "ML", "M", "MH", "H")
    }else
      if(nClass == 3 ){
        class_names <- c("L", "M", "H")
      }else{
        stop("incorrect parameter. Nclass must be 5 or 3")
      }

  # lists to store the data by area
  areas_list  = list()
  areas_list_disc  = list()
  areas_list  = testBuildSimulationData(nHarvests, nphases, nAreas, nVars)


  #for each area, create a list of dataframes
  #each dataframe has the data of a phenological phase

              #return(areas_list)

  for(area in 1:length(areas_list)){
    areas_list[area] = testCreateDataFrames(areas_list[area])
    #normalizes the harvest variable
      for(fase in 1:nphases){
        areas_list[[area]][[fase]][,4] <- f_minmax( areas_list[[area]][[fase]][,4])
      }
    }
  areas_list_disc = areas_list

  for(area in 1:length(areas_list)){#for each area
    for(fase in 1:length(areas_list[[1]])){#for each phase of the area
      areas_list_disc[[area]][[fase]] <-
        bnlearn ::discretize(areas_list[[area]][[fase]],
                             ordered = TRUE,
                             method = 'interval',
                             #method = 'quantile',
                             #method = 'hartemink', ibreaks = length(class_names)*2,
                             breaks = length(class_names))
      for(var in 1:dim(areas_list[[1]][[fase]])[2]){#para cada variável da área/fase
        levels(areas_list_disc[[area]][[fase]][,var]) <- class_names
      }
    }
  }
  return(c(areas_list, areas_list_disc))
}

testBuildSimulationData <- function (nHarvests, nPhases, nAreas = NULL, ...) {

  areasList <- list()
  if(is.null(nAreas)){
```

```
    # by default 7 areas are generated
    # each area has a relationship between variables and production

    areanames = c("Area_1", "Area_2", "Area_3",
                  "Area_4", "Area_5", "Area_6", "Area_7")
    areatype  = c(1,2,3,4,5,6,7)
    for (i in 1 : length(areanames)){
      areasList[[i]] <- c(areanames =  areanames[i], areatype  = areatype[i])
    }
  }else{
    # if the number of areas is defined, the types are randomly assigned
    for (i in 1 : nAreas){
      tempType = sample(1:6, 1)
      #tempType = 5
      tempName <- paste("Area", i, sep = "_")
      areasList[[i]] = c(areaName = tempName, areatype = tempType)
    }
  }

  # definition of variables by function testDefVars
  defProdVariables = testDefVars(...)

  completeSimValues = list()
  namesArea = array(dim = length(areasList))

  # For each area Calculate the values of the variables
  # and production by function testSetSimVarValues

  for (a in 1:length(areasList)) {

    #cat("Calculating area values ",a,"\n")

    completeSimValues[[a]] =
      testSetSimVarValues(nHarvests, areasList[[a]][2],defProdVariables,nPhases)
    namesArea[a] = paste("Area", a, "type", areasList[[a]][2], sep = "_")
  }
  names(completeSimValues) = namesArea

  return (completeSimValues)
}

testDefVars <- function (n_var = NULL, type_var = NULL, name_var = NULL){
  prodVariables = list()

  # If the number of variables is not defined, the default is the 3 below
  if(is.null(n_var)){
    # Definition of variables: name, minimum value, maximum value, type
    # type (up=1,osc=2,const=3)
    prodVariables[[1]] = c("PrecAcum",0,1,1)
    prodVariables[[2]] = c("Insol",0,1,2)
    prodVariables[[3]] = c("Compact",-1,1,3)

    return (prodVariables)
```

```
  }else
    if(!is.null(type_var) | !is.null(name_var)){
      # being informed the number of variables and their name and behavior
      # the variables are generated as follows:
      if(length(type_var)!= n_var | length(name_var)!= n_var){
        stop("incorrect parameters. If informed, types and names of
              variables must be informed for all variables")
      }else{
        for (i in 1 : n_var){
          # type (up=1,osc=2,const=3)
          if(type_var[i] == 3){
            prodVariables[[i]] = c(name_var[i], -1, 1, type_var[i])
          }else{
            prodVariables[[i]] = c(name_var[i], 0, 1, type_var[i])
          }
        }
      }
      return (prodVariables)
    }else{
      # Being informed the n of variables, without further details the generation
      # occurs randomly, as follows:
      for (i in 1 : n_var){
        tempType = sample(1:3, 1)
        tempName <- paste("x", i, sep = "_" )
        if(tempType == 3){
          prodVariables[[i]] = c(tempName, -1, 1, tempType)
        }else{
          prodVariables[[i]] = c(tempName, 0, 1, tempType)
        }
      }
      return (prodVariables)
    }
}

testSetSimVarValues <- function (nHarvests, areatype, prodvars, nPhases){
  crops <-list()
  names_crops <- array(dim = nHarvests)

  # defines variable names
  for(i in 1: nHarvests){
    # for each crop in the area
    # creates a matrix of 'number of phases' rows and
    # 'number of variables' columns

    phases <- matrix(nrow=nPhases, ncol= length(prodvars)+1)
    colnames(phases) <-  colnames(phases, do.NULL = FALSE, prefix = "X_")
    colnames(phases)[length(prodvars)+1] <-  "harvest"
    rownames(phases) <-  rownames(phases, do.NULL = FALSE, prefix = "phase_")
    crops[[i]] = phases
    names_crops[i] = paste("harvest", i, sep = "_")
  }
  names(crops) = names_crops
```

```
  # generates the values of the independent variables
  for (var in 1 : length(prodvars)){# for each variable
    for (harv in 1:nHarvests) {# for each crop
      random_value   = stats::runif(1,
                            min=as.numeric(prodvars[[var]][2]),
                            max=as.numeric(prodvars[[var]][3]))
      last_value_var = 0
      const_value    = random_value
      for (pha in 1:nPhases){ #pfor eachphase
        # type (up=1,osc=2,const=3)
        if (prodvars[[var]][4]==1){
          v_value = random_value + last_value_var
          last_value_var = v_value
        }else
          if (prodvars[[var]][4]==2){
            v_value = random_value
          }else
            if (prodvars[[var]][4]==3){
              # adding noise to variable type 3
                # to not interfere with network learning
              v_value = const_value + stats::rnorm(1, 0, .09)

            }

        crops[[harv]][pha,var] = v_value
        random_value = stats::runif(1,
                            min=as.numeric(prodvars[[var]][2]),
                            max=as.numeric(prodvars[[var]][3]))
      }#for pha in 1:nPhases
    }#for harv in 1:nHarvests
  }#for var in 1 : length(prodvars)

# Normalizing (between 0 and 1 or 1- and 1, depending on the type of variable)
# of the variable data before calculating the production values
  for(h in 1 : nHarvests){
    for(v in 1 : length(prodvars)){
      if(max(abs(crops[[h]][,v])) > 1){
        crops[[h]][,v] = f_minmax(crops[[h]][,v])

        # fix extreme values 0 and 1 by adding/subtracting
        # a random value in the 3rd place after the decimal point
        crops[[h]][which.max(crops[[h]][,v]),v] =
          max(crops[[h]][,v]) - stats::runif(1)/100
        crops[[h]][which.min(crops[[h]][,v]),v] =
          min(crops[[h]][,v]) + stats::runif(1)/100
      }
    }
  }

  # calculates the production value based on
  # the arbitrated relations with the variables
```

```
##### POR DEFAULT CONSIDERA-SE 3 VARIÁVES:
if(length(prodvars)==3){
  #  print('AQUI')

  # Area 1: production weight varies linearly with the values of all
  # variables in the first two phenological phases
  # Prod = (X11 + X12 + X21 + X22 + X31 + X32)
  if (areatype == 1) {

    for (h in 1: nHarvests){
      crops[[h]][,length(prodvars)+1] =
        sum(crops[[h]][,1:length(prodvars)])
    }
  }

  # Area 2: production weight varies with the square of X1
  # Prod = (X11^2 + X12^2 + X13^2 + X14^2 + X15^2)
  else if (areatype == 2) {
    for(h in 1:nHarvests) {
      crops[[h]][,length(prodvars)+1] = sum((crops[[h]][,1])^2)
    }
  }

  # Area 3: production weight varies with the square of X3
  else if (areatype == 3) {
    for(h in 1:nHarvests) {
      crops[[h]][,length(prodvars)+1] = sum((crops[[h]][,3])^2)
    }
  }

  # Area 4: the production weight is inversely proportional to the sum of X1 and X3
  # Prod = 1/(X11+X13) + 1/(...)
  else if (areatype == 4) {
    for(h in 1:nHarvests) {
      producaoPhase = 0
      for(p in 1:nPhases){
        temp = 1/(crops[[h]][p,1]+crops[[h]][p,3])
        producaoPhase =  producaoPhase + temp
      }
      crops[[h]][,length(prodvars)+1] =  producaoPhase
    }
  }

  # Area 5: the production weight decreases
  # with a weighting of the X2 values:
  # Prod = 1*X21 + 0.8*X22 + 0.6*X23 + 0.4*X24 + 0.2*X25
  else if (areatype == 5) {
    for(h in 1:nHarvests) {
      producaoPhase = 0
      chunk = 100/nPhases
      for(p in 1:nPhases){
        temp = (crops[[h]][p,2])*(100-(chunk*p-1))/100
        producaoPhase =  producaoPhase + temp
```

```
      }
      crops[[h]][,length(prodvars)+1] =  producaoPhase
    }
  }

  # Area 6: the weight of production grows
  # with a weighting of the values of X1:
  # Prod = 0.2*X11 + 0.4*X12 + 0.6*X13 + 0.8*X14 + 1*X15
  else if (areatype == 6){
    for(h in 1:nHarvests) {
      producaoPhase = 0
      chunk = 100/nPhases
      for(p in 1:nPhases){
        temp = ((crops[[h]][p,1])*(chunk*p))/100
        producaoPhase =  producaoPhase + temp
      }
      crops[[h]][,length(prodvars)+1] =  producaoPhase
    }
  }else if (areatype == 7){
  # Prod =
      #   50*X1_1 + 30*X2_1 + X3_1 +
      #   40*X1_2 + 20*X2_2 + X3_2 +
      #   30*X1_3 + 30*X2_3 + 5*X3_3 +
      #   20*X1_4 + 40*X2_4 + 10*X3_4 +
      #   10*X1_3 + 50*X2_5 + 20*X3_5
    #    crops[[harvest]][phase,var])
    for(h in 1:nHarvests){
      crops[[h]][,length(prodvars)+1] =
    (50*(crops[[h]][1,1])) + (30*(crops[[h]][1,2])) + (crops[[h]][1,3]) +
    (40*(crops[[h]][2,1])) + (20*(crops[[h]][2,2])) + (crops[[h]][2,3]) +
    (30*(crops[[h]][3,1])) + (30*(crops[[h]][3,2])) + (5*(crops[[h]][3,3]))+
    (20*(crops[[h]][4,1])) + (40*(crops[[h]][4,2])) + (10*(crops[[h]][4,3]))+
    (10*(crops[[h]][5,1])) + (50*(crops[[h]][5,2])) + (20*(crops[[h]][5,3]))
    }
  }

}else{# if more than 3 variables are defined

  # Area 1: the production weight varies linearly with the values
  # of all variables in the first two phenological phases
  # Prod = (X11 + X12 + X21 + X22 + X31 + X32)

  if (areatype == 1) {
    for (h in 1: nHarvests){
      crops[[h]][,length(prodvars)+1] = sum(crops[[h]][,1:length(prodvars)])
    }
  }

  # Area 2: the production weight varies with the square of the odd variables
  # Prod = sum(Xi1^2 + Xi2^2 + Xi3^2 + Xi4^2 + Xi5^2) | i=(2k+1), k(0:infinity)
  else if (areatype == 2) {
    for(h in 1:nHarvests) {
      producaoVar = 0
```

```
    for (v in 1 : length(prodvars)){
      if(v %% 2 == 0){next}
      temp = sum((crops[[h]][,v])^2)
      producaoVar =  producaoVar + temp
    }
    crops[[h]][,length(prodvars)+1] = producaoVar
  }
}

# Area 3: the production weight varies with the square of the even variables
# Prod = sum(Xi1^2 + Xi2^2 + Xi3^2 + Xi4^2 + Xi5^2) | i=(2k), k(0:infinity)
else if (areatype == 3){
  for(h in 1:nHarvests) {
    producaoVar = 0
    for (v in 1 : length(prodvars)){
      if(v %% 2 != 0){next}
      temp = sum((crops[[h]][,v])^2)
      producaoVar =  producaoVar + temp
    }
    crops[[h]][,length(prodvars)+1] = producaoVar
  }
}

# Area 4: the production weight is inversely proportional to the sum of
# the odd variables
# Prod = 1/(X11+X13) + 1/(...)
else if (areatype == 4) {
  for(h in 1:nHarvests) {
    producaoPhase = 0
    for(p in 1:nPhases){
      denominTemp = 0
      for (v in 1 : length(prodvars)){
        if(v %% 2 == 0){next}
        temp = sum(crops[[h]][p,v])
        denominTemp =  denominTemp + temp
      }
      temp2 = 1/denominTemp
      producaoPhase =  producaoPhase + temp2
    }
    crops[[h]][,length(prodvars)+1] =  producaoPhase
  }
}

# Area 5: the production weight decreases with a weighting
# of the sum of the odd variables
# Prod = 1*X21 + 0.8*X22 + 0.6*X23 + 0.4*X24 + 0.2*X25
else if (areatype == 5) {
  for(h in 1:nHarvests) {
    for(p in 1:nPhases){
      producaoVar = 0
      for (v in 1 : length(prodvars)){
        if(v %% 2 == 0){next}
        temp = sum((crops[[h]][p,v]))
```

```
            producaoVar =  producaoVar + temp
          }
          producaoPhase = 0
          chunk = 100/nPhases
          temp2 = producaoVar*(100-(chunk*p-1))/100
          producaoPhase =  producaoPhase + temp2
        }
        crops[[h]][,length(prodvars)+1] =  producaoPhase
      }
    }

    # Area 6: the production weight grows with a weighting
    # of the values of the even variables:
    # Prod = 0.2*X11 + 0.4*X12 + 0.6*X13 + 0.8*X14 + 1*X15
    else if (areatype == 6) {
      for(h in 1:nHarvests) {
        for(p in 1:nPhases){
          producaoVar = 0
          for (v in 1 : length(prodvars)){
            if(v %% 2 != 0){next}
            temp = sum((crops[[h]][p,v]))
            producaoVar =  producaoVar + temp
          }
          producaoPhase = 0
          chunk = 100/nPhases
          temp2 = producaoVar*(100-(chunk*p-1))/100
          producaoPhase =  producaoPhase + temp2
        }
        crops[[h]][,length(prodvars)+1] =  producaoPhase
      }
    }
  }#fim else

  return(crops)
}

testCreateDataFrames <- function (data){
  phase <- list()
  area <- list()
  for(k in 1:length(data)){
    for(j in 1:length(data[[1]][[1]][,1])){
      area_phase <- matrix(0, nrow=length(data[[1]]),
                           ncol=length(data[[1]][[1]][1,]))
      for(i in 1:length(data[[1]])){
        area_phase[i,] <- data[[1]][[i]][j,]
      }
      phase[[j]] <- data.frame(area_phase)
      colnames(phase[[j]]) <- colnames(data[[1]][[1]])
      rownames(phase[[j]]) <- rownames(phase[[j]])
    }
    area[[k]] <- phase
  }
  return(area)
```

```
}
```

C.2 Código fonte – cria redes estáticas - usuário – create_bn.R

```
runNetworks <- function(arealist, blacklist, whitelist){
  area <- list()
  out <- data.frame()
  name <- array()
  iname=1
  for(phase in 1:length(arealist)){
    area[[phase]] <- createNetworks(arealist[[phase]], blacklist, whitelist)

    out <- dplyr::bind_rows(out, area[[phase]][3])
    name[iname] <- paste("phase", phase, "hc_dag")
    iname = iname+1

    out <- dplyr::bind_rows(out, area[[phase]][6])
    name[iname] <- paste("phase", phase, "hc_dag_raw")
    iname = iname+1

    out <- dplyr::bind_rows(out, area[[phase]][9])
    name[iname] <- paste("phase", phase, "mmhc_dag")
    iname = iname+1

    out <- dplyr::bind_rows(out, area[[phase]][12])
    name[iname] <- paste("phase", phase, "mmhc_dag_raw")
    iname = iname+1
  }
  row.names(out$eval)<-name
  return(out$eval)
}

createNetworks <- function (areaphase, blacklist, whitelist){

  sample = caTools::sample.split(areaphase, SplitRatio = 0.75)
  training = subset(areaphase, sample == TRUE, )
  rownames(training)<-NULL
  test  = subset(areaphase, sample == FALSE)
  rownames(test)<-NULL

  # hc_dag =      Hill-Climbing with network topology definition
  # hc_dag_raw =  Hill-Climbing without defining the network topology
  # mmhc_dag =    Max-Min Hill-Climbing with network topology definition
  # mmhc_dag_raw = Max-Min Hill-Climbing without defining the network topology

  # Score-based Learning Algorithm

  hc_dag <- bnlearn::hc(training,
              whitelist = whitelist,
              blacklist = blacklist,
              debug = FALSE)

  hc_dag_raw <- bnlearn::hc(training, debug = FALSE)

  # Hybrid Learning Algorithm

  mmhc_dag <- bnlearn::mmhc(training,
```

```
                    whitelist = whitelist,
                    blacklist = blacklist,
                    debug = FALSE)

  mmhc_dag_raw <- bnlearn::mmhc(training, debug = FALSE)

  #ploting:

  #graphics::plot(hc_dag, main = "hc_dag")
  #graphics::plot(hc_dag_raw, main = "hc_dag_raw")
  #graphics::plot(mmhc_dag, main = "mmhc_dag")
  #graphics::plot(mmhc_dag_raw, main = "mmhc_dag_raw")

  # train networks
  hc_dag_fitted = bnlearn::bn.fit(hc_dag, training)
  hc_dag_raw_fitted = bnlearn::bn.fit(hc_dag_raw, training)

  mmhc_dag_fitted = bnlearn::bn.fit(mmhc_dag, training)
  mmhc_dag_raw_fitted = bnlearn::bn.fit(mmhc_dag_raw, training)



  # validation of networks
  return(validateNetwork(test, training, hc_dag_fitted,
                         hc_dag_raw_fitted, mmhc_dag_fitted,
                         mmhc_dag_raw_fitted))
}

validateNetwork <- function(test, train, dag_fitted1, dag_fitted2,
                            dag_fitted3, dag_fitted4) {
  # Define Target variables (Variables to be predicted)
  pred <-'harvest'
  # Evidence variables
  #(Variables that you will give information to the BN to do the prediction)
  evid <- names(train)[!names(train) %in% pred]

  results1 <- bnMultiVarPrediction(bnFit = dag_fitted1,
                                   trainSet = train,
                                   testSet = test,
                                   to_predict = pred,
                                   to_evidence = evid,
                                   calcFunction = 'predict')

  results2 <- bnMultiVarPrediction(bnFit = dag_fitted2,
                                   trainSet = train,
                                   testSet = test,
                                   to_predict = pred,
                                   to_evidence = evid,
                                   calcFunction = 'predict')

  results3 <- bnMultiVarPrediction(bnFit = dag_fitted3,
                                   trainSet = train,
                                   testSet = test,
```

```
                                    to_predict = pred,
                                    to_evidence = evid,
                                    calcFunction = 'predict')

  results4 <- bnMultiVarPrediction(bnFit = dag_fitted4,
                                    trainSet = train,
                                    testSet = test,
                                    to_predict = pred,
                                    to_evidence = evid,
                                    calcFunction = 'predict')
  # Metrics Evaluation

  metrics1 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results1$dominantList,
                                          predProbList = results1$probList)
  metrics2 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results2$dominantList,
                                          predProbList = results2$probList)

  metrics3 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results3$dominantList,
                                          predProbList = results3$probList)
  metrics4 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results4$dominantList,
                                          predProbList = results4$probList)

  return(c(metrics1, metrics2, metrics3, metrics4))
}
```

C.3 Código fonte – cria redes estáticas - testes – create_bn_test.R

```
testRunNetworks <- function(arealist, areatype){
  area <- list()
  out <- data.frame()
  name <- array()
  iname=1
  for(phase in 1:length(arealist)){
    area[[phase]] <- testCreateNetworks(arealist[[phase]], areatype)

    out <- dplyr::bind_rows(out, area[[phase]][3])
    name[iname] <- paste("phase", phase, "hc_dag")
    iname = iname+1

    out <- dplyr::bind_rows(out, area[[phase]][6])
    name[iname] <- paste("phase", phase, "hc_dag_raw")
    iname = iname+1

    out <- dplyr::bind_rows(out, area[[phase]][9])
    name[iname] <- paste("phase", phase, "mmhc_dag")
    iname = iname+1

    out <- dplyr::bind_rows(out, area[[phase]][12])
    name[iname] <- paste("phase", phase, "mmhc_dag_raw")
    iname = iname+1
  }
  row.names(out$eval)<-name
  return(out$eval)
}

testCreateNetworks <- function (areaphase, areatype){

  sample = caTools::sample.split(areaphase, SplitRatio = 0.75)
  training = subset(areaphase, sample == TRUE, )
  rownames(training)<-NULL
  test  = subset(areaphase, sample == FALSE)
  rownames(test)<-NULL


  blacklist = data.frame(from = c("X_1","X_1",
                                  "X_2", "X_2",
                                  "X_3", "X_3",
                                  "harvest",  "harvest",  "harvest" ),
                         to = c("X_2", "X_3", #from v1
                                "X_1", "X_3", #from v2
                                "X_1", "X_2", #from v3
                                "X_1", "X_2", "X_3")) #from harvest

  # builds whitelist according to area type
  if(areatype == 1){
    whitelist = data.frame(from = c("X_1", "X_2", "X_3"),
                           to = c("harvest", "harvest", "harvest"))
  }else if(areatype == 2){
    whitelist = data.frame(from = c("X_1"),
                           to = c("harvest"))
```

```
}else if(areatype == 3){
  whitelist = data.frame(from = c("X_3"),
                         to = c("harvest"))
}else if(areatype == 4){
  whitelist = data.frame(from = c("X_1", "X_3"),
                         to = c("harvest", "harvest"))
}else if(areatype == 5){
  whitelist = data.frame(from = c("X_2"),
                         to = c("harvest"))
}else if(areatype == 6){
  whitelist = data.frame(from = c("X_1"),
                         to = c("harvest"))
}else if(areatype == 7){
  whitelist = data.frame(from = c("X_1", "X_2", "X_3"),
                         to = c("harvest", "harvest", "harvest"))
}
```

```
# hc_dag:       Hill-Climbing with network topology definition
# hc_dag_raw:   Hill-Climbing without defining the network topology
# mmhc_dag:     Max-Min Hill-Climbing with network topology definition
# mmhc_dag_raw: Max-Min Hill-Climbing without defining the network topology

# Score-based Learning Algorithm

hc_dag <- bnlearn::hc(training,
            whitelist = whitelist,
            blacklist = blacklist,
            debug = FALSE)

hc_dag_raw <-bnlearn::hc(training, debug = FALSE)


# Hybrid Learning Algorithm

mmhc_dag <- bnlearn::mmhc(training,
                 whitelist = whitelist,
                 blacklist = blacklist,
                 debug = FALSE)

mmhc_dag_raw <- bnlearn::mmhc(training, debug = FALSE)

#ploting:

# graphics::plot(hc_dag, main = "hc_dag")
# graphics::plot(hc_dag_raw, main = "hc_dag_raw")
# graphics::plot(mmhc_dag, main = "mmhc_dag")
# graphics::plot(mmhc_dag_raw, main = "mmhc_dag_raw")

# train networks
```

```
  hc_dag_fitted = bnlearn::bn.fit(hc_dag, training)
  hc_dag_raw_fitted = bnlearn::bn.fit(hc_dag_raw, training)

  mmhc_dag_fitted = bnlearn::bn.fit(mmhc_dag, training)
  mmhc_dag_raw_fitted = bnlearn::bn.fit(mmhc_dag_raw, training)


  # validation of networks
  return(testValidateNetwork(test, training, hc_dag_fitted,
                             hc_dag_raw_fitted, mmhc_dag_fitted,
                             mmhc_dag_raw_fitted))
}

testValidateNetwork <- function(test, train, dag_fitted1, dag_fitted2,
                                dag_fitted3, dag_fitted4) {
  # Define Target variables (Variables to be predicted)
  pred <-'harvest'
  # Evidence variables
  #(Variables that you will give information to the BN to do the prediction)
  evid <- names(train)[!names(train) %in% pred]

  results1 <- bnMultiVarPrediction(bnFit = dag_fitted1,
                                   trainSet = train,
                                   testSet = test,
                                   to_predict = pred,
                                   to_evidence = evid,
                                   calcFunction = 'predict')

  results2 <- bnMultiVarPrediction(bnFit = dag_fitted2,
                                   trainSet = train,
                                   testSet = test,
                                   to_predict = pred,
                                   to_evidence = evid,
                                   calcFunction = 'predict')

  results3 <- bnMultiVarPrediction(bnFit = dag_fitted3,
                                   trainSet = train,
                                   testSet = test,
                                   to_predict = pred,
                                   to_evidence = evid,
                                   calcFunction = 'predict')

  results4 <- bnMultiVarPrediction(bnFit = dag_fitted4,
                                   trainSet = train,
                                   testSet = test,
                                   to_predict = pred,
                                   to_evidence = evid,
                                   calcFunction = 'predict')
  # Metrics Evaluation

  metrics1 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results1$dominantList,
                                          predProbList = results1$probList)
```

```
  metrics2 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results2$dominantList,
                                          predProbList = results2$probList)

  metrics3 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results3$dominantList,
                                          predProbList = results3$probList)
  metrics4 <- bnMetricsMultiVarPrediction(reference = test[pred],
                                          prediction = results4$dominantList,
                                          predProbList = results4$probList)

  return(c(metrics1, metrics2, metrics3, metrics4))
}
```

C.4 Código fonte – cria redes dinâmicas – create_Dbn.R

```
createDbn <- function(area){
  #area <- areas_raw[[1]]
  temp <- area[[length(area)]]
  nvar <- dim(temp)[2]
  size <- length(area)
  # Create the expanded df with the n of desired time-cuttings for the D-network
  df_fold <- dbnR::fold_dt(temp, size)
  rowsdf <- dim(df_fold)[1]

  # Filling expanded df with area phase values
  inc <- 1
  for(fase in size:1){
    df_fold[,(((nvar*inc)-nvar)+1):(nvar*inc)] <-
      area[[fase]][1:rowsdf,]
    inc <- inc+1
  }
  # Adding big noise to 'harvest' columns so they do not affect network learning

  for(i in 2:(size)){
    df_fold[,nvar*i] <- df_fold[,nvar*i] +
      matrix(stats::rnorm(dim(df_fold)[1], 0, 200), ncol = 1)
  }

  ## Separating into training and test

  spliter <- array(0, dim(df_fold)[1])
  sampleDBN = caTools::sample.split(spliter, SplitRatio = 0.75)
  trainingDBN = subset(df_fold, sampleDBN == TRUE, )
  rownames(trainingDBN)<-NULL
  testDBN  = subset(df_fold, sampleDBN == FALSE)
  rownames(testDBN)<-NULL

  # Learning the dbn

  dag_natPsoho <- dbnR::learn_dbn_struc(dt = trainingDBN,
                                        size = size,
                                        f_dt = trainingDBN,
                                        method = "natPsoho")

  dag_dmmhc <- dbnR::learn_dbn_struc(dt = NULL,
                                     size = size,
                                     f_dt = trainingDBN,
                                     method = "dmmhc",
                                     intra = F)

  # If existing, remove arcs from nodes 'harvest_t'
  for(i in 1:(size-1)){
    testing <-  paste("harvest_t_", i, sep ="")
    if(length(which(bnlearn::arcs(dag_natPsoho)[,"from"] == testing)) > 0){
      rows <- which(bnlearn::arcs(dag_natPsoho)[,"from"] == testing)
      bnlearn::arcs(dag_natPsoho) <- bnlearn::arcs(dag_natPsoho)[-rows,]
    }
  }
```

```
for(i in 1:(size-1)){
  testing <-  paste("harvest_t_", i, sep ="")
  if(length(which(bnlearn::arcs(dag_dmmhc)[,"from"] == testing)) > 0){
    rows <- which(bnlearn::arcs(dag_dmmhc)[,"from"] == testing)
    bnlearn::arcs(dag_dmmhc) <- bnlearn::arcs(dag_dmmhc)[-rows,]
  }
}

# plot(dag_natPsoho)
# plot(dag_dmmhc)


# Training the network
dag_natPsoho_fited <- dbnR::fit_dbn_params(dag_natPsoho, trainingDBN)

dag_dmmhc_fited <- dbnR::fit_dbn_params(dag_dmmhc, trainingDBN)

# dag_natPsoho
predict_dag_natPsoho_approx <- dbnR::forecast_ts(testDBN, dag_natPsoho_fited,
                                                 obj_vars = "harvest_t_0",
                                                 rep = 1,mode = "approx",
                                                 ini = 1, len = length(testDBN),
                                                 print_res = F, plot_res = F)

predict_dag_natPsoho_exact <- dbnR::forecast_ts(testDBN, dag_natPsoho_fited,
                                                obj_vars = "harvest_t_0",
                                                rep = 1, mode = "exact",
                                                ini = 1, len = length(testDBN),
                                                print_res = F, plot_res = F)
#dag_dmmhc
predict_dag_dmmhc_approx <- dbnR::forecast_ts(testDBN, dag_dmmhc_fited,
                                              obj_vars = "harvest_t_0",
                                              rep = 1, mode = "approx",
                                              ini = 1, len = length(testDBN),
                                              print_res = F, plot_res = F)

predict_dag_dmmhc_exact <- dbnR::forecast_ts(testDBN, dag_dmmhc_fited,
                                             obj_vars = "harvest_t_0",
                                             rep = 1, mode = "exact",
                                             ini = 1, len = length(testDBN),
                                             print_res = F, plot_res = F)


metrics_df_natPsoho_approx =
  data.frame(obs = predict_dag_natPsoho_approx$orig$harvest_t_0,
             pred = predict_dag_natPsoho_approx$pred$harvest_t_0)

metrics_df_natPsoho_exact =
  data.frame(obs = predict_dag_natPsoho_exact$orig$harvest_t_0,
             pred = predict_dag_natPsoho_exact$pred$harvest_t_0)

metrics_df_dmmhc_approx =
  data.frame(obs = predict_dag_dmmhc_approx$orig$harvest_t_0,
```

```
              pred = predict_dag_dmmhc_approx$pred$harvest_t_0)

  metrics_df_dmmhc_exact =
    data.frame(obs = predict_dag_dmmhc_exact$orig$harvest_t_0,
              pred = predict_dag_dmmhc_exact$pred$harvest_t_0)

  natPsoho_approx =  c(caret::defaultSummary(metrics_df_natPsoho_approx),
                       accurCalc(metrics_df_natPsoho_approx))

  natPsoho_exact =  c(caret::defaultSummary(metrics_df_natPsoho_exact),
                      accurCalc(metrics_df_natPsoho_exact))

  dmmhc_approx = c(caret::defaultSummary(metrics_df_dmmhc_approx),
                   accurCalc(metrics_df_dmmhc_approx))

  dmmhc_exact = c(caret::defaultSummary(metrics_df_dmmhc_exact),
                  accurCalc(metrics_df_dmmhc_exact))

  metrics_dags = data.frame(cbind(natPsoho_approx,
                                  dmmhc_approx,
                                  natPsoho_exact,
                                  dmmhc_exact))

  return(metrics_dags)
}
```

C.5 Código fonte – cria redes dinâmicas - teste – create_Dbn_test.R

```
createDbn_test <- function(area, areaType){
  temp <- area[[length(area)]]
  nvar <- dim(temp)[2]
  size <- length(area)
  # Create the expanded df with the n of desired time-cuttings for the D-network
  df_fold <- dbnR::fold_dt(temp, size)
  rowsdf <- dim(df_fold)[1]

  # Filling expanded df with area phase values
  inc <- 1
  for(fase in size:1){
    df_fold[,(((nvar*inc)-nvar)+1):(nvar*inc)] <-
      area[[fase]][1:rowsdf,]
    inc <- inc+1
  }
  # Adding big noise to 'harvest' columns so they do not affect network learning

  for(i in 2:(size)){
    df_fold[,nvar*i] <- df_fold[,nvar*i] +
      matrix(stats::rnorm(dim(df_fold)[1], 0, 200), ncol = 1)
  }

  ## Separating into training and test

  spliter <- array(0, dim(df_fold)[1])
  sampleDBN = caTools::sample.split(spliter, SplitRatio = 0.50)
  trainingDBN = subset(df_fold, sampleDBN == TRUE, )
  rownames(trainingDBN)<-NULL
  testDBN  = subset(df_fold, sampleDBN == FALSE)
  rownames(testDBN)<-NULL

  # Learning the dbn

  dag_natPsoho <- dbnR::learn_dbn_struc(dt = trainingDBN,
                                        size = size,
                                        f_dt = trainingDBN,
                                        method = "natPsoho")

  dag_dmmhc <- dbnR::learn_dbn_struc(dt = NULL,
                                     size = size,
                                     f_dt = trainingDBN,
                                     method = "dmmhc",
                                     intra = F)

  dag_manual <- dag_dmmhc
  topology <- setTopology(areaType)
  bnlearn::arcs(dag_manual) <- topology




  # If existing, remove arcs from nodes 'harvest_t'
```

```
for(i in 1:(size-1)){
  testing <-  paste("harvest_t_", i, sep ="")
  if(length(which(bnlearn::arcs(dag_natPsoho)[,"from"] == testing)) > 0){
    rows <- which(bnlearn::arcs(dag_natPsoho)[,"from"] == testing)
    bnlearn::arcs(dag_natPsoho) <- bnlearn::arcs(dag_natPsoho)[-rows,]
  }
}

for(i in 1:(size-1)){
  testing <-  paste("harvest_t_", i, sep ="")
  if(length(which(bnlearn::arcs(dag_dmmhc)[,"from"] == testing)) > 0){
    rows <- which(bnlearn::arcs(dag_dmmhc)[,"from"] == testing)
    bnlearn::arcs(dag_dmmhc) <- bnlearn::arcs(dag_dmmhc)[-rows,]
  }
}

# Training the network
dag_natPsoho_fited <- dbnR::fit_dbn_params(dag_natPsoho, trainingDBN)
dag_dmmhc_fited <- dbnR::fit_dbn_params(dag_dmmhc, trainingDBN)
dag_manual_fited <- dbnR::fit_dbn_params(dag_manual, trainingDBN)

#FORECAST
# dag_natPsoho
predict_dag_natPsoho_approx <- dbnR::forecast_ts(testDBN, dag_natPsoho_fited,
                                                 obj_vars = "harvest_t_0",
                                                 rep = 1,mode = "approx",
                                                 ini = 1, len = dim(testDBN)[1],
                                                 print_res = F, plot_res = F)



#dag_dmmhc
predict_dag_dmmhc_approx <- dbnR::forecast_ts(testDBN, dag_dmmhc_fited,
                                              obj_vars = "harvest_t_0",
                                              rep = 1, mode = "approx",
                                              ini = 1, len = dim(testDBN)[1],
                                              print_res = F, plot_res = F)

#dag_manual
predict_dag_manual_approx <- dbnR::forecast_ts(testDBN, dag_manual_fited,
                                               obj_vars = "harvest_t_0",
                                               rep = 1, mode = "approx",
                                               ini = 1, len = dim(testDBN)[1],
                                               print_res = F, plot_res = F)

metrics_df_natPsoho_approx =
  data.frame(obs = predict_dag_natPsoho_approx$orig$harvest_t_0,
             pred = predict_dag_natPsoho_approx$pred$harvest_t_0)

metrics_df_dmmhc_approx =
  data.frame(obs = predict_dag_dmmhc_approx$orig$harvest_t_0,
             pred = predict_dag_dmmhc_approx$pred$harvest_t_0)
```

```
  metrics_df_predict_dag_manual_approx =
    data.frame(obs = predict_dag_manual_approx$orig$harvest_t_0,
               pred = predict_dag_manual_approx$pred$harvest_t_0)

  natPsoho_approx =  c(caret::defaultSummary(metrics_df_natPsoho_approx),
                       accurCalc(metrics_df_natPsoho_approx))

  dmmhc_approx = c(caret::defaultSummary(metrics_df_dmmhc_approx),
                   accurCalc(metrics_df_dmmhc_approx))

  dag_manual_approx = c(caret::defaultSummary(metrics_df_predict_dag_manual_approx),
                        accurCalc(metrics_df_predict_dag_manual_approx))

  metrics_dags = data.frame(cbind(natPsoho_approx,
                                  dmmhc_approx,
                                  dag_manual_approx))

  return(metrics_dags)
}
```

C.6 Código fonte – complementares - auxiliary.R

```
f_minmax <- function(x){
  minmax = (max(x)-min(x))
  if(minmax==0){
    return(min(x))
  }
  return((x - min(x))/minmax)
}

geraGraph <-function(areaN, file, subtitle){

  text_size <- 20

  areaN$networks <- rep(c("hc_dag", "hc_dag_raw", "mmhc_dag", "mmhc_dag_raw"),
                        each = 1)
  areaN$season <- rep(c("phase 1", "phase 2", "phase 3", "phase 4", "phase 5"),
                          each = 4)
  areaN$season <- factor(areaN$season, levels =
                             c("phase 1", "phase 2", "phase 3", "phase 4", "phase 5"))

  grDevices::png(file, width = 1024, height = 768)

  graphics::plot(
    ggplot2::ggplot(areaN, ggplot2::aes(season, accuracy, fill = networks)) +
      ggplot2::scale_y_continuous(limits = c(0,100), breaks = seq(0,100,25)) +
      ggplot2::labs(
        subtitle = subtitle
        )+
      ggplot2::scale_fill_brewer(palette = "Spectral") + # Spectral; Paired; Set3
      ggplot2::geom_col(position = "dodge",
                        show.legend = TRUE, linetype = 1, alpha = .65) +
      ggplot2::geom_text(ggplot2::aes(label = accuracy), vjust = 1,
                         position = ggplot2::position_dodge(.9),
                         size = 5, color = "black")+
      ggplot2::theme(axis.ticks.length = ggplot2::unit(.3, "cm"),
                     axis.text =
                       ggplot2::element_text(size =
                                               text_size, colour = "black"),
                     axis.title =
                       ggplot2::element_text(size =
                                               text_size, colour = "black"),
                     axis.ticks =
                       ggplot2::element_line(colour = "black"),
                     panel.border =
                       ggplot2::element_rect(colour =
                                               "black", fill = NA, size = 0.5),
                                             legend.position="bottom",
                     legend.text = ggplot2::element_text(colour="black",
                                                         size=text_size, face="bold"),

                     legend.title =  ggplot2::element_text(colour="black",
                                                           size=text_size, face="bold"),

                     plot.title.position = "panel",
```

```
                    plot.caption.position = "panel",
                    title = ggplot2::element_text(colour="black",
                                                  size=text_size, face="bold")
                    # panel.grid =  element_blank()
      )+
      ggplot2::geom_hline(yintercept = mean(areaN$accuracy),
                          linetype = "solid", size = .2))

  grDevices::dev.off()
}

defClasses <- function(n_classes, df_nome) {
  val_max <- max(df_nome, na.rm = TRUE)
  val_min <- min(df_nome, na.rm = TRUE)
  amp_total <- val_max - val_min
  amp_classe <- amp_total / n_classes
  classes <- vector()
  classes <- val_min + amp_classe
  for (i in 1:(n_classes -1)) {
    classes[i+1] <- classes[i] + amp_classe
  }
  return(classes)
}

classficator <- function(input, varclass, class_names) {
  input <- replace(input, input <= varclass[1], class_names[1])
  for (i in 2 : length(varclass)){
    input <- replace(input, input > varclass[i-1] &
                       input <= varclass[i], class_names[i])
  }
  return(input)
}

accurCalc <- function(df){
  #classificando
  #class_names <- c("L", "ML", "M", "MH", "H")
  class_names <- c("L", "M", "H")
  class_x1 <- defClasses(3, df$obs)
  # class_x1 <- defClasses(3, df)
  class_x1[length(class_x1)] <- max(df)
  x1_class <- classficator(df, class_x1, class_names)
  x1_class$obs <- factor(x1_class$obs)
  x1_class$pred <- factor(x1_class$pred, levels = levels(x1_class$obs))

  #teste <- caret::confusionMatrix(x1_class$pred, x1_class$obs)
  return((caret::confusionMatrix(x1_class$pred, x1_class$obs))$overall[1])
}

setTopology <-function(areaType){
  if(areaType == 1){
    from <- c("X_1_t_4", "X_2_t_4", "X_3_t_4", "harvest_t_4",
              "X_1_t_3", "X_2_t_3", "X_3_t_3", "harvest_t_3",
              "harvest_t_2",
```

```
            "harvest_t_1")
  to <- c("harvest_t_4", "harvest_t_4", "harvest_t_4", "harvest_t_3",
          "harvest_t_3", "harvest_t_3", "harvest_t_3", "harvest_t_2",
          "harvest_t_1",
          "harvest_t_0")
  topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
  colnames(topology_temp) = c("from", "to")
  topology_temp[,1] <- from
  topology_temp[,2] <- to
}else
  if(areaType == 2){
    from <- c("X_1_t_4", "harvest_t_4",
              "X_1_t_3", "harvest_t_3",
              "X_1_t_2", "harvest_t_2",
              "X_1_t_1", "harvest_t_1",
              "X_1_t_0")
    to <- c("harvest_t_4", "harvest_t_3",
            "harvest_t_3", "harvest_t_2",
            "harvest_t_2", "harvest_t_1",
            "harvest_t_1", "harvest_t_0",
            "harvest_t_0")
    topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
    colnames(topology_temp) = c("from", "to")
    topology_temp[,1] <- from
    topology_temp[,2] <- to

  }else
    if(areaType == 3){
      from <- c("X_3_t_4", "harvest_t_4",
                "X_3_t_3", "harvest_t_3",
                "X_3_t_2", "harvest_t_2",
                "X_3_t_1", "harvest_t_1",
                "X_3_t_0")
      to <- c("harvest_t_4", "harvest_t_3",
              "harvest_t_3", "harvest_t_2",
              "harvest_t_2", "harvest_t_1",
              "harvest_t_1", "harvest_t_0",
              "harvest_t_0")
      topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
      colnames(topology_temp) = c("from", "to")
      topology_temp[,1] <- from
      topology_temp[,2] <- to
    }else
      if(areaType == 4){
        from <- c("X_1_t_4", "X_3_t_4", "harvest_t_4",
                  "X_1_t_3", "X_3_t_3", "harvest_t_3",
                  "X_1_t_2", "X_3_t_2", "harvest_t_2",
                  "X_1_t_1", "X_3_t_1", "harvest_t_1",
                  "X_1_t_0", "X_3_t_0")
        to <- c("harvest_t_4", "harvest_t_4", "harvest_t_3",
                "harvest_t_3", "harvest_t_3", "harvest_t_2",
                "harvest_t_2", "harvest_t_2", "harvest_t_1",
                "harvest_t_1", "harvest_t_1", "harvest_t_0",
```

```
            "harvest_t_0", "harvest_t_0")
    topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
    colnames(topology_temp) = c("from", "to")
    topology_temp[,1] <- from
    topology_temp[,2] <- to
  }else
    if(areaType == 5){
      from <- c("X_2_t_4", "harvest_t_4",
                "X_2_t_3", "harvest_t_3",
                "X_2_t_2", "harvest_t_2",
                "X_2_t_1", "harvest_t_1",
                "X_2_t_0")
      to <- c("harvest_t_4", "harvest_t_3",
              "harvest_t_3", "harvest_t_2",
              "harvest_t_2", "harvest_t_1",
              "harvest_t_1", "harvest_t_0",
              "harvest_t_0")
      topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
      colnames(topology_temp) = c("from", "to")
      topology_temp[,1] <- from
      topology_temp[,2] <- to
    }else
      if(areaType == 6){
        from <- c("X_1_t_4", "harvest_t_4",
                  "X_1_t_3", "harvest_t_3",
                  "X_1_t_2", "harvest_t_2",
                  "X_1_t_1", "harvest_t_1",
                  "X_1_t_0")
        to <- c("harvest_t_4", "harvest_t_3",
                "harvest_t_3", "harvest_t_2",
                "harvest_t_2", "harvest_t_1",
                "harvest_t_1", "harvest_t_0",
                "harvest_t_0")
        topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
        colnames(topology_temp) = c("from", "to")
        topology_temp[,1] <- from
        topology_temp[,2] <- to
      }else
        if(areaType == 7){
          from <- c("X_1_t_4", "X_2_t_4", "X_3_t_4", "harvest_t_4",
                    "X_1_t_3", "X_2_t_3", "X_3_t_3", "harvest_t_3",
                    "X_1_t_2", "X_2_t_2", "X_3_t_2", "harvest_t_2",
                    "X_1_t_1", "X_2_t_1", "X_3_t_1", "harvest_t_1",
                    "X_1_t_0", "X_2_t_0", "X_3_t_0")
          to <- c("harvest_t_4", "harvest_t_4", "harvest_t_4", "harvest_t_3",
                  "harvest_t_3", "harvest_t_3", "harvest_t_3", "harvest_t_2",
                  "harvest_t_2", "harvest_t_2", "harvest_t_2", "harvest_t_1",
                  "harvest_t_1", "harvest_t_1", "harvest_t_1", "harvest_t_0",
                  "harvest_t_0", "harvest_t_0", "harvest_t_0")
          topology_temp <- matrix(0, nrow = length(from), ncol = 2 )
          colnames(topology_temp) = c("from", "to")
          topology_temp[,1] <- from
          topology_temp[,2] <- to
```

```
            }

  return(topology_temp)
}
```

C.7 Código fonte – complementares - tests.R

```
test_datagen <- function(){

  nHarvests = 1000
  nphases = 5
  nAreas = 6
  nVars = 3
  nClass = 5

  areas <- testRunDataGen(nHarvests, nphases, nAreas, nVars, nClass)
  return(areas)
}

test_createBN <- function(){
  nHarvests = 1000
  nphases = 5
  nAreas = 6
  nVars = 3
  nClass = 5

  areas <- testRunDataGen(nHarvests, nphases, nAreas, nVars, nClass)
  areas_raw <- areas[1:(length(areas)/2)]
  areas_dis <- areas[((length(areas)/2)+1):length(areas)]
  area1 <- testRunNetworks(areas_dis[[1]], 1)
  return(area1)
}

test_createDBN <- function(){
  nHarvests = 1000
  nphases = 5
  nAreas = 6
  nVars = 3
  nClass = 5

  areas <- testRunDataGen(nHarvests, nphases, nAreas, nVars, nClass)
  areas_raw <- areas[1:(length(areas)/2)]
  areas_dis <- areas[((length(areas)/2)+1):length(areas)]
  area1 <- createDbn(areas_raw[[1]])
  return(area1)
}
```

APÊNDICE D – DOCUMENTAÇÃO – AGROBAYES

Package 'AgroBayes'

February 7, 2023

Type Package

Title Generates static and dynamic Bayesian networks for forecasting crop results

Version 0.1.0

Author Guilherme Afonso Halal [aut, cre], Ana Paula Ludtke Ferreira [ctb]

Maintainer Guilherme Afonso Halal <guilhermehalal@gmail.com>

Description This package permits, from variables related to agricultural production, to generate static and dynamic Bayesian networks that allow forecasting crop results. The package allows comparing the effectiveness of models generated from some network structure learning methods. The networks are generated based on functions available in the bnlearn (static networks) and dbnR (dynamic networks) packages. The forecasting of the crop production can be done from sets of data geographically separated (specific areas of the plantation) and also from chronological cuts (phenological phases).
All functions identified with the term TEST FUNCTION in the documentation and with a name starting with 'test', were added to the package to run the demonstration of the package's functionalities. The /AgroBayes/vignettes/demo.Rmd provides an interactive example of how the package works.
The final user must organize the dataset in dataframes representing the interval referring to a phenological phase and group them in lists according to the area of the plantation site.
Disclaimer: The user should be careful regarding the use of networks inferred by data. In the tests carried out, most of the generated networks had a topology that does not match the expected one. The use of network generation functions should be done with prior knowledge of the existing relationships between the available variables and the production result so that the results obtained by inference can be compared with the expected networks. In future versions of the package there will be modifications in order to try to make the inference process more accurate, allowing the use of networks inferred by data in a more generalized way.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr,
rmarkdown,
testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Imports bnlearn, dbnR, Hmisc, dplyr, caTools, ggplot2, data.table, caret, stats, graphics, grDevices

R topics documented:

`createDbn` *Creates dynamic Bayesian networks for performance evaluation*

Description

Using harvest data from all phenological phases of the cultivar, from a specific area of the plantation, dynamic Bayesian networks are generated (using the `dbnR::learn_dbn_struc` functions with `natPsoho` and `dmmhc` methods), trained (using the `dbnR::fit_dbn_params` function) and evaluated for performance (using `dbnR::forecast_ts` and `caret::defaultSummary`). Two networks are created, one using `natPsoho` learning method and other using `dmmhc` method. The DBN are tested using `dbnR::forecast_ts` with `exact` and `approx` methods then, from the return of these functions, the RMSE, R-squared and MAE metrics are calculated and returned

Usage

```
createDbn(area)
```

Arguments

`area`	A dataframe of continuous data, from a specific area of the plantation to be used

Value

A dataframe with the dynamics networks metrics (RMSE, R-squared, MAE)

Examples

```
metricsDbn = createDbn(area_1)
```

createNetworks *Creates Bayesian networks for performance evaluation*

Description

Using harvest data from a phenological phase of the cultivar, from a specific area of the plantation, Bayesian network are generated (using the bnlearn::hc and bnlearn::mmhc functions), trained (using the bnlearn::bn.fit function) and evaluated for performance (using validateNetwork). Four networks are created, two from the pre-established topology and two learned only from the presented data.

Usage

```
createNetworks(areaphase, blacklist, whitelist)
```

Arguments

areaphase	the dataframe of discretized data to be used
blacklist	a dataframe of character string with two columns, it is passed as a parameter to bnlearn learn functions in order to avoid these arcs composing the final network
whitelist	a dataframe of character string with two columns, it is passed as a parameter to bnlearn learn functions in order to guarantee these arcs composing the final network

Value

Network evaluation metrics, as calculated in the validateNetwork function

Examples

```
metricsArea1Phase1 = list()
blacklist = data.frame(
from = c("X_1","X_1",
         "X_2", "X_2",
         "X_3", "X_3",
         "harvest",  "harvest",  "harvest" ),
to = c("X_2", "X_3", #from v1
       "X_1", "X_3", #from v2
       "X_1", "X_2", #from v3
       "X_1", "X_2", "X_3")) #from col
whitelist = data.frame(
from = c("X_1", "X_2", "X_3"),
to = c("harvest", "harvest", "harvest"))
areaphase = data.frame(area1_phase_1)
metricsArea1Phase1 = createNetworks (areaphase, blacklist, whitelist)
```

runNetworks *Executes* createNetworks *function to an area*

Description

Executes the createNetworks function for all phenological phases of an area. It also organizes the generated metrics in a dataframe.

Usage

```
runNetworks(arealist, blacklist, whitelist)
```

Arguments

arealist	list os dataframes of all phenological phase on a specific area of the plantation
blacklist	a dataframe of character string with two columns, it is passed as a parameter to bnlearn learn functions in order to avoid these arcs composing the final network
whitelist	a dataframe of character string with two columns, it is passed as a parameter to bnlearn learn functions in order to guarantee these arcs composing the final network

Value

A dataframe organizing the network metrics generated in the createNetworks function. The lines represent the network performance learned in each phenological phase of the area (arealist)

Examples

```
blacklist = data.frame(
from = c("X_1","X_1",
         "X_2", "X_2",
         "X_3", "X_3",
         "harvest",  "harvest",  "harvest" ),
to = c("X_2", "X_3", #from v1
       "X_1", "X_3", #from v2
       "X_1", "X_2", #from v3
       "X_1", "X_2", "X_3")) #from harvest
whitelist = data.frame(
from = c("X_1", "X_2", "X_3"),
to = c("harvest", "harvest", "harvest"))
arealist <- list(area1)
metricsArea1 <- createNetworks (arealist, blacklist, whitelist)
```

`testBuildSimulationData` *Initialize the values of all variables*

Description

Initialized by the `testRunDataGen` function, it generates all the values of the independent variables and the dependent variable, returning a list of areas with harvest data.

Usage

```
testBuildSimulationData(nHarvests, nPhases, nAreas = NULL, ...)
```

Arguments

`nHarvests`	number of harvests.
`nPhases`	number of phenological phases.
`nAreas`	number of areas of the plantation site if not informed the default is 6 areas.
`...`	parameters passed to `testDefVars` function

Value

list of areas with harvest data.

Examples

```
nHarvests = 1000
nphases = 5
nAreas = 10
nVars = 6
areas_list  = testBuildSimulationData(nHarvests, nphases)
areas_list_2  = testBuildSimulationData(nHarvests, nPhases, nAreas, nVars)
```

`testCreateDataFrames` *Organizes data in dataframes so it can be used in Bayesian learning functions*

Description

Called by the `testRunDataGen` function. Function receives the list of variables referring to an area and organizes it into dataframes, one dataframe for each phenological phase.

Usage

```
testCreateDataFrames(data)
```

Arguments

data	list of variables referring to an area

Value

list of dataframes, one for each phenological stage.

Examples

```
area_1_df = testCreateDataFrames(area_1_list)
```

---

testDefVars *Defines the type and quantity of independent variables*

Description

Called by the testBuildSimulationData function. Function allows defining the type and quantity of initialized independent variables for data generation. It is possible to define a number of variables greater than 3. If the type of variable is defined, it is mandatory to inform a list with the name of the variables as well. If only the number of variables are defined, the default nomenclature is X_1, X_2, ..., X_n-1, X_n is applied, with randomly defined behavior. If the number of variables is not defined, 3 variables will be initialized, one that always grows over time, a constant and one that oscillates over time.

Usage

```
testDefVars(n_var = NULL, type_var = NULL, name_var = NULL)
```

Arguments

n_var	number of variables (not counting the dependent variable) if not informed the default is 3 variables.
type_var	a list of types of variables. If informed the name_var parameter must be informed too.
name_var	a list of names of variables. If informed the type_var parameter must be informed too.

Value

list of variables, that is a list containing (name of the variable, type of variable, value min and value max)

Examples

```
defProdVariables = testDefVars(...)
defProdVariables = testDefVars(
4,
c("precipitation", "Mn_rate", "Zn_rate", "avg_temp"),
 c(1, 3, 3, 2))
```

`testRunDataGen` *Starts data generation*

Description

It receives the parameters for the `testBuildSimulationData` function to generate simulated data for carrying out the package tests. It also distributes the data generated in lists of dataframes representing the set of phenological phases of each productive area. The data are still treated so that there is a copy of the data set discretized and another not, for the purpose of comparing performance between dynamic and static networks.

Usage

```
testRunDataGen(nHarvests, nphases, nAreas, nVars, nClass, ...)
```

Arguments

`nHarvests`	number of harvests.
`nphases`	number of phenological phases.
`nAreas`	number of areas of the plantation site.
`nVars`	number of variables (not counting the dependent variable).
`nClass`	Number of classes for dataframe discretization. Must be 5 or 3.
`...`	parameters passed to `testBuildSimulationData` function

Value

list with two dataframes one with continuous data and the other with discrete data

Examples

```
areas <- testRunDataGen(nHarvests, nphases, nAreas, nVars, nClass)
```

testSetSimVarValues *Generates the values of dependent and independent variables*

Description

Called by the testBuildSimulationData function. Function starts the values of dependent and independent variables The 'harvest' variables is is calculated from rules that depend on the number of independent variables and the type of area where this harvest was generated. There are 6 types of areas for standard situations, 3 independent variables, and 6 types of areas for situations where there are more than 3 variables.

Usage

```
testSetSimVarValues(nHarvests, areatype, prodvars, nPhases)
```

Arguments

nHarvests	number of harvests.
areatype	integer that defines the type of area. It determines the kind of relationship between the independent variables and the 'harvest' variable
prodvars	list of independent variables, that is a list containing (name of the variable, type of variable, value min and value max)
nPhases	number of phenological phases.

Value

list of values of all variables

Examples

```
variables =
testSetSimVarValues(nHarvests, areatype, defProdVariables, nPhases)
```

validateNetwork *Generates Bayesian networks performance evaluation*

Description

Using the functions available in the repository https://github.com/KaikeWesleyReis/bnlearn-multivar-prediction- calculates the metrics of the four Bayesian networks generated in the createNetworks function when executed in context of runNetworks function.

Usage

```
validateNetwork(
  test,
  train,
  dag_fitted1,
  dag_fitted2,
  dag_fitted3,
  dag_fitted4
)
```

Arguments

test	dataframe to be used to test the Bayesian networks. It is composed of a 25 the createNetworks function.
train	dataframe to be used to train the Bayesian networks. It is composed of a 75 the createNetworks function.
dag_fitted1	Fitted Bayesian network to be tested
dag_fitted2	Fitted Bayesian network to be tested
dag_fitted3	Fitted Bayesian network to be tested
dag_fitted4	Fitted Bayesian network to be tested

Value

List of values returned from bnMetricsMultiVarPrediction.

Index

ANEXO A – CÓDIGO FONTE – NÃO CRIADO PELO AUTOR

```
# trecho copiado do repositório
# https://github.com/KaikeWesleyReis/bnlearn-multivar-prediction-metrics

# FUNCTION - Multi Variable Discrete prediction
bnMultiVarPrediction <- function(bnFit, trainSet, testSet, to_predict,
                                 to_evidence, nSamples = NULL,
                                 calcFunction = NULL){
  # Probabilities predictions for each sample
  pred_list_prob = list()

  # Dominant output predicted for each sample
  pred_list_domi = list()

  # Auxiliar variables
  N <- nrow(testSet) # Number of samples
  np <- length(to_predict) # Number of variables to predict

  # Loop into all possible variables to Generate our output format
  for(j in to_predict){
    # TARGET LEVELS (CATEGORIES)
    tv_lvls =  levels(trainSet[,j])
    # DATAFRAME - PROBABILITIES PREDICTIONS
    prob_pred <- stats::setNames(data.frame(matrix(ncol = length(tv_lvls),
                                                   nrow = N)), tv_lvls)
    # DATAFRAME - DOMINANT OUTCOME
    domi_pred <- stats::setNames(data.frame(matrix(ncol = 1, nrow = N)), j)
    # LISTS - APPEND
    pred_list_prob[[j]] = prob_pred
    pred_list_domi[[j]] = domi_pred
  }

  # Multi Var Prediction :: PREDICT FUNCTION
  if(calcFunction == 'predict' || is.null(calcFunction)){
    for (i in 1:N){
      # Prediction process
      for(j in to_predict){
        predicted = stats::predict(bnFit, j,
                                   testSet[i, names(testSet) %in% to_evidence],
                                   prob = TRUE, method = 'bayes-lw')
        ## TAKING IMPORTANT RESULTS
        dominant = as.character(predicted)
        probs = attr(predicted,'prob')
        ## DOMINANT OUTPUT
        pred_list_domi[[j]][i,j] = dominant
        ## PROBABILITY OUTPUT
        tv_lvls = colnames(pred_list_prob[[j]])
        ## LOOP INTO ALL LEVELS TO SAVE YOUR PROBABILITY
        for(k in tv_lvls){
          pred_list_prob[[j]][i,k] = as.numeric(probs[k,])
        }
      }
    }
  }
```

```
  # Multi Var Prediction :: CPDIST FUNCTION
  else if(calcFunction == 'cpdist'){
    # nSamples Verification
    if(is.null(nSamples) || typeof(nSamples) != 'double'){ # Default value
      # for N samples is 10.000 samples generated
      nSamples <- 10000
    }
    # Prediction process
    for (i in 1:N){
      predicted =
        bnlearn::cpdist(fitted = bnFit, nodes = to_predict,
                        evidence =
                          as.list(testSet[i, names(testSet) %in% to_evidence]),
                        n = nSamples, method = 'lw')
      for(j in to_predict){ # Loop into all variables to be predicted
        # PROBABILITY OUTPUT
        probs = table(predicted[,j])/nrow(predicted)
        # DOMINANT OUTPUT
        pred_list_domi[[j]][i,j] = names(which.max(probs))
        # KEEP PROB VALUES
        tv_lvls = colnames(pred_list_prob[[j]])
        for(k in tv_lvls){ # Loop into all lvls to save the probability
          pred_list_prob[[j]][i,k] = as.numeric(probs[k])
        }
      }
    }
  }

  # Turn dominant output in factors - help metrics CM
  for(j in to_predict){
    #levels(pred_list_domi[[j]]) <- levels(trainSet[,j])
    pred_list_domi[[j]] <- factor(x = unlist(pred_list_domi[[j]],
                                             use.names = F),
                                  levels = levels(trainSet[,j]))
  }
  # Returning
  ret_list <- list("probList" = pred_list_prob,
                   "dominantList" = pred_list_domi)
  return(ret_list)
}

# FUNCTION - Metrics :: Confusion Matrix by OVA and Scoring Rules
bnMetricsMultiVarPrediction <- function(reference, prediction, predProbList){
  # AUXILIAR VARs
  np <- length(names(prediction))
  to_predict <- names(prediction)
  N <- nrow(reference)

  # METRICS DATASET CREATION
  metricsType <- c('accuracy','accuracyOVA','sensibilityOVA',
                   'specificityOVA','precisionOVA','f1-scoreOVA','mccOVA',
                   'sphericalPayoff','brierLoss','logLoss')
  metrics <- stats::setNames(data.frame(matrix(ncol = length(metricsType),
```

```
                                          nrow = length(to_predict)),
                                    row.names = to_predict), metricsType)


# CONFUSION MATRIX
cm_list <- stats::setNames(vector(mode = "list", length = np), to_predict)
for(j in to_predict){
  cm_list[[j]] <- table(prediction[[j]], reference[[j]])

  #### PRINT - CM
  #cat(paste('CONFUSION MATRIX - ',j,' Variable:\n'))
  #print(cm_list[[j]])
  #cat('\n')

}

# ONE VS ALL (OVA) CALCULATION
ova_list <- stats::setNames(vector(mode = "list", length = np), to_predict)
for(j in to_predict){
  # OVA INIT
  lvls_var = colnames(cm_list[[j]])
  ova_list[[j]] <- stats::setNames(vector(mode = "list",
                                          length = length(lvls_var)),
                                   lvls_var)

  ### PRINT
  #cat(paste('ONE VS ALL - ',j,'Variable:\n\n'))

  # LOOP TO DEVELOP EACH v CLASS VS ALL MATRIX FOR A j VARIABLE
  for(v in lvls_var){
    ## MATRIX CREATION
    ova_lvl <- matrix(0,nrow = 2, ncol = 2)
    row.names(ova_lvl) = c(v,'All')
    colnames(ova_lvl) = c(v,'All')
    ## RENAME cm_list FROM OUR j VARIABLE TO BETTER CODE READING
    cm = cm_list[[j]]
    ## AUXILIAR VARIABLES
    rs = rowSums(cm)
    cs = colSums(cm)
    n = sum(cm)

    ## TAKING BINARIES VALUES FROM A BINARY C.M.
    tp = cm[v,v]
    fn = rs[v] - cm[v,v]
    fp = cs[v] - cm[v,v]
    tn = n - rs[v] - cs[v] + cm[v,v]

    ## INSERT THE VALUES IN OVA C.M.
    ova_lvl[v,v] <- tp
    ova_lvl[v,'All'] <- fn
    ova_lvl['All',v] <- fp
    ova_lvl['All','All'] <- tn
```

```
    ## SAVE OVA INSIDE THE LIST FOR THAT j TARGET
    ova_list[[j]][[v]] <- ova_lvl

    #### PRINT - OVA
    #print(ova_list[[j]][[v]])
    #cat('\n')

  }
}

# CONFUSION MATRIX METRICS - ONE VS ALL :: SEN,
##SPECIFICITY, RECALL, ACCURACY, F1-SCORE, MCC, ACCURACY OVA
for(j in to_predict){
  cm <- cm_list[[j]]
  # ACCURACY
  cmACC <- 100*(sum(diag(cm))/sum(cm))

  ### SAVE INTO DATAFRAME RESULT
  metrics[j,'accuracy'] <- round(cmACC,2)

  ### PRINT - ACCURACY MULTI-CLASS
  cat(paste('#### CONFUSION MATRIX METRICS - ',j,' Variable ####\n'))
  cat(paste('+ ACCURACY: ', round(cmACC,2),'\n'))

  ## OVA :: AUXILIAR VARIABLES - REPRESENT VECTOR FOR EACH v LEVELS VS ALL
  acc = c()
  sen = c()
  spe = c()
  pre = c()
  f1s = c()
  mcc = c()
  ## OVA :: AUXILIAR VARs
  lvls <- names(ova_list[[j]])
  len_lvls <- length(lvls)
  ## OVA :: LOOP INTO ALL l LEVELS FROM j
  ##VARIABLE TO CALCULATE METRICS FOR EACH LEVEL
  for(l in lvls){
    ## AUXILIAR VARIABLE TO IMPROVE CODE READING
    ovaCM <- ova_list[[j]][[l]]
    ## GETTING BINARY TERMS FROM ovaCM
    tp = ovaCM[l,l]
    fn = ovaCM[l,'All']
    fp = ovaCM['All',l]
    tn = ovaCM['All','All']

    ## CALCULATE METRICS FOR THAT LVL AND SUM WITH PREVIOUS PART 1
    acc = c(acc,((tp+tn)/(tp+tn+fp+fn)))
    sen = c(sen,(tp/(tp+fn)))
    spe = c(spe,(tn/(tn+fp)))
    pre = c(pre,(tp/(tp+fp)))

    # AUXILIAR CALCULATION FOR MCC AND F1S
    sen_unq = tp/(tp+fn)
```

```
    pre_unq = tp/(tp+fp)
    f1s_unq = 2*((sen_unq*pre_unq)/(sen_unq+pre_unq))
    mcc_num = (tp*tn) - (fp*fn)
    mcc_den = (tp+fp)*(tp+fn)*(tn+fp)*(tn+fn)

    ## CALCULATE METRICS FOR THAT LVL AND SUM WITH PREVIOUS PART 2
    f1s = c(f1s,f1s_unq)
    mcc = c(mcc,(mcc_num/sqrt(mcc_den)))
  }
  ## OVA :: RESULT FOR EACH j VARIABLE BASED ON A MEAN
  ##FROM ALL OVA MATRIX FROM THAT VARIABLE i.e. ALL LEVELS OVA MATRIX
  cmACC_ova <- 100*(sum(acc, na.rm = T)/len_lvls)
  cmSEN_ova <- 100*(sum(sen, na.rm = T)/len_lvls)
  cmSPE_ova <- 100*(sum(spe, na.rm = T)/len_lvls)
  cmPRE_ova <- 100*(sum(pre, na.rm = T)/len_lvls)
  cmF1S_ova <- 100*(sum(f1s, na.rm = T)/len_lvls)
  cmMCC_ova <- 100*(sum(mcc, na.rm = T)/len_lvls)

  ### PRINT - OVA METRICS
  cat(paste('+ ACCURACY OVA: ', round(cmACC_ova,2),'\n'))
  cat(paste('+ SENSIBILITY OVA: ', round(cmSEN_ova,2),'\n'))
  cat(paste('+ SPECIFICITY OVA: ', round(cmSPE_ova,2),'\n'))
  cat(paste('+ PRECISION OVA: ', round(cmPRE_ova,2),'\n'))
  cat(paste('+ F1-SCORE OVA: ', round(cmF1S_ova,2),'\n'))
  cat(paste('+ MCC OVA: ', round(cmMCC_ova,2),'\n'))

  ### SAVE INTO DATAFRAME RESULT
  metrics[j,'accuracyOVA'] <- round(cmACC_ova,2)
  metrics[j,'sensibilityOVA'] <- round(cmSEN_ova,2)
  metrics[j,'specificityOVA'] <- round(cmSPE_ova,2)
  metrics[j,'precisionOVA'] <- round(cmPRE_ova,2)
  metrics[j,'f1-scoreOVA'] <- round(cmF1S_ova,2)
  metrics[j,'mccOVA'] <- round(cmMCC_ova,2)

}

# SCORING RULES CALC
for(j in to_predict){
  ## AUXILIAR VARIABLES
  aux_prob <- predProbList[[j]]
  cor_test <- reference[j]
  states <- colnames(aux_prob)

  ## TERMS FOR THE FORMULAS :: init
  pc = 0
  pj_2 = 0
  sum_pj = 0

  ## TERMS FOR :: SPHERICAL PAYOFF
  smp_spher_payoff = 0
  srSP = 0

  ## TERMS FOR :: BRIER LOSS
```

```
    smp_brier_loss = 0
    srBL = 0

    ## TERMS FOR :: LOG LOSS
    smp_log_loss = 0
    srLL = 0

    ## LOOP THROUGH ALL THE N SAMPLES PREDICTED
    for(i in 1:N){
      correct_state = cor_test[i,]
      ## TERMS CALCULATION
      pc = aux_prob[i,correct_state]
      sum_pj = 0
      for(s in states){
        pj_2 = aux_prob[i,s] * aux_prob[i,s]
        sum_pj = sum_pj + pj_2
      }

      ## SCORING RULE CALC FOR EACH SAMPLE - SP
      smp_spher_payoff = pc/sqrt(sum_pj)
      srSP = srSP + smp_spher_payoff

      ## SCORING RULE CALC FOR EACH SAMPLE - BL
      #smp_brier_loss = 1-(2*pc+sum_pj)
      smp_brier_loss = 2*pc-sum_pj
      srBL = srBL + smp_brier_loss

      ## SCORING RULE CALC FOR EACH SAMPLE - LL
      smp_log_loss = (-log(pc))
      srLL = srLL + smp_log_loss
    }

    ## RESULT - SPHERICAL PAYOFF (MOAC)
    srSP = srSP/N
    srBL = srBL/N
    srLL = srLL/N

    ### PRINT - SCORING RULES
    cat(paste('#### SCORING RULES METRICS - ',j,' Variable ####\n'))
    cat(paste('+ SPHERICAL PAYOFF: ', round(srSP,2),'\n'))
    cat(paste('+ BRIER LOSS: ', round(srBL,2),'\n'))
    cat(paste('+ LOG LOSS: ', round(srLL,2),'\n'))

    ### SAVE INTO DATAFRAME RESULT
    metrics[j,'sphericalPayoff'] <- round(srSP,2)
    metrics[j,'brierLoss'] <- round(srBL,2)
    metrics[j,'logLoss'] <- round(srLL,2)
  }

  # RETURN
  ret_list <- list('cmList' = cm_list, 'ovaList' = ova_list, 'eval' = metrics)

}
```