Felipe Schneider Costa

APRENDIZAGEM ESTRUTURAL DE REDES BAYESIANAS PELO MÉTODO DE MONTE CARLO E CADEIAS DE MARKOV

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Ciência da Computação
Orientadora:
Prof.ª Dr.ª Silvia Modesto Nassar

Florianópolis, SC
2013

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Costa, Felipe Schneider
APRENDIZAGEM ESTRUTURAL DE REDES BAYESIANAS PELO MÉTODO DE MONTE CARLO E CADEIAS DE MARKOV / Felipe Schneider Costa ; orientadora, Silvia Modesto Nassar - Florianópolis, SC, 2013.
90 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Ciência da Computação. 2. Redes Bayesianas. 3. Aprendizagem de máquina. 4. Monte Carlo e Cadeias de Markov. I. Nassar, Silvia Modesto . II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Felipe Schneider Costa

APRENDIZAGEM ESTRUTURAL DE REDES BAYESIANAS PELO MÉTODO DE MONTE CARLO E CADEIAS DE MARKOV

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Ciência da Computação, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Local, 04 de dezembro de 2013.

_____
Prof. Ronaldo dos Santos Mello, Dr.
Coordenador do Curso

Banca Examinadora:

_____
Prof.ª Silvia Modesto Nassar, Dr.ª
Orientadora

_____
Prof. Julio Cesar Nievola, Dr.
Pontifícia Universidade Católica do Paraná

_____
Prof.ª Maria Marlene de Souza Pires, Dr.ª

_____
Prof. Paulo José de Freitas Filho, Dr.

Dedico este trabalho aos meus pais, Angelo (em memória), que sempre me apoiou por considerar o estudo como o fator principal no desenvolvimento de uma pessoa, e Ester, por todo apoio e carinho; e a Jacque, esposa, companheira e amiga, por estar sempre presente, me incentivando e apoiando antes e durante a pesquisa.

AGRADECIMENTOS

Gostaria de agradecer a toda minha família por estarem sempre presentes em minha vida e por seu apoio.

À professora Silvia que, além de orientadora, sempre foi uma grande amiga.

À professora Marlene, aos amigos Aldo e Gisara e aos professores Selner e Obelheiro, por todo apoio e confiança.

Agradeço também a todos os professores e funcionários do PPGCC e ao amigo Nilo pelo incentivo na reta final.

“Um passo à frente e você não está mais no mesmo lugar.”

Chico Science

RESUMO

Esta dissertação aborda a aplicação dos métodos de Monte Carlo via Cadeias de Markov na aprendizagem de estruturas de redes Bayesianas. Estes métodos têm se mostrado extremamente eficientes nos cálculos aproximados de problemas nos quais é impossível obter uma solução exata. Neste sentido, apresenta um método para gerar estruturas de redes Bayesianas a partir dos dados para que possam ser utilizadas para realizar consultas sobre o domínio do problema e também que permitam extrair conhecimento sobre o problema através dos modelos gráficos gerados. Inicialmente, através do uso de técnicas de verificação de independência condicional entre os nós da rede, alguns vértices (conexões entre os nós) da estrutura inicial foram fixados e não mais alterados, visando minimizar o uso de recursos computacionais. Após fixar esses vértices, o próximo passo consistiu em construir uma estrutura inicial de rede (conectar os demais nós da rede não fixados no passo anterior) a ser alterada durante toda a execução do algoritmo. Para isso, foram utilizados algoritmos de busca heurística. De posse de um modelo inicial de rede e seguindo o fluxo dos métodos de Monte Carlo e Cadeias de Markov, a próxima etapa alterava esse modelo, a cada iteração do algoritmo, de forma aleatória, visando encontrar o modelo que melhor representasse os dados. Os algoritmos de geração de amostras de rede utilizados nessa etapa selecionavam dois nós e uma operação a ser realizada no vértice de conexão entre esses nós (incluir, excluir ou inverter), sempre de forma aleatória. Depois de verificar se a operação realizada na estrutura atual da rede gerava uma rede válida (sem ciclos), a rede era aceita como novo estado da cadeia. Finalmente, para comparar os modelos de rede e selecionar o melhor entre eles, foram utilizadas métricas de *score*. Analisando as redes geradas durante as execuções do algoritmo, juntamente com os dados capturados na submissão dos casos de teste, pôde-se concluir que os resultados mostraram-se muito satisfatórios, devido, principalmente, às taxas de erros apresentadas nas matrizes de classificação. Como exemplo, na submissão de um dos conjuntos de testes a uma das redes gerada pelo algoritmo, apenas 7% (sete) dos dados foram classificados incorretamente. Pode-se crer que os bons resultados obtidos devem-se ao processo utilizado na coleta de modelos de rede, no qual foram salvos os melhores modelos durante toda a execução do programa.

Palavras-chave: Redes Bayesianas. Aprendizagem de máquina. Monte Carlo, Cadeias de Markov.

ABSTRACT

This paper discusses the application of the methods of Markov Chain Monte Carlo in the learning of structures of Bayesian networks. These methods have proved to be extremely effective in approximate calculations of problems in which it is impossible to obtain an exact solution. In this sense, it presents a method for generating structures of Bayesian networks from data that can be used to perform queries on the problem domain and also for extracting knowledge about the problem through the graphic models generated. Initially, through the use of verification techniques for conditional independence between the network nodes, some vertices (connections between nodes) of the initial structure were fixed and not altered in order to minimize the use of computational resources. After fixing these vertices, the next step was to build an initial network structure (connect other network nodes not set in the previous step) to be changed throughout the execution of the algorithm. For this, heuristic search algorithms are used. With this initial network model and following the flow of the Monte Carlo and Markov chains methods, the next step alter this model, in each iteration of the algorithm, randomly, aiming to find the model that best represents the data. The algorithms for generating samples of network used in this step selected two nodes and an operation to be performed at the vertice of connection between these nodes (add, delete or reverse), always randomly. After checking that the operation performed on the current network structure generated a valid network (without cycles), the network was accepted as a new state of the chain. Finally, to compare the network models and select the best among them, metrics score are used. Analyzing the networks generated during the execution of the algorithm, along with the data captured in the submission of test cases, it can be concluded that the results were very satisfactory, mainly due to error rates presented in the matrix of classification. As an example, submission of one of the test sets to the network generated by the algorithm, only 7% (seven) of data were misclassified. It is believed that the good results are due to the process used to collect network models, where it saves the best models throughout the execution of the program.

Keywords: Bayesian Networks. Machine learning. Monte Carlo Markov Chain.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

SUMÁRIO

1. INTRODUÇÃO

Alan Turing teve um papel fundamental no desenvolvimento da Inteligência Artificial (IA). Ele foi o responsável pela criação do teste de Turing (TURING, 1950), projetado para fornecer uma definição operacional satisfatória de inteligência, propondo um método para verificar se máquinas são capazes de pensar. A capacidade de aprendizagem nas máquinas, um dos requisitos avaliados pelo teste, pode ser definida como a capacidade de se adaptar a novas circunstâncias, além de detectar e extrapolar padrões. Partindo desse ponto de vista, para que um sistema seja considerado inteligente, ele deve ser capaz de alcançar o melhor resultado possível, de acordo com seu objetivo, ou quando há incerteza, o melhor resultado esperado (RUSSEL; NORVIG, 2004).

Aprendizagem e inteligência estão intimamente relacionadas. Em geral se espera que um sistema capaz de aprender mereça ser chamado de inteligente e, inversamente, um sistema que é considerado inteligente deve, entre outras coisas, ser capaz de aprender. Aprender tem a ver com o autoaprimoramento do comportamento futuro baseado em experiência passada. Mais precisamente, de acordo com o ponto de vista da inteligência artificial (IA), aprendizagem pode ser informalmente definida como a aquisição de novos conhecimentos e habilidades e a sua incorporação em atividades futuras do sistema, de forma a levar a uma melhora no seu desempenho (WEISS, 1999).

A aprendizagem de máquina mostrou ser uma área fértil de pesquisa, produzindo uma série de algoritmos para a solução de problemas. Esses algoritmos variam com relação aos objetivos, à disponibilidade de dados de treinamento, às estratégias de aprendizagem e à linguagem de representação do conhecimento que eles empregam. Entretanto, todos eles aprendem através de buscas, num espaço de conceitos possíveis, para encontrar uma generalização aceitável (LUGER, 2004).

Basicamente, é possível afirmar que a aprendizagem de máquina se divide em duas linhas: a aprendizagem supervisionada (aprender a partir de exemplos das entradas e saídas do sistema) e não supervisionada (aprendizagem apenas através dos padrões da entrada). Dentro de cada linha, dependendo da abordagem, ainda se pode adotar estratégias de aprendizagem indutiva (ou aprendizagem através de exemplos), aprendizagem por hábito ou aprendizagem de conceitos.

Os métodos de aprendizagem indutiva são baseados no princípio de que se for encontrada uma função capaz de mapear corretamente um

grande conjunto de dados de treinamento em classificações, então ela também mapeará corretamente dados não observados anteriormente. Agindo dessa forma a função será capaz de *generalizar* a partir de um conjunto de dados de treinamento (COPPIN, 2010).

Na aprendizagem indutiva supervisionada cada exemplo é descrito pelos valores de um conjunto de atributos e sempre possui uma classe (ou rótulo) associada. A ideia geral consiste em induzir um conceito utilizando esses exemplos rotulados, denominado conjunto de exemplos de treinamento, realizando generalizações e especializações, tal que o conceito (hipótese) induzido seja capaz de predizer a classe (rótulo) de futuros exemplos. Nesse caso, o conceito induzido é visto como um classificador (SANCHES, 2003).

Com relação ao tipo do atributo classe, caso ele seja contínuo, o problema de indução é conhecido como regressão e, caso seja discreto, o problema é conhecido como classificação.

Na aprendizagem não supervisionada, os exemplos não possuem uma classe correspondente. Nesse caso o indutor analisa os exemplos fornecidos e tenta determinar padrões entre eles (CHEESEMAN; STUTZ, 1996).

Num sentido prático, essas técnicas de aprendizagem têm sido bastante utilizadas na mineração de dados. A mineração de dados busca descobrir técnicas para descrever padrões estruturais em dados, como ferramenta para explicá-los e fazer previsões a partir deles. Os dados assumirão a forma de um conjunto de exemplos, como clientes que mudam sua preferência em relação a determinado produto, por exemplo. A saída terá a forma de previsões sobre esses exemplos para prever, por exemplo, se determinado cliente alterou ou não sua preferência. A saída também pode incluir uma descrição sobre a estrutura utilizada para classificar exemplos desconhecidos e que serve para fornecer uma representação explícita do conhecimento adquirido (WITTEN; WITTEN, 2011).

Sabe-se que muitas tarefas, incluindo diagnóstico de falhas, reconhecimento de padrões e previsões, podem ser vistas como classificação (CHENG; GREINER, 1999). A classificação é uma tarefa base em análise de dados e reconhecimento de padrões que requer a construção de um classificador, ou seja, uma função que atribua uma etiqueta de classe a exemplos descritos por um conjunto de variáveis. A inferência de classificadores em conjuntos de dados com casos pré-classificados é um problema central na aprendizagem de máquina. Várias abordagens para esse problema se baseiam em representações

funcionais, tais como árvores de decisão, redes neurais e regras (FRIEDMAN; GEIGER, 1997).

Pode-se então dizer que mesmo de posse de todos esses recursos, em um processo de aprendizagem, os sistemas não têm acesso a toda a verdade sobre seu ambiente e precisam, dessa forma, agir sob incerteza. A inferência estatística, através de métodos probabilísticos, é uma ferramenta poderosa, pois permite aos sistemas raciocinar sob incerteza. A Teoria da Probabilidade atribui um grau de crença (entre 0 e 1) a cada alternativa, tornando possível aos agentes lidarem com a incerteza através da computação da evidência observada de probabilidades posteriores, para proposições de consulta (RUSSEL; NORVIG, 2004).

Utilizando os fundamentos da Teoria da Probabilidade, em 1988, Judea Pearl criou o conceito denominado de Redes Bayesianas de Crença (PEARL, 1988). Por representarem o formalismo semântico da probabilidade (probabilidade conjunta) de uma forma compacta e clara aos olhos humanos - a estrutura gráfica (FRIEDMAN; GOLDSZMIDT, 1997) - e trabalharem com incertezas em sistemas inteligentes do mundo real, essas redes passaram a desempenhar um papel importante em uma vasta área de aplicações a partir da década de noventa (BINDER, 1997).

Normalmente a estrutura de uma rede Bayesiana é construída por um especialista do domínio. Em função de fatores como tempo e custo pode ser vantajoso utilizar um processo automático de criação dessa estrutura. Embora a construção da estrutura da rede Bayesiana a partir dos dados traga muitos benefícios, não é uma tarefa trivial. Várias pesquisas têm sido realizadas no sentido de aprimorar os resultados obtidos. Isso acontece porque o tamanho do conjunto das possíveis estruturas da rede (dentre os quais será escolhido o que melhor representa os dados) cresce exponencialmente conforme o número de variáveis aumenta, e dessa forma é classificado como NP-Completo (CHICKERING; HECKERMAN; MEEK, 2004). A alternativa então é fazer uso de algoritmos de busca, que com um esforço computacional aceitável consigam encontrar uma solução aproximada, ou seja, uma estrutura que represente bem o conhecimento contido nos dados.

Dessa forma, o foco deste trabalho se concentra na tarefa de aprendizagem não supervisionada de estruturas de redes Bayesianas, utilizando o método de Monte Carlo e Cadeias de Markov e apoiado por um processo de descoberta de relações de interdependência entre as variáveis, ou seja, nas ligações entre os nós da rede. A intenção é minimizar o esforço computacional necessário para a construção da estrutura, bem como a construção de um modelo que seja o mais simples possível.

1.1 JUSTIFICATIVA

Normalmente a estrutura da rede Bayesiana, que representa graficamente a dependência entre as variáveis do domínio, ou seja, o conhecimento causal sobre o domínio, é construída por um especialista. Entretanto, nos casos em que as bases de dados possuem uma grande quantidade de variáveis, o tempo que os especialistas precisam para "descobrir" o conhecimento através da análise dos dados torna o processo impraticável. Em algumas dessas situações, como em modelos celulares com milhares de variáveis, o ganho fornecido pelo resultado do processo pode estar apenas na descoberta de relacionamentos desconhecidos entre as variáveis do modelo, sem que para isso todo o modelo precise ser considerado válido.

Diante desses problemas fica claro que a criação de um processo automático de aprendizagem dos relacionamentos entre as variáveis do domínio pode ser de grande utilidade, podendo ser utilizado para a descoberta de conhecimento em bases de dados, bem como para a criação de modelos que possam ser utilizados para responder questionamentos sobre problemas específicos do domínio da aplicação.

O presente trabalho se enquadra na área de *Ciência da Computação*, mais especificamente na linha de pesquisa *Inteligência Computacional,* uma vez que um dos objetivos desta linha de pesquisa consiste em desenvolver pesquisas na área de criação de teorias e modelos para a aquisição de conhecimento.

1.2 OBJETIVO GERAL

Este trabalho tem como objetivo geral a criação de um método para aprendizagem de estruturas de redes Bayesianas, de forma não supervisionada, utilizando simulação de Monte Carlo via Cadeias de Markov (MCMC).

1.3 OBJETIVOS ESPECÍFICOS

Os objetivos específicos são:

- investigar técnicas necessárias para moldar o relacionamento entre os nós de uma rede, utilizando conceitos de independência condicional;
- construir um algoritmo que utilize os conceitos pesquisados;
- avaliar a adequação do método proposto para o algoritmo MCMC.

1.4 PROCEDIMENTOS METODOLÓGICOS

São necessárias várias etapas para desenvolver um algoritmo não supervisionado de aprendizagem de estrutura de redes Bayesianas utilizando métodos de Monte Carlo e Cadeias de Markov. Esses métodos, de forma geral, simulam um experimento com a finalidade de determinar propriedades probabilísticas (encontrando uma aproximação da medida de probabilidade através de tentativas) de uma população, a partir de uma amostragem aleatória dos componentes dessa população.

Dessa forma, a pesquisa passa pelo estudo de técnicas de verificação de independência condicional entre as variáveis visando minimizar o esforço computacional necessário para a construção da estrutura e também buscando construir um modelo de rede que seja o mais simples possível. Os testes estatísticos utilizados nesta etapa tiveram como objetivo fixar alguns vértices entre os nós na estrutura inicial da rede, reduzindo assim a quantidade de vértices que seriam alterados durante a execução do algoritmo. Para construir a estrutura inicial das redes foram utilizados algoritmos de busca heurística e para comparar os modelos de rede e selecionar o melhor entre eles utilizou-se métricas de *score*. Uma vez que a pesquisa está centrada no uso do método de Monte Carlo e Cadeias de Markov, foi necessário fazer uso de algoritmos de geração de amostras. Nessa fase, eram selecionados dois nós e uma operação a ser realizada no vértice de conexão entre esses nós (incluir, excluir ou inverter), sempre de forma aleatória. Depois de verificar se a operação realizada na estrutura atual da rede gerava uma rede válida (sem ciclos), a nova rede era aceita como novo estado da cadeia. Como restrições ao tema de pesquisa, não foram consideradas variáveis contínuas e dados incompletos[1].

1.5 CLASSIFICAÇÃO DA PESQUISA

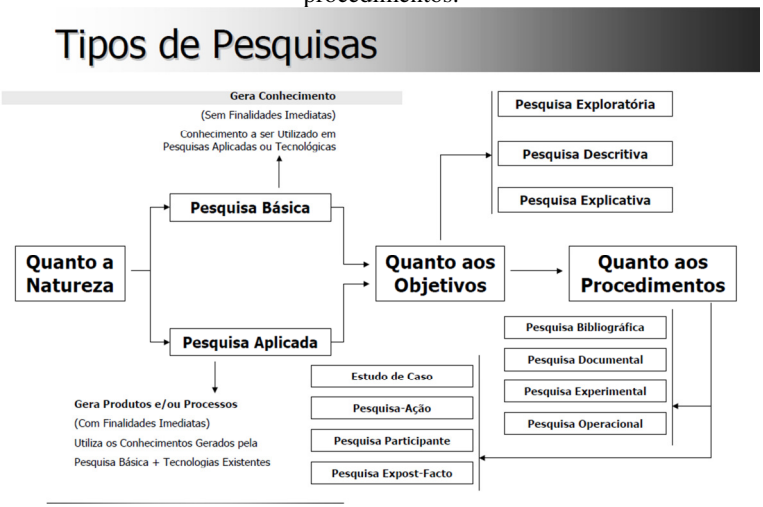
A metodologia utilizada na pesquisa, quanto a sua natureza, é caracterizada como aplicada, pois segundo Silva e Menezes (2001) a pesquisa aplicada objetiva gerar conhecimentos para aplicação prática e dirigidos à solução de problemas específicos, envolvendo verdades e interesses locais. A pesquisa aplicada ou tecnológica (Figura 1) tem como objetivo alcançar a inovação em um produto ou processo, frente a uma demanda ou necessidade preestabelecida (JUNG, 2003). Neste tipo

[1] Dados incompletos referem-se a um exemplo com ***n*** variáveis sendo algumas delas sem valor definido.

de pesquisa, o resultado a ser medido é a solução concreta do problema proposto, representado por um novo produto ou um novo processo e a sua aceitação pelo mercado consumidor. No que diz respeito aos procedimentos esta pesquisa se classifica como experimental já que o método desenvolvido provoca alterações no ambiente sob estudo de forma a observar se estas intervenções produzem os resultados esperados (WAZLAWICK, 2008). Com relação à abordagem do problema é do tipo qualitativa e com relação aos objetivos é exploratória.

- Quanto à Natureza : pesquisa aplicada
- Quanto aos objetivos : pesquisa exploratória
- Quanto aos procedimentos : pesquisa experimental
- Quanto ao método de abordagem : pesquisa qualitativa.

Figura 1 - Classificação da pesquisa quanto à natureza, objetivo e aos procedimentos.



Fonte: Jung (2003).

2 FUNDAMENTAÇÃO TEÓRICA

2.1 REDES BAYESIANAS

O conceito de dependência é extremamente importante na teoria da probabilidade. Dois eventos são independentes se a probabilidade de ocorrer o evento A não interfere na ocorrência ou não do evento B.

Uma rede Bayesiana de crença é um grafo orientado acíclico, no qual os nós representam evidências ou hipóteses e no qual um arco que conecta dois desses nós representa a dependência entre eles (COPPIN, 2010).

2.1.1 Incerteza e Probabilidade

A teoria da probabilidade estipula uma alternativa à lógica, para lidar com os domínios de julgamento, atribuindo a cada sentença um grau de crença entre 0 e 1, proporcionando assim um método para resumir a incerteza. A probabilidade associada a uma informação, na ausência de quaisquer outras informações, é chamada de probabilidade a *priori* ou incondicional. Quando o agente obtém informações relativas às variáveis aleatórias, as probabilidades são definidas como a *posteriori* ou condicionais. De posse das probabilidades a *posteriori,* o agente é capaz de responder proposições de consulta através da computação das evidências observadas. Esse método é chamado de inferência probabilística (RUSSEL; NORVIG, 2004).

Para garantir que as probabilidades de um determinado par de variáveis some 1, é preciso utilizar um processo chamado *normalização*, no qual as probabilidades a *posteriori* são divididas por um valor fixo (COPPIN, 2010).

O cálculo das probabilidades, quando o domínio do problema envolve um grande número de variáveis, pode ter um custo muito alto, por isso a distribuição conjunta de probabilidades não fornece um algoritmo viável para implementação, em função do tempo e da quantidade de dados necessários, além do fato de que não leva em conta independência entre variáveis.

2.1.2 Redes Bayesianas

O teorema de Bayes, apresentado na Equação 1, base de todos os sistemas modernos de Inteligência Artificial para inferência probabilística, permite que através de asserções de independência seja

possível simplificar expressões, descobrindo novas relações de dependência entre as variáveis. Essa simplificação é possível porque essas asserções, que se baseiam no conhecimento sobre o domínio do problema, reduzem drasticamente a quantidade de informações necessárias para especificar as distribuições de probabilidade (RUSSEL; NORVIG, 2004).

$$\boldsymbol{P(B|A) = \frac{P(A|B).P(B)}{P(A)}} \qquad \textbf{(1)}$$

onde

P(B|A) = probabilidade de ocorrer o evento B dado que o evento A ocorreu

P(A|B) = probabilidade de ocorrer o evento A dado que o evento B ocorreu

P(A) = probabilidade do evento A

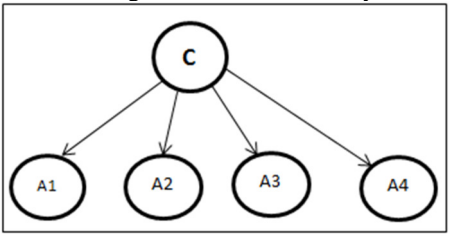
P(B) = probabilidade do evento B

B = evento B

A = evento A

Redes Bayesianas (Pearl, 1988) são ferramentas poderosas para a representação do conhecimento e inferência em condições de incerteza que só foram consideradas como classificadores com a descoberta do classificador ingênuo de Bayes. Surpreendentemente eficaz, o classificador ingênuo de Bayes é basicamente um tipo simples de rede Bayesiana em que todas as variáveis são consideradas independentes umas das outras, dado o nó de classificação (CHENG; GREINER, 1999).

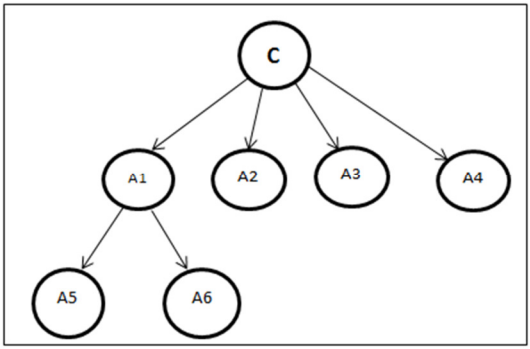
Figura 2 - Estrutura gráfica de uma rede Bayesiana ingênua.



Uma rede Bayesiana é um modo sistemático de representar as relações de independência condicional entre as variáveis, através de uma estrutura de dados (grafos orientados), onde cada nó é identificado com

informações de probabilidade quantitativa. Os grafos são dirigidos e acíclicos, nos quais os nós representam variáveis; os arcos significam a existência de influência causal direta entre as variáveis ligadas; e a intensidade destas influências é expressa por probabilidades condicionais (PEARL, 1988). São utilizadas para representar o conhecimento do domínio por meio de relações de dependência entre variáveis aleatórias (graficamente), probabilidades a priori e probabilidades condicionais entre variáveis.

Figura 3 - Estrutura gráfica de uma rede Bayesiana hierárquica (com nós intermediários).



As redes Bayesianas permitem calcular eficientemente probabilidades a *posteriori* de qualquer variável aleatória (inferência), por meio de uma definição recursiva do teorema de Bayes.

O classificador ingênuo de Bayes, apresentado na Equação 2, é a forma mais simples de representação de redes Bayesianas, na qual todas as variáveis são independentes, dado o valor da variável classe. Esta condição é chamada de independência condicional. Apesar de a hipótese de independência condicional raramente ser verdadeira (ZHANG, 2004), o classificador ingênuo de Bayes, surpreendentemente, superou muitos classificadores sofisticados em um grande número de conjuntos de dados, especialmente onde as variáveis não são fortemente correlacionadas (CHENG; GREINER, 1999).

$$\boldsymbol{d = argmax_{c \in C} \propto P(c) \prod_{i=1}^{n} P(a_i|c)} \qquad \textbf{(2)}$$

onde

d = classe de saída com a máxima probabilidade a posteriori
a = vetor de valores das variáveis do modelo

C = variável de saída
c = valores que a variável de saída pode assumir
n = número de variáveis
a_i = i-ésima variável

Na representação gráfica de uma rede Bayesiana ingênua (Figura 2), todos os nós estão conectados ao nó de classificação (nó "C" na figura) e nenhuma outra conexão é permitida. Essa suposição de independência condicional de todos os nós, dado o nó de saída, não existe em uma rede Bayesiana hierárquica (Figura 3), conforme a Equação 3:

$$\boldsymbol{P(X_1, \dots, X_n) = \prod_{i=1}^{n} P(X_i | pa(X_i))} \quad \textbf{(3)}$$

onde

$P(X_i|pa(X_i))$ = probabilidade da variável dado seus pais
$X_1,\dots,X_n$ = conjunto de variáveis
$pa(X_i)$ = conjunto de variáveis pais de *X_i*
n = número de variáveis
i = i-ésima variável

2.2 APRENDIZAGEM DE MÁQUINA

Aprendizagem de máquina é o foco de pesquisas relacionadas à análise automatizada de dados em larga escala. Historicamente as pesquisas têm se concentrado nos modelos biologicamente inspirados e as metas em longo prazo, de grande parte da comunidade, são orientadas para a produção de modelos e algoritmos que possam processar a informação, tão bem como sistemas biológicos (BARBER, 2012).

Ela engloba estudos de métodos computacionais para adquirir novos conhecimentos, novas habilidades e novos meios de organizar o conhecimento já existente. Um sistema capaz de aprender é aquele que se modifica automaticamente, no sentido de que ele possa fazer as mesmas tarefas sobre um mesmo domínio de conhecimento, de uma maneira cada vez mais eficiente (SIMON, 1983).

2.2.1 Tipos de aprendizagem de máquina

De um modo geral os dois tipos principais de aprendizagem de máquina são a aprendizagem supervisionada e a não supervisionada.

Nos dois casos o interesse principal é por métodos que consigam generalizar bem dados não vistos anteriormente. Nesse sentido, os dados utilizados são divididos de forma que uma parte seja utilizada para treinar o sistema e outra para testar o desempenho do modelo criado no treinamento.

No tipo de aprendizagem denominada *supervisionada*, cada exemplo é descrito pelos valores de um conjunto de atributos e possui, obrigatoriamente, uma classe associada. A classe, também conhecida como rótulo do exemplo, é um atributo especial que descreve uma instância do fenômeno de interesse, ou seja, o conceito que se deseja induzir. A ideia geral consiste em induzir um conceito utilizando esses exemplos rotulados, denominado conjunto de exemplos de treinamento, realizando generalizações e especializações, tal que o conceito (hipótese) induzido seja capaz de predizer a classe (rótulo) de futuros exemplos. Nesse caso, o conceito induzido é visto como um classificador (SANCHES, 2003).

Na aprendizagem *não supervisionada*, os exemplos não possuem uma classe correspondente. Nesse caso, o indutor analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira (CHEESEMAN; STUTZ, 1996). Após a determinação dos agrupamentos, normalmente, é necessária uma análise para determinar o que cada agrupamento significa no contexto do problema que está sendo analisado. Portanto, a escolha de qual tipo de aprendizagem indutiva (supervisionada ou não supervisionada) utilizar, depende dos exemplos estarem ou não rotulados com o atributo classe.

Existem várias estratégias de aprendizagem, como aprendizagem por hábito, por instrução, por dedução e aprendizagem por analogia.

Os métodos de aprendizagem indutiva são baseados no princípio de que se for encontrada uma função capaz de mapear corretamente um conjunto de dados de treinamento em classificações, então ela também mapeará corretamente dados não observados anteriormente (COPPIN, 2010).

Caso o tipo do atributo classe seja contínuo, o problema de indução é conhecido como regressão e, caso seja discreto, o problema é conhecido como classificação.

2.2.2 Aprendizagem da estrutura de redes

Os algoritmos de aprendizagem de redes Bayesianas estão divididos entre aprendizagem de parâmetros, que diz respeito à

aprendizagem das distribuições de probabilidade condicional, e aprendizagem da estrutura do grafo dirigido acíclico. A aprendizagem dos parâmetros é relativamente simples se a estrutura da rede ideal for conhecida, pois se recai em um problema de maximização da função de verossimilhança (CHENG; GREINER, 2001).

Existem várias técnicas utilizadas pelos algoritmos de aprendizagem de estruturas. Uma delas seleciona a rede que melhor define os dados com base em uma medida de pontuação, outra procura identificar as relações de independência condicional existentes entre os vértices, através do uso de testes estatísticos como Qui-quadrado e Informação Mútua, e a partir delas encontrar a melhor estrutura da rede. Esses métodos são conhecidos como algoritmos baseados em independência condicional (CHENG; GREINER, 2001).

Uma abordagem bastante utilizada para a construção de estruturas de redes é iniciar com um modelo que não contém nenhum vínculo e ir adicionando os nós pais, sempre ajustando os parâmetros e medindo o modelo criado através de alguma métrica. Pode-se utilizar também um modelo inicial e, utilizando um algoritmo de subida da encosta ou de têmpera simulada, fazer alterações na estrutura retornando os parâmetros após cada alteração na estrutura. As modificações podem incluir inversão, adição ou eliminação de arcos (RUSSEL; NORVIG, 2004).

Outro método utilizado passa por definir dois componentes no algoritmo: uma função para a avaliação de uma rede com base nos dados e um método para pesquisar através do espaço de redes possíveis. A qualidade de uma determinada rede é medida pela probabilidade dos dados fornecidos por essa rede. Calculam-se as probabilidades que a rede atribui a cada instância e multiplicam-se essas probabilidades em conjunto ao longo de todas as instâncias. Na prática, isso rapidamente produz números muito pequenos para serem devidamente representados, por isso utiliza-se a soma dos logaritmos das probabilidades, em vez de seu produto. A quantidade resultante é a probabilidade logarítmica da rede condicionada aos dados (WITTEN; FRANK, 2011).

2.2.2.1 Estimação de máxima verossimilhança

O método de máxima verossimilhança é uma das técnicas mais populares para derivar estimadores. O estimador de máxima verossimilhança é o valor para o qual a amostra observada é mais provável (CASELLA; BERGER, 2010). Se os $\boldsymbol{x_i}$ são independentes e identicamente distribuídos (iid), segundo uma distribuição de

probabilidade ***f(.)*** com parâmetros $\boldsymbol{\theta}$, então a verossimilhança completa dos dados é definida pela Equação 4.

$$\boldsymbol{L(\theta|X) = L(\theta_1, \dots, \theta_K|x_1, \dots, x_n) = \prod_{i=1}^{n} f(x_i|\theta_1, \dots, \theta_k)} \qquad \textbf{(4)}$$

onde

n = número de variáveis
i = índice da variável
k = índice do parâmetro
K = número de parâmetros
x_i *= i-ésima variável*
θ_k *= k-ésimo parâmetro da variável*

2.2.2.2 Métricas de *score* para seleção de modelos de redes

Uma vez construído o modelo da rede, é preciso escolher, dentre todos os modelos gerados, os que melhor representam os dados. Uma prática bastante utilizada é penalizar a complexidade da rede, e assim selecionar os modelos mais simples. Para realizar essa tarefa são utilizadas métricas de *score*.

O processo de seleção de um modelo de rede, de uma forma geral, busca a estrutura de rede que tenha a maior probabilidade considerando os dados, para em seguida utilizar essa estrutura para responder a consultas. Em particular, procura-se uma estrutura de rede que satisfaça

$$\boldsymbol{G^* = \mathbf{argmax}_G\, P(G|D)} \qquad \textbf{(5)}$$

na qual $\boldsymbol{G}$ é a estrutura da rede e $\boldsymbol{D}$ é o conjunto de dados. Isto é o mesmo que procurar por uma estrutura que satisfaça

$$\boldsymbol{G^* = \mathbf{argmax}_G \frac{P(D|G)P(G)}{P(D)}} \qquad \textbf{(6.1)}$$

e uma vez que a probabilidade marginal ***P(D)*** não depende da estrutura da rede, pode-se simplificar a Equação 6.1 da seguinte forma

$$\boldsymbol{G^* = \mathbf{argmax}_G\, P(D|G)P(G)} \qquad \textbf{(6.2)}$$

A equação 6.2 também pode ser escrita como

$$\boldsymbol{log}(\boldsymbol{G}^{*}) = \mathbf{argmax}_{\boldsymbol{G}} \ \ \boldsymbol{logP}(\boldsymbol{D}|\boldsymbol{G}) + \boldsymbol{logP}(\boldsymbol{G}) \qquad \textbf{(7)}$$

Este *score* é conhecido como "*Bayesian scoring measure"* (DARWICHE, 2009). Como a probabilidade *a priori* ***P(G)*** do modelo gráfico normalmente é igual para todos os modelos, pode-se desconsiderá-la nos cálculos (Equação 5).

Existem outros modelos de *score* utilizados para selecionar modelos de rede, como por exemplo MDL (*Minimum Description Length)*, muito semelhante ao *score* BIC (*Bayesian Information Criteria*). Os dois últimos incorporam um termo para penalizar a complexidade do modelo gráfico, conforme a Equação 8.

$$\boldsymbol{BIC}\ (\boldsymbol{G}|\boldsymbol{D}) = \boldsymbol{log}\ \boldsymbol{P}(\boldsymbol{G}|\boldsymbol{D}) - \frac{\boldsymbol{log}\ \boldsymbol{N}}{\mathbf{2}}\ \boldsymbol{dim}(\boldsymbol{G}) \qquad \textbf{(8)}$$

Onde ***dim(G)*** representa o número de parâmetros independentes nas tabelas de probabilidade conjunta da rede e ***N*** é o número de casos da base de dados, conforme as Equações 9 e 10.

$$\left|\left|\boldsymbol{G}\right|\right| = \sum ||\boldsymbol{X_i U_i}|| \qquad \textbf{(9)}$$

onde

$$||\boldsymbol{X_i U_i}|| = \left|\left|(\boldsymbol{X_i^{\#}} - \mathbf{1})\boldsymbol{U_i^{\#}}\right|\right| \qquad \textbf{(10)}$$

onde

X_i = *variáveis* $X_1,...,X_n$ *do modelo*
U_i = *nós pais*
$X_i^{\#}$ = *número de classes do nó i*
$U_i^{\#}$ = *número de classes do nó pai*

2.2.2.3 Fator de Bayes

Uma forma extremamente simples utilizada para comparar modelos de rede sem que seja necessária a utilização de uma constante de normalização, muitas vezes desconhecida, pode ser obtida através da utilização do fator de Bayes (BARBER, 2012), conforme mostra a Equação 11.

$$\frac{\boldsymbol{P}(\boldsymbol{G_i}|\boldsymbol{D})}{\boldsymbol{P}(\boldsymbol{G_j}|\boldsymbol{D})} = \frac{\boldsymbol{P}(\boldsymbol{D}|\boldsymbol{G_i})}{\boldsymbol{P}(\boldsymbol{D}|\boldsymbol{G_j})}\ \frac{\boldsymbol{P}(\boldsymbol{G_i})}{\boldsymbol{P}(\boldsymbol{G_j})} \qquad \textbf{(11)}$$

2.2.3 Algoritmos de Busca

> *A tarefa que um sistema simbólico enfrenta, quando a ele é apresentado um problema e o espaço do problema, é a de como utilizar seus limitados recursos de processamento para gerar possíveis soluções, uma após a outra, até encontrar uma que satisfaça o teste que define o problema. Se o sistema simbólico tiver algum controle sobre a ordem em que as potenciais soluções são geradas, então é desejável dispor esta ordem de modo que as soluções reais tenham uma maior probabilidade de surgirem antes que as demais. Um sistema simbólico exibiria inteligência na medida em que ele conseguisse fazer isso. Inteligência para um sistema com recursos limitados de processamento consiste em fazer escolhas sábias sobre o que será feito a seguir...* (NEWELL; SIMON, 1975).

A solução de problemas por meio de busca heurística (busca com uso de conhecimento) tem sido bastante utilizada e pesquisada desde a origem da Inteligência Artificial. Isso se deve ao fato de que em muitos casos não é possível examinar todos os estados possíveis e dessa forma encontrar a solução exata, seja devido à quantidade de recursos computacionais ou mesmo ao tempo necessário para a conclusão da tarefa. Problemas desse tipo são classificados como NP-Completos, ou seja, impossíveis de calcular no pior caso.

Os métodos de busca são avaliados (para que o método seja considerado mais útil do que outro) de acordo com algumas propriedades. Podemos citar como exemplo a complexidade que define o espaço e tempo necessários para a resolução do problema; a completude que define se o método consegue ou não atingir um estado objetivo; e quanto a ser ótimo que é aplicada aos métodos que conseguem encontrar a melhor solução dentro de um espaço de busca (COPPIN, 2010).

Na aprendizagem de redes Bayesianas (construção da estrutura da rede) vários algoritmos (métodos de busca) têm sido utilizados, como por exemplo, "busca gulosa", "subida da encosta", "têmpera simulada" e "EM[2] estrutural".

[2] Esperança-Maximização

A busca gulosa procura a melhor solução, o mais próximo possível à meta, acreditando que isso levará a uma solução rápida. O algoritmo de subida da encosta se move sempre em direção ao valor mais alto, isto é, encosta acima. Foram criadas muitas variantes para o algoritmo de subida da encosta. Na sua versão mais básica, o algoritmo de subida da encosta pode ficar parado em um máximo local (RUSSEL; NORVIG, 2004).

O algoritmo de têmpera simulada tem sido bastante utilizado para resolver problemas de otimização de grande escala, na qual um extremo global desejado está oculto entre vários extremos locais mais pobres (PRESS et al., 2011).

O algoritmo de têmpera simulada é também chamado de Simulação de Metrópole de Monte Carlo, sendo aplicado a problemas combinatoriais multivalorados, nos quais é necessário escolher valores para muitas variáveis, a fim de produzir um valor específico para alguma função global dependente de todas as variáveis do sistema. Esse valor é visto como a energia do sistema e, geralmente, o objetivo do algoritmo é encontrar a energia mínima para o sistema (COPPIN, 2010). O laço de repetição mais interno do algoritmo é muito semelhante à subida da encosta, porém, em vez de escolher o melhor movimento, ele escolhe um movimento aleatório. Se o movimento melhorar a situação, ele será aceito, caso contrário, o algoritmo aceitará o movimento com alguma probabilidade menor que 1. A probabilidade diminui exponencialmente com a "má qualidade" do movimento. A probabilidade também diminui à medida que a “temperatura” reduz, de forma que movimentos "ruins" têm maior probabilidade de serem aceitos no início (RUSSEL; NORVIG, 2004).

2.3 CADEIAS DE MARKOV MONTE CARLO

2.3.1 Método de simulação de Monte Carlo

O método de Monte Carlo consiste em simular um experimento com a finalidade de determinar propriedades probabilísticas (encontrando uma aproximação da medida de probabilidade através de tentativas) de uma população, a partir de uma amostragem aleatória dos componentes dessa população.

Segundo Evans e Olson (1998), a simulação de Monte Carlo é um processo de amostragem, cujo objetivo é permitir a observação do desempenho de uma variável de interesse, em razão do comportamento de variáveis que carregam elementos de incerteza. Embora seja um

conceito simples, a operacionalização desse processo requer o auxílio de métodos matemáticos, como a geração de números pseudoaleatórios, nos quais se destaca o método da transformada inversa.

O método de Monte Carlo requer os seguintes componentes:

- Função densidade de probabilidade (FDP).
- Gerador de números aleatórios (GNA).
- Gerador de variáveis aleatórias (GVA).

As amostras obtidas devem ser aleatórias. Para isso, é preciso obter uma sequência de números que atendam a essa restrição. Computacionalmente, esta sequência é facilmente obtida, utilizando um GNA. Os números gerados são na verdade pseudoaleatórios. Apesar disso, fornecem aproximações razoáveis de números aleatórios inteiros e podem ser utilizados para obter amostras de alguma população de interesse (FREITAS FILHO, 2008).

As variáveis do sistema que está sendo modelado devem ser descritas por uma Função Densidade de Probabilidade (FDP). As FDP's são modelos probabilísticos construídos a partir de uma análise estatística das amostras coletadas de forma a identificar qual a distribuição de probabilidade mais se aproxima da amostra. Uma FDP pode ser representada por uma função não negativa com a área, formada entre o eixo das abscissas e a curva da função, sendo igual a 1 (um). Os eventos analisados são representados por intervalos no eixo das abscissas, enquanto suas probabilidades de ocorrência correspondem às áreas sob a curva relativas a esses intervalos.

No método de Monte Carlo, uma vez definida a FDP, ela normalmente é acionada tendo como parâmetros de entrada valores gerados aleatoriamente (números aleatórios) e parâmetros específicos da distribuição, como, por exemplo, média, desvio padrão, mínimo e máximo.

O modelo Gaussiano, por exemplo, possui dois parâmetros que são a média (ex. $\boldsymbol{\mu=20}$) e o desvio padrão (ex. $\boldsymbol{\sigma=1}$). A partir da FDP pode-se obter a probabilidade de ocorrência de um valor dentro de uma dada faixa de valores do eixo x.

Depois de definidas as distribuições de probabilidade associadas às variáveis aleatórias, gera-se uma amostra simulada de valores destas distribuições. A geração da amostra se dá a partir da simulação de um número $\boldsymbol{n}$ de observações de cada uma das variáveis aleatórias que pertencem ao modelo criado (LUSTOSA; GARCIA; BARROS, 2010).

Após este processo pode-se verificar a aderência da amostra simulada com a amostra coletada a partir de testes não paramétricos (FREITAS FILHO, 2008).

2.3.2 Cadeias de Markov

Uma cadeia de Markov é um sistema dinâmico cujos vetores de estado, numa sucessão de intervalos de tempo, são vetores de probabilidade (Equação12).

$$\boldsymbol{P(X_{n+1} = j | X_0, X_1, X_2, \ldots, X_n) = P_{ij} = P(X_{n+1} = j | X_n = i)} \qquad \textbf{(12)}$$

onde

X = variável aleatória
1,2, ..., n = possíveis estados da cadeia
P_{ij} = probabilidade de transição (probabilidade de o sistema estar no estado i se na observação anterior estava no estado j)

A matriz $\boldsymbol{P=[P_{ij}]}$ é chamada de *matriz de transição* da cadeia de Markov (ANTON, 2006).

2.3.2.1 Ergodicidade

Uma cadeia de Markov definida pela matriz de transição deve ser ergódica, ou seja, todo estado deve ser acessível a partir de qualquer outro estado e não pode ter nenhum ciclo estritamente periódico (RUSSEL; NORVIG, 2004).

2.3.2.2 Distribuição estacionária

Diz-se que uma cadeia alcançou sua distribuição estacionária quando $\boldsymbol{\pi_{t=}\pi_{t+1}}$, definida pela Equação 13.

$$\boldsymbol{\pi(x') = \sum_x \pi(x) q(x \rightarrow x')} \qquad \text{para todo } x'. \qquad \textbf{(13)}$$

onde

x = estado anterior da cadeia
x' = estado atual da cadeia
$\pi(x')$=distribuição alvo
$q(x \rightarrow x')$ = distribuição de proposta

2.3.2.3 Reversibilidade

A probabilidade de estar em um estado pouco provável ***s*** mas procurando por um estado provável ***s'*** precisa ser a mesma de estar em um estado ***s'*** e procurando por um estado menos provável ***s***, conforme a Equação 14 (BROOKS et al., 2011).

$$\boldsymbol{\pi(x)q(x \rightarrow x') = \pi(x')q(x' \rightarrow x)} \qquad \textbf{(14)}$$

onde

x = estado anterior da cadeia
x' = estado atual da cadeia
π(x')=distribuição alvo
q(x⟶x') = distribuição de proposta

2.3.3 Monte Carlo e Cadeias de Markov

O método de Monte Carlo consiste em simular um experimento com a finalidade de determinar propriedades probabilísticas (encontrando uma aproximação da medida de probabilidade através de tentativas) de uma população, a partir de uma amostragem aleatória dos componentes dessa população.

O método de Monte Carlo e Cadeias de Markov (MCMC) é um método de amostragem aleatório. O objetivo do método é visitar um ponto $\boldsymbol{x}$ com uma probabilidade proporcional a alguma função de distribuição $\boldsymbol{\pi}(\mathbf{x})$ dada. A motivação para utilizar essa técnica é que métodos Bayesianos implementados utilizando MCMC representam uma maneira poderosa de se estimar os parâmetros de um modelo e seu grau de incerteza (PRESS et al., 2011).

2.3.3.1 Distribuição de proposta

A distribuição de proposta permite gerar amostras de interesse e define a probabilidade de o processo passar de um estado para outro. A única restrição para a escolha da distribuição de proposta é que a sucessão de passos por ela gerada possa alcançar qualquer ponto da distribuição de interesse (PRESS et al., 2011).

2.3.3.2 Distribuição alvo

A distribuição $\pi(x)$ não é bem uma distribuição de probabilidade, pois não é necessariamente normalizada para ter uma integral unitária na região de amostragem, porém é proporcional a uma probabilidade (PRESS et al., 2011). Uma questão central na simulação via cadeias de Markov é determinar as condições sob as quais existe uma distribuição estacionária $\pi(.)$. As iterações de um núcleo de transição convergem para a distribuição invariante na medida em que $n \rightarrow \infty$, sendo n o número de iterações da cadeia de Markov. Nos métodos MCMC, a distribuição estacionária $\pi(.)$ é conhecida (a menos de uma constante de normalização), e é a distribuição alvo, da qual se deseja amostrar. Para gerar amostras de $\pi(.)$, busca-se encontrar uma distribuição de proposta que convirja para $\pi(.)$. Após várias iterações da cadeia, a distribuição das amostras geradas por simulação é, aproximadamente, a distribuição alvo (BORGES, 2008).

2.3.3.3 Equilíbrio detalhado

A propriedade conhecida como equilíbrio detalhado (Equação 14) pode ser interpretada como o significado de que o "fluxo de saída" esperado a partir de cada estado é igual ao "fluxo de entrada" de todos os estados e é equivalente à propriedade de Reversibilidade em uma Cadeia de Markov.

2.3.3.4 Algoritmo de Metropolis-Hastings

A ideia principal do algoritmo de Metropolis-Hastings é a de obter uma distribuição de proposta (que satisfaça o equilíbrio detalhado), gerar uma amostra a partir dela e decidir se se mantém o processo no estado atual ou se aceita a amostra gerada.

2.3.3.4.1 Estado inicial

Devido à propriedade da ergodicidade, a distribuição de equilíbrio pode ser atingida a partir de qualquer estado inicial x_0 da cadeia. Mesmo valores muito improváveis de x_0 serão visitados pela cadeia de Markov em equilíbrio uma vez, depois de um grande lapso de tempo. É importante ressaltar que valores iniciais muito improváveis, de acordo com as propriedades da cadeia, também serão sucedidos por

valores improváveis, até que se atinja uma parte mais provável da distribuição de interesse (PRESS et al., 2011).

2.3.3.4.2 Probabilidade de aceitação

A partir de um estado inicial $\boldsymbol{x_1}$, gera-se um ponto candidato $\boldsymbol{x_{2c}}$ a partir da distribuição de proposta. Em seguida, calcula-se a probabilidade de aceitação $\boldsymbol{\alpha(x_1 \mid x_{2c})}$ a partir da Equação 16.

$$\boldsymbol{P(x_1, x_{2c}) = \min\left(1, \frac{\pi(x_{2c}) q(x_1|x_{2c})}{\pi(x_1) q(x_{2c}|x_1)}\right)} \quad \textbf{(16)}$$

A partir de $\boldsymbol{P(x_1 \mid x_{2c})}$, aceita-se o ponto candidato, fazendo $\boldsymbol{x_2 = x_{2c}}$, caso contrário, rejeita-se o ponto candidato e fazendo $\boldsymbol{x_2 = x_1}$. O resultado líquido desse processo é a probabilidade de transição. Essa probabilidade satisfaz o balanço detalhado (PRESS et al., 2011).

2.3.3.4.3 Burn-in

B*urn-in* é um termo coloquial que descreve a prática de descartar as iterações iniciais (durante o período de *burn-in*) da execução. O tamanho desta fase depende do valor inicial e da taxa de convergência da cadeia. A ideia principal da etapa de *burn-in* é fazer com que a cadeia "esqueça" o estado inicial (GILKS; RICHARDSON; SPIEGELHALTER, 1996).

2.3.3.4.4 Convergência

Uma das formas de monitorar a convergência é detectar o momento em que a cadeia de Markov "esqueceu" o seu estado inicial. Isso pode ser feito comparando o desempenho de várias execuções da cadeia, iniciadas em pontos diferentes, verificando o ponto em que elas se tornam semelhantes. A abordagem mais comum é desenhar em um único gráfico, as séries das várias execuções da cadeia e verificar se existem sequências distintas, demonstrando, nesse caso, que nem todas as séries se misturaram. Também se pode utilizar uma análise de variância, em uma abordagem mais quantitativa, para monitorar a convergência. Nesse caso, a aproximação da convergência ocorre quando a variância entre as diferentes sequências não são maiores do que a variância de cada sequência (GILKS; RICHARDSON; SPIEGELHALTER, 1996).

2.4 TESTES DE DEPENDÊNCIA

2.4.1 Teste de correlação de Pearson

Pode-se dizer que duas variáveis, ***X*** e ***Y*** estão positivamente correlacionadas quando elementos com valores pequenos de ***X*** tendem a ter valores pequenos de ***Y*** e elementos com valores grandes de ***X*** tendem a ter valores grandes de ***Y***. Da mesma forma, diz-se que estão negativamente correlacionados quando elementos com valores pequenos de ***X*** tendem a ter valores grandes de ***Y*** e elementos com valores grandes de ***X*** tendem a ter valores pequenos de ***Y*** (BARBETTA; REIS; BORNIA, 2004).

O coeficiente de correlação linear de Pearson descreve a correlação linear entre duas variáveis aleatórias e é definido pela Equação 17.

$$\boldsymbol{r} = \frac{\boldsymbol{n}(\sum_{i=1}^{n}(\boldsymbol{x_i} \,.\, \boldsymbol{y_i}) - (\sum \boldsymbol{x_i}) \,.\, (\sum \boldsymbol{y_i})}{\sqrt{\boldsymbol{n}(\sum \boldsymbol{x_i^2}) - (\sum \boldsymbol{x_i})^2} \;.\; \sqrt{\boldsymbol{n}(\sum \boldsymbol{y_i^2}) - (\sum \boldsymbol{y_i})^2}} \qquad \textbf{(17)}$$

onde

x_i' = *n-ésimo valor de x*
y_i' = *n-ésimo valor de y*
n = número de pares de dados presentes

2.4.2 Teste de Informação Mútua

A noção de independência é um caso especial de um conceito mais geral conhecido como informação mútua (DARWICHE, 2009), conforme a Equação 18.

$$\boldsymbol{MI(X;Y)} \stackrel{\text{def}}{=} \sum_{\boldsymbol{x,y}} \mathbf{P}(\boldsymbol{x,y}) \log_2 \frac{\mathbf{P}(\boldsymbol{x,y})}{\mathbf{P}(\boldsymbol{x})\,\mathbf{P}(\boldsymbol{y})} \qquad \textbf{(18)}$$

O resultado da função de informação mútua será não negativo e igual a zero somente se as variáveis ***X*** e ***Y*** forem independentes. Em termos mais gerais, a informação mútua mede a extensão em que a observação de uma variável irá reduzir a incerteza sobre a outra. Ou seja, mede a quantidade de informação que a variável ***X*** provê a respeito da variável ***Y*** e que pode ser obtida através dos valores da função de

entropia (***ENT(X)***) e entropia condicional (***ENT(X|y)***), conforme as Equação 19.

$$\boldsymbol{MI(X;Y) = ENT(X) - ENT(X|Y)} \qquad \textbf{(19)}$$

onde

$$\boldsymbol{ENT(X) = -\sum_x \mathbf{P}(x) \log_2 \mathbf{P}(x)} \qquad \textbf{(19.1)}$$

e

$$\boldsymbol{ENT(X|Y) = \sum_y \mathbf{P}(y)\, ENT(X|y)} \qquad \textbf{(19.2)}$$

2.4.3 Teste de associação qui-quadrado

O teste qui-quadrado ($\boldsymbol{\chi^2}$) é utilizado para verificar se existe associação entre duas variáveis qualitativas (categóricas), ***X*** e ***Y***, com base em uma amostra de observações disposta numa tabela de contingência com ***L*** linhas e ***C*** colunas (***L≥ 2, C ≥ 2),*** correspondentes às categorias de ***X*** e ***Y,*** respectivamente. A hipótese nula (***H_0)*** aceita independência entre categorias de ***X*** e ***Y,*** enquanto a hipótese alternativa (***H_1)*** afirma associação entre ***X*** e ***Y*** (BARBETTA; REIS; BORNIA, 2004). A distância $\boldsymbol{\chi^2}$ é uma medida da discrepância entre as frequências esperadas e observadas, sendo obtida pela Equação 20:

$$\boldsymbol{x^2 = \sum_{i=1}^{L} \sum_{j=1}^{C} \left[\frac{(O_{ij} - E_{ij})^2}{Eij} \right]} \qquad \textbf{(20)}$$

onde

O_{ij} = frequência observada na linha *i* e coluna *j;*

E_{ij} = frequência esperada na linha *i* e coluna *j,* supondo H_0 verdadeira.

Sob $\boldsymbol{H_0}$, a estatística $\boldsymbol{\chi^2}$ segue uma distribuição qui-quadrado com graus de liberdade igual a:

$$\boldsymbol{gl = (L-1)(C-1)} \qquad \textbf{(20.1)}$$

Pode-se dizer que duas variáveis são independentes se as diferenças entre as frequências observadas e esperadas nas categorias

forem muito pequenas (próximas a zero). Sendo assim, o teste qui-quadrado (χ^2) é utilizado para verificar a frequência com que um acontecimento observado se desvia ou não da frequência com que ele é esperado para não associação.

3 ESTADO DA ARTE

Atualmente existem muitos trabalhos destinados à construção automática de modelos de redes, todos com o objetivo de construir uma rede que melhor represente os dados. A seguir serão apresentados trabalhos recentes, bem como trabalhos que têm servido de referências para as pesquisas atuais.

No trabalho de Cano, Masegosa e Moral (2011) é proposto um método que utiliza o conhecimento do especialista e do sistema proposto para construir a rede. O algoritmo realiza o primeiro trabalho, sugerindo uma rede, e o especialista realiza os ajustes necessários principalmente no que diz respeito à dependência entre as variáveis da rede. O trabalho também faz uso de MCMC para aproximar a probabilidade posterior em função dos dados através do método de amostragem *Importance Sampling* (IS).

Liao, Qiu e Zeng (2011) partem do pressuposto de que os nós de uma rede Bayesiana têm características de imprecisão e aleatoriedade simultaneamente, sendo assim o trabalho propõe uma Rede Fuzzy-Bayesiana (RFB). O trabalho define as probabilidades fuzzy e uma tabela de probabilidade condicional fuzzy (TPCF) para expressar a relação entre as variáveis. Para otimizar a aprendizagem estrutura e parâmetros de aprendizagem é utilizado um algoritmo genético, onde são fixados os parâmetros de rede, e ao mesmo tempo, os parâmetros das funções de pertinência.

Em Guo (2009) foi proposto um método onde o algoritmo *Expectation Maximization* (EM) é utilizado para a aprendizagem de parâmetros e o método *Markov Chain Monte Carlo* (MCMC) é utilizado para gerar amostras de estruturas de rede. Diferente de outros, além das técnicas acima, neste trabalho também foi utilizado o método de amostragem MCMC *Accepting-Rejection* (AR) na função de seleção de amostras do algoritmo MCMC.

Em Zhang e Liu (2008) o algoritmo seleciona um conjunto de amostras otimizadas para ajuste *on-line* dos parâmetros de uma rede já existente. As amostras da estrutura da rede geradas pelo método de MCMC são avaliadas pelo método *importance sampling (*IS) antes da estrutura da rede ser atualizada. Ao final, o algoritmo decide se o modelo criado deve substituir o anterior através da comparação de modelos.

O trabalho de Jun e Li (2009) também utiliza atualização *on-line* da estrutura de uma rede já existente. Para seleção de amostras é utilizado MCMC *importance sampling (*IS). Baseado na *informação*

mútua[3] entre os nós da rede o algoritmo cria estruturas paralelas de Cadeias de Markov que convergem para a distribuição Boltzmann[4] (utilizada na busca Têmpera Simulada). A comparação dos modelos de rede é feita através da métrica BDe (*Bayesian Dirichlet likehood equivalence*)[5].

Riggelsen (2005) propõe a utilização da cobertura de Markov ao invés do tratamento detalhados dos nós, dividindo a rede em blocos (*Markov Blankets*) visando minimizar os tratamentos de dependência. A ideia central do trabalho reside no fato de que o método de MCMC trabalha melhor porque existe uma dependência natural entre os nós dentro de uma cobertura de Markov.

Janzura e Nielsen (2006) utilizam o algoritmo de Têmpera Simulada e MCMC. Gogate e Dechter (2011) utilizam MCMC *importance sampling (*IS) e propõem um método para minimizar a rejeição de amostras através de uma busca de amostras baseada em restrições.

Shi e Xing (2008) propõem uma variação no MCMC utilizando *informação mútua* para determinar a independência condicional entre duas variáveis. A rede obtida é considerada como estado inicial para uma Cadeia de Markov. Usando os operadores de rede (adicionar, excluir e inverter), obtêm-se novos valores para a Cadeia de Markov. Ao final da iteração a Cadeia de Markov resultante é utilizada para construir a nova estrutura da rede. O trabalho de Larranaga et al. (1996) é baseado na busca da melhor ordenação das variáveis do modelo utilizando algoritmos genéticos. A qualidade da ordenação é avaliada com o algoritmo K2. Masegosa e Moral (2013) primeiramente constróem um esqueleto inicial da estrutura da rede através de algoritmos de busca, como subida da encosta, antes de iniciar a busca por estruturas de rede através do método de MCMC. Para comparar os modelos de rede construídos através da aplicação do algoritmo MCMC, utilizam a métrica conhecida como "*Bayesian scoring measure"*.

O uso de testes estatísticos para identificar relações e construir a estrutura gráfica da rede tem sido utilizado com frequência (CHENG; GREINER, 1999). Friedman e Geiger (1997) utilizam um processo de seleção de variáveis (e descarte de outras), além de utilizarem a função de informação mútua para quantificar as relações de dependência entre as variáveis do modelo de dados. Zhang e Sheng (2004) propõem uma

[3] Para mais informações veja Preiss (2011) página 782.
[4] Para mais informações veja Preiss (2011) página 571.
[5] Veja Darwiche (2009) página 501.

nova explicação sobre o excelente desempenho do classificador ingênuo de Bayes, introduzindo o conceito de dependência local. No estudo de Lee, Gutierrez e Dou (2011) são atribuídos pesos às variáveis do conjunto de dados utilizando a medida de Kullback-Leibler.

O trabalho de Naeem e Asghar (2013) introduz um novo modelo de cálculo da métrica de informação mútua (*integration to segregation*), utilizando a propriedade marginal e a propriedade conjunta de pares de variáveis do conjunto de dados para definir a ordem dos nós na construção da rede. Um dos principais ganhos obtidos com o uso desta métrica, segundo o autor, é o de evitar problemas de *overfitting*, causados pelo excesso de conexões entre os nós da rede.

Salama e Freitas (2013) utilizam em seu trabalho o algoritmo de otimização de colônia de formigas para descobrir a estrutura da rede. Niinimäki e Koivisto (2013) propõem a utilização de amostragem de ordem dos nós da rede ao invés de amostrar a estrutura da rede em si, mas ao contrário de outros trabalhos, não utilizam o método MCMC para gerar as amostras e sim o método amostragem *annealed importance* (NEAL, 2001).

Como pode ser visto, a maior parte dos trabalhos utiliza algum algoritmo de busca para percorrer o conjunto de estruturas de redes de forma aleatória e alguma métrica associada para avaliar as estruturas construídas. Uma vez que o espaço de estruturas de rede é grande demais, dependendo da quantidade de variáveis do modelo, o caminho das pesquisas tem sido no sentido de descobrir formas eficientes de restringir o espaço de busca.

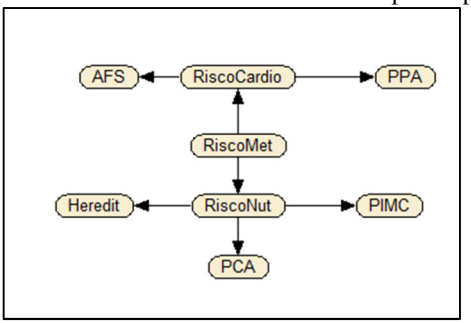
4 MATERIAIS E MÉTODOS

Este capítulo descreve os dois modelos de dados utilizados nos testes do algoritmo construído e também todos os componentes do algoritmo.

4.1 CONJUNTOS DE DADOS

Nesta pesquisa foram utilizadas dois modelos de redes, o primeiro dos conjuntos de dados utilizado nesta pesquisa foi coletado em pacientes atendidos no ambulatório de Nutrologia do Hospital Universitário da Universidade Federal de Santa Catarina – HU/UFSC/Brasil, no período de novembro de 2010 a novembro de 2011. As variáveis selecionadas para a criação das redes são referentes a dados antropométricos de atividade física, pressão arterial e avaliação do estado nutricional dos pacientes.

Figura 4 – Modelo da rede Pacientes HU criado pelo especialista.



Fizeram parte da amostra 120 crianças e adolescentes com idades entre 5 e 17 anos. A coleta de dados atendeu às diretrizes para pesquisa envolvendo seres humanos, estabelecida pela resolução nº 196/96 do Conselho Nacional de Saúde (Brasil) (MAYER, 2012). A Tabela 1 apresenta a lista das variáveis utilizadas nesta rede.

Todas essas variáveis são mensuradas em nível qualitativo ordinal, isto é, suas classes apresentam uma relação de ordem (*rank*) entre elas. Além dos dados coletados, também foram criados (através da aplicação do método de amostragem MCMC) mais três conjuntos de dados com 100, 500 e 1000 casos. Os casos foram criados utilizando as probabilidades fornecidas pelo especialista do domínio de acordo com o modelo apresentado na Figura 4. Os dados coletados foram reservados

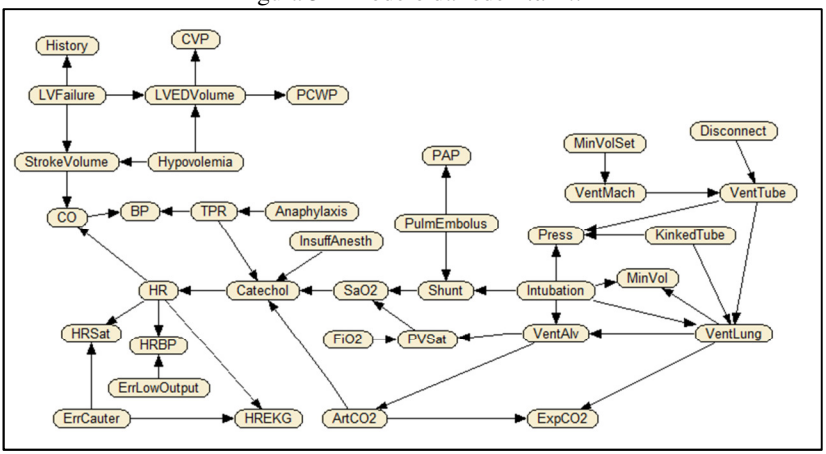
para os testes das redes. Este conjunto de dados será referenciado deste ponto em diante de conjunto de dados de *pacientes HU*.

Tabela 1 - Lista de variáveis utilizadas no modelo da rede pacientes HU.

Variável	N° de Classes
Pressão Arterial (PPA)	3
Circunferência abdominal (PCA)	2
Atividade física semanal (AFS)	3
Classificação antropométrica (PIMC)	5
Classificação antropométrica pais (H)	3
Risco cardíaco (RCV)	3
Risco nutricional (RN)	3
Risco metabólico (RM)	3

Na utilização do teste χ^2 é recomendável que o valor mínimo de frequência esperada seja maior ou igual a cinco (FREITAS FILHO, 2008). Este critério não é atendido no conjunto de dados utilizado, nas variáveis de *Atividade Física Semanal (AFS), Pressão Arterial (PPA)* e *Classificação Antropométrica Pais (H)* na rede ingênua e nas variáveis *Atividade Física Semanal (AFS)* e *Pressão Arterial (PPA)* na rede hierárquica. Apesar disso, neste trabalho, esta recomendação foi desconsiderada.

Figura 5 – Modelo da rede *Alarm*.



O segundo modelo de rede utilizado foi a rede *Alarm* (Figura 5). Originalmente descrita por Beinlich et al. (1989) como uma rede de

monitoramento de pacientes em terapia intensiva. Esta rede é composta por 37 nós de duas, três ou quatro classes e 46 arcos (Anexo A). As probabilidades utilizadas para a geração dos exemplos foram retiradas do site Norsys (2012). Foram geradas amostras de 100, 500 e 1000 casos para aprendizagem, além de mais 500 casos para testes.

4.2 ALGORITMO PROPOSTO

Este capítulo apresenta o método proposto para a aprendizagem de estruturas de redes Bayesianas através da aplicação do algoritmo de Markov Chain Monte Carlo.

4.2.1 Geração de amostras

A função de geração de amostras implementada no algoritmo gera inicialmente três valores: a operação a ser realizada na estrutura da rede e dois valores indicando os dois nós nos quais será aplicada a operação. Esses valores serão utilizados para alterar a estrutura atual da rede, que, no caso de o algoritmo estar no ponto inicial, pode ser uma estrutura totalmente desconectada ou um modelo ingênuo de rede, ou outra estrutura qualquer, no caso de a estrutura inicial já ter sido modificada durante as iterações do algoritmo.

4.2.2 Operações

As operações possíveis geradas pela função de amostragem são inclusão, remoção ou inversão de um arco na estrutura atual da rede. A geração do valor pseudorrandômico, que definirá qual operação será realizada, é feita a partir de uma distribuição uniforme. Após a geração da operação e dos nós envolvidos, o algoritmo verifica se devem ser considerados os resultados dos testes estatísticos, de acordo com os parâmetros fornecidos na execução do programa. Os testes estatísticos têm a função de impedir que um arco entre dois nós, que possua uma correlação forte indicada por eles, possa ser removido. Ou seja, uma vez adicionado, este arco não pode mais ser removido da estrutura da rede, limitando assim a quantidade de configurações possíveis dos modelos de rede. Além disso, esse procedimento evita todo o processamento de recálculo de parâmetros que se constitui em um dos trechos do algoritmo que mais consume recursos computacionais.

4.2.3 Verificação de ciclos

Como uma rede Bayesiana não pode conter ciclos, é preciso a cada alteração na estrutura da rede verificar se ela contém ciclos ou não, e caso possua, esta amostra deve ser desprezada e uma nova amostra deve ser gerada. Essa verificação é feita utilizando o algoritmo de busca em profundidade (CORMEN et al., 2002).

4.2.4 Grafos Conectados

Outra restrição utilizada no algoritmo proposto é a de não aceitar amostras de estruturas de rede que possuam algum nó sem nenhuma conexão, ou seja, totalmente desconectado de todos os outros nós da rede. Esta verificação também é feita através de uma busca em profundidade.

4.2.5 Probabilidade de Aceitação

Após a geração de uma amostra válida, o algoritmo precisa decidir se aceita ou não esta amostra de acordo com probabilidade de aceitação (Equação 16). Todos os procedimentos descritos no processo de geração de amostras configuram a distribuição de proposta utilizada no algoritmo.

4.2.5.1 Estimação de parâmetros

Sempre que a estrutura da rede é alterada, é preciso recalcular as probabilidades condicionais. Este trecho do algoritmo considera as frequências relativas dos estados das variáveis do modelo.

4.2.5.2 Cálculo do *score*

Após o ajuste nas probabilidades, é possível calcular o *score* da rede gerada para que seja possível comparar o modelo antigo com o novo. Neste trabalho foi utilizada a métrica de *score* conhecida como BIC (*Bayesian Information Criteria*) (Equação 8) para realizar este cálculo. Esta métrica tenta identificar o modelo da rede que melhor representa os dados, ou seja, valoriza a estrutura de rede que tenha a maior probabilidade posterior, considerando os dados. Além disso, penaliza a complexidade da rede (redes com muitos arcos entre os nós), valorizando os modelos mais simples.

4.2.5.3 Fator de Bayes

Como não se dispõe de uma constante de normalização para conhecer a probabilidade de um modelo de rede específico, neste trabalho optou-se por utilizar um cálculo do fator de Bayes (Equação 11). Considerando a Equação 16, e de acordo com a probabilidade calculada, aceita-se o ponto candidato, substituindo o modelo atual da rede pela amostra, caso contrário, rejeita-se o ponto candidato sem alterar o modelo atual da rede. O resultado desse processo se constitui na probabilidade de transição que, por sua vez, satisfaz a propriedade do balanço detalhado.

4.3 IMPLEMENTAÇÃO DO ALGORITMO

Este item descreve a ferramenta construída para viabilizar a realização dos testes do algoritmo proposto. Foram criadas estruturas de dados para armazenar e manipular os dados, estruturas de redes e probabilidades.

No fluxo do algoritmo inicialmente são geradas amostras aleatórias de estruturas de redes Bayesianas (Quadro 1), partindo de modelos totalmente desconectados ou de modelos ingênuos de redes Bayesianas. Ou seja, a estrutura inicial da rede, passada como parâmetro para o algoritmo no início da execução do programa, não possui nenhuma conexão, ou então todos os nós estão conectados a um nó de saída escolhido previamente. Essa forma de trabalho tem a única intenção de fornecer um ponto inicial diferente para o algoritmo, já que, de acordo com o funcionamento dos algoritmos de MCMC, a rapidez com que a cadeia irá convergir depende também do seu estado inicial.

O algoritmo de geração de amostras consiste basicamente em gerar valores aleatórios, através de uma distribuição uniforme para a operação (adição, remoção ou inversão) que será realizada na estrutura da rede (linha 9), assim como os dois nós envolvidos (linhas 7 e 8), ou seja, os nós que serão conectados ou desconectados de acordo com a operação. Nas linhas 13 a 15 do algoritmo, caso a operação gerada seja de remoção e de acordo com o teste estatístico que está sendo utilizado (qui-quadrado, informação mútua ou Pearson), o algoritmo verificará se irá permitir ou não a realização da operação na estrutura da rede. Isso acontece porque, conforme descrito anteriormente, os nós que foram identificados como dependentes pelo teste estatístico aplicado são fixados no início da execução do algoritmo e não podem mais ser removidos.

Quadro 1 – Algoritmo para geração de amostras de rede.

```
Algoritmo genSample
1. Input:   redeIn  -  Rede Bayesiana
2. Output: redeOut  -  Rede Bayesiana
3.
4. while ( true )
5.       redeOut = redeIn
6.
7.       arcX  ~  uniform(0,MAX_NODE)
8.       arcY  ~  uniform(0, MAX_NODE)
9.       oper  ~  uniform(0,MAX_OPER)
10.
11.      // se operação é DEL e o arco foi fixado por algum
12.      // dos testes estatísticos, gera novos valores
13.      if oper = DEL and nodeFixed(arcX, arcY)
14.         continue;
15.      endif
16.
17.      altRede(redeOut, oper, arcX, arcY)
18.
19.      if hasCycle(redeOut)
20.         continue;
21.      endif
22.
23.      if  notConnected(redeOut)
24.         continue;
25.      endif
26.
27. endwhile
```

Seguindo o fluxo da execução do algoritmo de geração de amostras, o próximo passo, uma vez validadas as condições necessárias, consiste na alteração da estrutura da rede original (linha 17). Depois de alterada a estrutura da rede, é preciso verificar se a nova rede possui ciclos (linhas 19 a 21), o que não é permitido, uma vez que redes Bayesianas são gráficos acíclicos dirigidos. A outra validação realizada na estrutura da rede é verificar se existe algum nó desconectado na rede, condição esta que foi considerada como uma restrição. O motivo para essa escolha está relacionado com a tarefa de comparação dos scores das redes, uma vez que o tamanho (quantidade de nós de cada modelo de rede) é diferente. Os modelos de rede gerados pelo algoritmo que atendessem aos critérios descritos anteriormente foram aceitos pelo algoritmo.

De posse de um novo modelo de rede, faz-se necessário recalcular os parâmetros (ou probabilidades) do modelo, o que no algoritmo que pode ser visto no Quadro 2, acontece na linha 7. Após o cálculo das novas probabilidades do modelo de rede, é necessário calcular o novo valor de score para esse modelo (linha 9). Neste trabalho foi adotado o score BIC (Equação 8). De posse do valor do score, resta comparar a rede anterior com a rede atual para decidir qual dos dois modelos é o que melhor representa os dados.

Quadro 2 – Algoritmo Metrópolis-Hastings.

Algoritmo MCMC

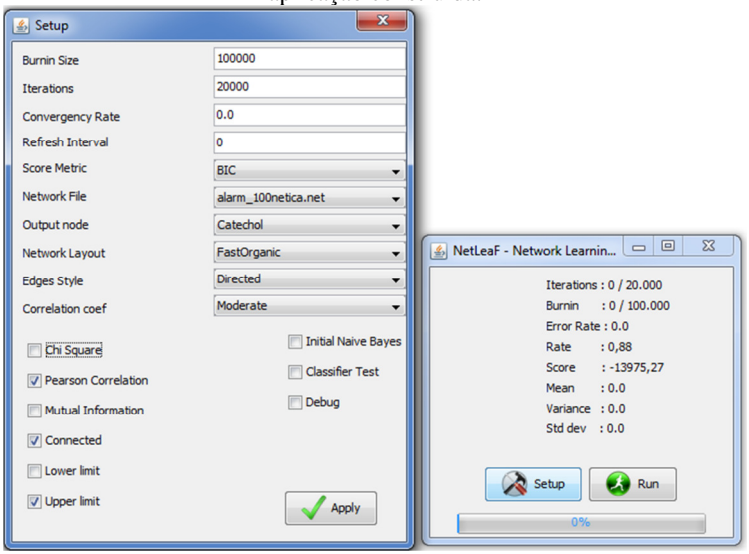
```
1. Input:   rede  -  Rede Bayesiana
2.          casos – casos com valores para as variáveis
3. Output: dag – gráfico acíclico dirigido

4. loop = 1
5. while ( loop < iteracoes )
6.       sampleDAG = genSample(rede)
7.       rede = LearnParameter(rede,casos)
8.       // Cálculo – Equação 8
9.       novoScore = computing BICScore(rede,casos)
10.       // Cálculo – Equação 11
11.      fatorBayes = novoScore / oldSCORE
12.       u  ~  uniform(0,1)
13.       if  u < min(1, fatorBayes )
14.          oldSCORE = novoSCORE
15.          rede = sampleDAG
16.       endif
17. endwhile
```

O algoritmo presente no Quadro 2 demonstra a ideia central do algoritmo de MCMC. Relembrando que essa ideia consiste em escolher um movimento aleatório (linha 6) e caso esse movimento melhore a situação, ele sempre será aceito (a função de distribuição uniforme utilizada gera apenas valores menores que 1). Caso contrário, o algoritmo aceita o movimento com uma probabilidade menor que 1 (linhas 13 a 16). Quanto menor o valor retornado no cálculo do *score,* menor a probabilidade de aceitação. Como o algoritmo "caminha" naturalmente em busca de modelos de rede que possuam um *score* maior, quanto maior a quantidade de movimentos, menor será a probabilidade de aceite de movimentos ruins (*scores* baixos). Isso ocorre porque o valor comparado com a probabilidade de aceite, na verdade

não é bem uma probabilidade, pois não está normalizada (linha 11), é obtido a partir de uma distribuição uniforme (linha 12).

Na Figura 6 são apresentadas a tela inicial de controle de execução do algoritmo e a tela de configuração, onde são selecionados os tipos de teste e restrições para cada execução. Também foram implementados métodos para cálculo de *score*, cálculo das estatísticas qui-quadrado e correlação de Pearson, o algoritmo de Metrópolis-Hasting e o algoritmo de busca em profundidade. O cálculo da métrica informação mútua foi realizado através do pacote criado por Pocock (2013) e os métodos para consultas nas redes geradas foram viabilizados através da utilização da biblioteca Netica-J (NORSYS, 2012).

Figura 6 – Tela inicial de controle de execução e tela de configuração da aplicação construída.



4.3.1 Linguagem de programação

Para o desenvolvimento do programa desenvolvido neste trabalho, foi utilizada a linguagem de programação Java no ambiente de desenvolvimento Eclipse. O código foi desenvolvido utilizando o método de desenvolvimento de sistemas orientados a objetos.

4.3.2 Estruturas de dados

Para representar as estruturas de redes ou grafos foi utilizado o conceito de matriz de adjacência (CORMEN, 2002) em função da praticidade para realizar as operações sobre os arcos do modelo de rede (adicionar, remover ou inverter). Na representação de matriz de adjacências de um grafo $G = (V, E)$, os vértices são numerados de forma arbitrária (V representa o conjunto de nós da rede e E representa o conjunto de arcos). Assim, a representação de matriz de adjacências de um grafo G consiste em uma matriz $|V| \times |V|$ $A = (a_{ij})$ tal que

$$a_{ij} = \begin{cases} 1 \text{ se } (i,j) \in E \\ 0 \text{ em caso contrário} \end{cases} \quad (21)$$

Este tipo de representação de grafos exige memória $\Theta(V^2)$, independente do número de arcos no grafo.

4.3.3 Ferramentas de análise dos testes

Foram realizadas várias sequências de testes com cada um dos dois conjuntos de dados (*Alarm* e pacientes HU). A configuração dos testes foi executada várias vezes em sequências de quatro execuções independentes para cada base de dados. Por exemplo, para cada uma das duas bases de dados foram realizados conjuntos de quatro execuções independentes em cada um dos três conjuntos de dados (100, 500 e 1.000 casos). A quantidade de ciclos nas fases de *Burn-in* e simulação foi configurada de acordo com a quantidade de dados e de variáveis do conjunto de dados: quanto maior o tamanho de dados e a quantidade de variáveis, maior a quantidade de iterações do algoritmo.

Em cada uma das execuções, os valores dos *scores* obtidos em cada iteração foram coletados e utilizados para gerar gráficos, e para que fosse possível através da comparação dos dados de cada conjunto das quatro execuções, visualizar a convergência entre as execuções do algoritmo. Além da análise visual, considerada como análise informal da convergência (GAMERMAN; LOPES, 2006), existem métodos considerados como formais para a realização dessa tarefa. A verificação formal da convergência é baseada em propriedades estatísticas das cadeias simuladas. Dentre estes métodos está o método de Gelman e Rubin (1992) que é baseado na análise de duas ou mais cadeias paralelas, inicializadas em diferentes pontos (GALVANIN, 2007). Este

método, disponível no pacote "boa" (SMITH, 2007) para o software estatístico "R" (2013) foi utilizado neste trabalho para realizar a análise formal da convergência. Ele se baseia em uma comparação, para cada uma das variáveis, entre a variância amostral dentro das cadeias e entre as cadeias. Esta comparação é utilizada para se estimar o fator para o qual o parâmetro de escala da distribuição marginal a posteriori ($\hat{R}$), pode ser reduzido à medida que o tamanho da amostra cresça. Gelman e Rubin (1992) sugerem aceitar como garantia de convergência valores de $\hat{R} \leq 1{,}2$.

Ao final de cada execução, foram salvas as sete (7) melhores estruturas de redes (redes com os maiores *scores*) e nesta rede foram aplicados os conjuntos de dados de teste. A partir desses testes, utilizou-se o método conhecido como matriz de confusão (ou matriz de classificação) para apresentar o resumo dos resultados. A ideia do método é bastante simples e consiste basicamente em uma matriz quadrada que contém todas as classes possíveis, tanto nas linhas quanto nas colunas. As colunas da matriz recebem os valores de resposta gerados pelas redes e as linhas recebem os valores da classe de saída de acordo com o padrão-ouro (MARSLAND, 2009).

4.3.4 Recursos computacionais

Todos os testes do algoritmo foram realizados em um computador com processador Intel Core i3, equipado com 4 gigabytes de memória e um disco rígido de 500 gigabytes. O sistema operacional utilizado foi o Microsoft Windows, versão 7.

5 RESULTADOS

Os resultados desta pesquisa envolvem a análise dos gráficos de convergência e dos testes das melhores redes geradas pelo algoritmo. A análise dos gráficos de convergência permite verificar as situações nas quais as cadeias demonstram convergência no menor número de iterações. Nesse caso são feitas comparações entre as execuções do algoritmo com e sem o uso dos testes de associação, correlação e informação mútua. Da mesma forma, as submissões dos conjuntos de teste aos melhores modelos gerados pelas redes também são comparadas com e sem o uso dos testes estatísticos.

Tabela 2 - Parâmetros das execuções do algoritmo.

Qui-quadrado	Pearson	Informação Mútua	Qtde. execuções	Rede Ingênua	Rede sem arcos
Não	Não	Não	2	Não	Sim
Não	Não	Não	2	Sim	Não
Sim	Não	Não	2	Não	Sim
Não	Sim	Não	2	Não	Sim
Não	Não	Sim	2	Não	Sim
Sim	Sim	Não	2	Não	Sim
Sim	Não	Sim	2	Não	Sim
Sim	Não	Não	2	Sim	Não
Não	Sim	Não	2	Sim	Não
Não	Não	Sim	2	Sim	Não
Sim	Sim	Não	2	Sim	Não
Sim	Não	Sim	2	Sim	Não

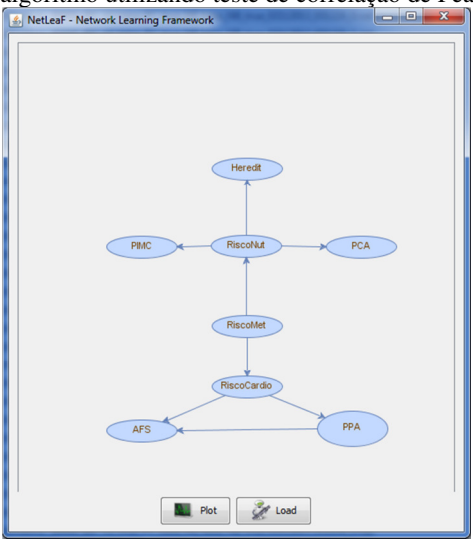
Para cada conjunto de dados, foram coletados dados de vinte quatro (24) execuções, duas (2) para cada configuração diferente, conforme mostra a Tabela 2. Realizaram-se execuções do algoritmo sem aplicação dos testes estatísticos, com aplicação de apenas um teste e do teste qui-quadrado combinado, ou com o teste de correlação de Pearson (qui-quadrado e Pearson), ou com o teste de Informação Mútua (qui-quadrado e Informação Mútua). No caso do coeficiente de correlação de Pearson, e da rede *Alarm,* o teste foi aplicado apenas nos nós com classes *Low, Normal e High.* Em cada uma dessas configurações, utilizaram-se, como redes iniciais, redes Bayesianas Ingênuas ou redes sem nenhum arco.

Tabela 3 - Quantidade de iterações em cada execução do algoritmo.

Conjunto de dados	Qtde. iterações
Alarm 100	120.000
Alarm 500	250.000
Alarm 1000	350.000
Pacientes HU 100	12.000
Pacientes HU 500	30.000
Pacientes HU 1000	60.000

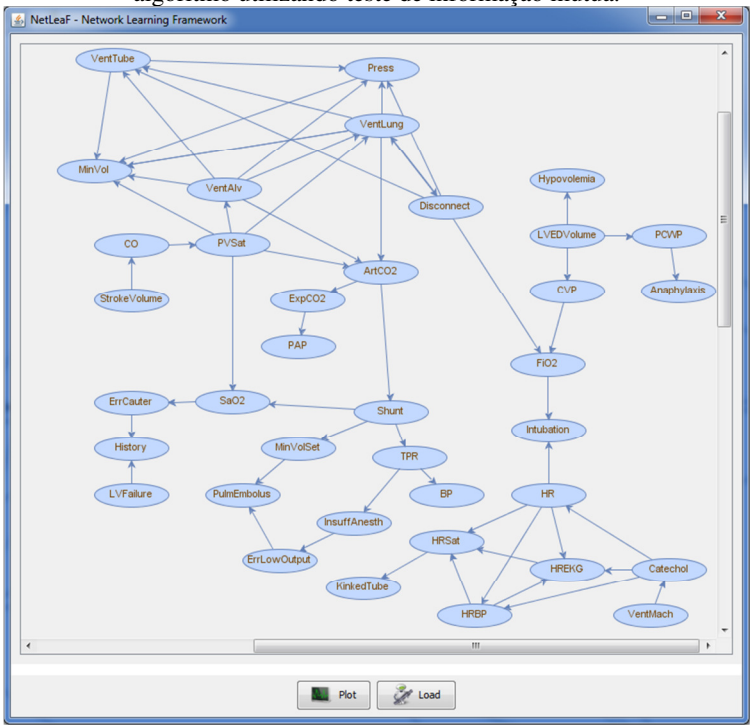
Em cada uma dessas execuções, a quantidade de iterações do algoritmo foi ajustada de acordo com a quantidade de casos do conjunto de dados (Tabela 3).

Figura 7 – Modelo de rede relativo ao conjunto de dados Pacientes HU gerado pelo algoritmo utilizando teste de correlação de Pearson.



A Figura 7 mostra um modelo de rede, para o conjunto de dados Pacientes HU, gerado pelo algoritmo. Para o conjunto de dados *Alarm*, o modelo gerado pelo algoritmo é apresentado na Figura 8.

Figura 8 – Modelo de rede relativo ao conjunto de dados *Alarm* gerado pelo algoritmo utilizando teste de informação mútua.



5.1 RESULTADOS DOS TESTES DAS REDES

Nesta seção são apresentados os resultados obtidos nos testes das redes gerados pelo algoritmo. No caso da rede Pacientes HU, os casos reais do conjunto de dados coletados foram utilizados para realizar os testes. O nó Risco Metabólico foi escolhido como nó de saída. A Tabela 4 mostra um resumo dos melhores resultados dos testes na rede Pacientes HU para cada tipo de teste estatístico utilizado no algoritmo.

Tabela 4 - Resultados dos testes das redes para o conjunto de dados Pacientes HU.

Método	**Taxa Erros**
Sem uso testes	37,50%
Qui-Quadrado	26,67%
Pearson	24,17%
Informação Mútua	26,67%

As Tabelas 5, 6, 7 e 8 apresentam as matrizes de classificação para os testes das redes. Cada uma das tabelas exibe os resultados da rede gerada de acordo com o método utilizado na execução do algoritmo.

Tabela 5 – Matriz de classificação para o resultado do teste da rede Pacientes HU gerada sem o uso de testes estatísticos.

Padrão-ouro	**Rede Pacientes HU**		
	Baixo	Moderado	Grave
Baixo	31	8	4
Moderado	14	20	4
Grave	2	13	24
Taxa de erro	**37,50%**		

Tabela 6 – Matriz de classificação para o resultado do teste da rede Pacientes HU gerada com o uso do teste qui-quadrado.

Padrão-ouro	**Rede Pacientes HU**		
	Baixo	Moderado	Grave
Baixo	34	8	1
Moderado	14	21	3
Grave	0	6	33
Taxa de erro	**26,67%**		

Tabela 7 – Matriz de classificação para o resultado do teste da rede Pacientes HU gerada com o uso do teste de correlação de Pearson.

Padrão-ouro	**Rede Pacientes HU**		
	Baixo	Moderado	Grave
Baixo	37	5	1
Moderado	12	25	1
Grave	1	9	29
Taxa de erro	**24,17%**		

Tabela 8 – Matriz de classificação para o resultado do teste da rede Pacientes HU gerada com o uso do teste de Informação Mútua.

Padrão-ouro	**Rede Pacientes HU**		
	Baixo	Moderado	Grave
Baixo	38	4	1
Moderado	16	19	3
Grave	2	6	31
Taxa de erro	**26,67%**		

Para os testes da rede *Alarm, o* nó *Intubation* foi selecionado como nó de saída, já que na estrutura original desta rede (BEINLICH et al., 1989) esse nó não possui nós pais. Os dados utilizados nos testes foram gerados com a ferramenta Netica. A Tabela 8 apresenta um resumo com os melhores resultados dos testes da rede *Alarm* de acordo com a configuração utilizada durante a execução do algoritmo.

Tabela 9 - Resultados dos testes das redes para o conjunto de dados *Alarm.*

Método	**Taxa Erros**
Sem uso testes	9,80%
Qui-Quadrado	9,80%
Pearson	8,60%
Informação Mútua	7,00%

As Tabelas 10, 11, 12 e 13 apresentam as matrizes de classificação para os testes das redes. Cada uma das tabelas exibe os resultados da rede gerada de acordo com o método utilizado na execução do algoritmo.

Tabela 10 – Matriz de classificação para o resultado do teste da rede *Alarm* gerada sem o uso de testes estatísticos.

Padrão-ouro	**Rede *Alarm***		
	Normal	*Esophageal*	*OneSided*
Normal	451	0	14
Esophageal	14	0	0
OneSided	21	0	0
Taxa de erro	**9,80%**		

Tabela 11 – Matriz de classificação para o resultado do teste da rede *Alarm* gerada com o uso do teste qui-quadrado.

Padrão-ouro	**Rede *Alarm***		
	Normal	*Esophageal*	*OneSided*
Normal	451	0	14
Esophageal	14	0	0
OneSided	21	0	0
Taxa de erro	**9,80%**		

Tabela 12 – Matriz de classificação para o resultado do teste da rede *Alarm* gerada com o uso do teste de correlação de Pearson.

Padrão-ouro	**Rede *Alarm***		
	Normal	*Esophageal*	*OneSided*
Normal	454	0	11
Esophageal	12	0	2
OneSided	18	0	3
Taxa de erro	**8,60%**		

Tabela 13 – Matriz de classificação para o resultado do teste da rede *Alarm* gerada com o uso do teste de Informação Mútua.

Padrão-ouro	**Rede *Alarm***		
	Normal	*Esophageal*	*OneSided*
Normal	465	0	0
Esophageal	14	0	0
OneSided	21	0	0
Taxa de erro	**7,00%**		

Com relação ao uso de recursos computacionais, na Tabela 14 são apresentados os tempos médios da execução de cada tipo de teste realizado. Pode-se ver na tabela que o tempo aumenta conforme aumenta a quantidade de iterações e também de acordo com a quantidade de casos no conjunto de testes. Isso acontece principalmente pelo tempo gasto na função de cálculo do *score.*

Tabela 14 – Tempo médio da execução dos conjuntos de teste.

Conjunto de teste	**Quantidade de Iterações**	**Quantidade de casos**	**Tempo (segundos)**
Pacientes HU	12.000	100	5
Pacientes HU	30.000	500	39
Pacientes HU	60.000	1000	126
Alarm	120.000	100	348
Alarm	250.000	500	2298
Alarm	350.000	1000	6180

Os resultados apresentados nesta seção foram obtidos através da submissão dos conjuntos de teste às melhores redes geradas pelo algoritmo. Para isso, as sete melhores redes, ou seja, as redes com os *scores* mais altos foram salvas ao final de cada execução do algoritmo e utilizadas nesses testes. Foram realizados testes em todas as redes selecionadas pela ferramenta e foi registrado apenas o melhor resultado

de cada conjunto de redes em cada configuração diferente, de acordo com o método utilizado na geração do modelo da rede.

5.2 GRÁFICOS DE CONVERGÊNCIA

Durante as execuções do algoritmo coletaram-se dados para gerar os gráficos de convergência. A convergência, neste caso, é representada pelo momento em que os valores de *score* obtidos a cada iteração se aproximam e se mantêm próximos, mesmo tendo partido de pontos diferentes. Dentre todos os gráficos gerados, serão comentados aqueles que apresentaram comportamentos relevantes.

Nas Figuras 9 e 10 são apresentados os gráficos da rede Pacientes HU onde foi utilizado o teste de correlação de Pearson no conjunto de dados com 100 casos. Na Figura 9 são apresentadas as duas execuções do algoritmo que utilizaram como ponto de partida uma rede totalmente desconectada. Na Figura 10 são apresentadas as duas execuções do algoritmo que utilizaram como ponto de partida uma rede ingênua, ou seja, todos os nós conectados ao nó de saída, no caso desta rede, foi escolhido o nó Risco Metabólico. Examinando os gráficos, é possível verificar que a cadeia converge por volta da iteração 1000 em ambos os gráficos.

A Figura 11 apresenta o gráfico da rede Pacientes HU, no qual foi utilizado o teste de correlação de Pearson no conjunto de dados com 500 casos. As duas execuções do algoritmo deste gráfico utilizaram como ponto de partida uma rede ingênua. Examinando o gráfico, percebe-se que a cadeia converge por volta da iteração 2500.

A Figura 12 apresenta o gráfico da rede Pacientes HU, no qual não foram utilizados testes estatísticos para o conjunto de dados com 500 casos. As duas execuções do algoritmo neste gráfico utilizaram como ponto de partida uma rede vazia. Na Figura 13, estão os dados relativos à execução dos testes para esta mesma rede mas desta vez com 1000 casos.

Neste gráfico é possível perceber uma grande variação nos valores do *score* de cada iteração, demonstrando uma dificuldade da cadeia em convergir.

Figura 9 – Gráfico de convergência - Rede Pacientes HU com teste de correlação de Pearson (rede vazia como rede inicial) com conjuntos de dados de 100 casos.

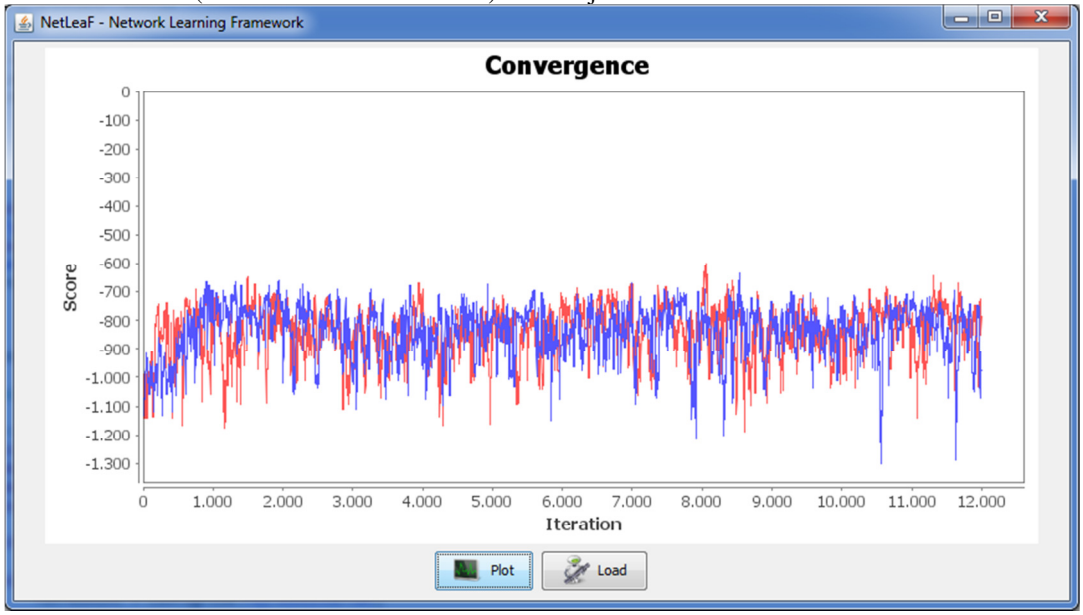


Figura 10 – Gráfico de convergência - Rede Pacientes HU com teste de correlação de Pearson (rede ingênua como rede inicial) com conjuntos de dados de 100 casos.

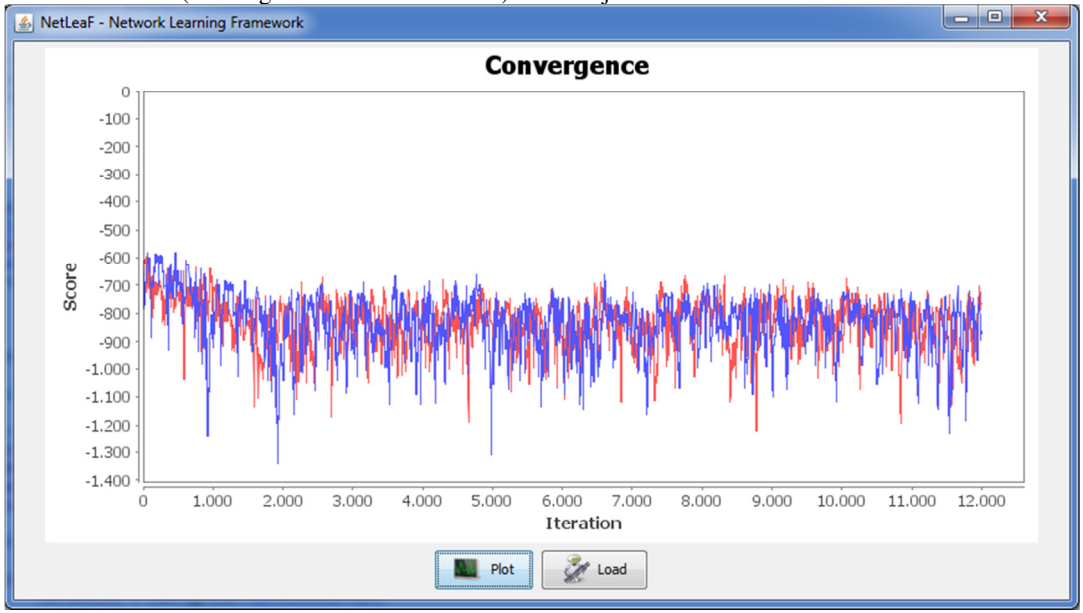


Figura 11 – Gráfico de convergência - Rede Pacientes HU com teste de correlação de Pearson (rede ingênua como rede inicial) com conjuntos de dados de 500 casos.

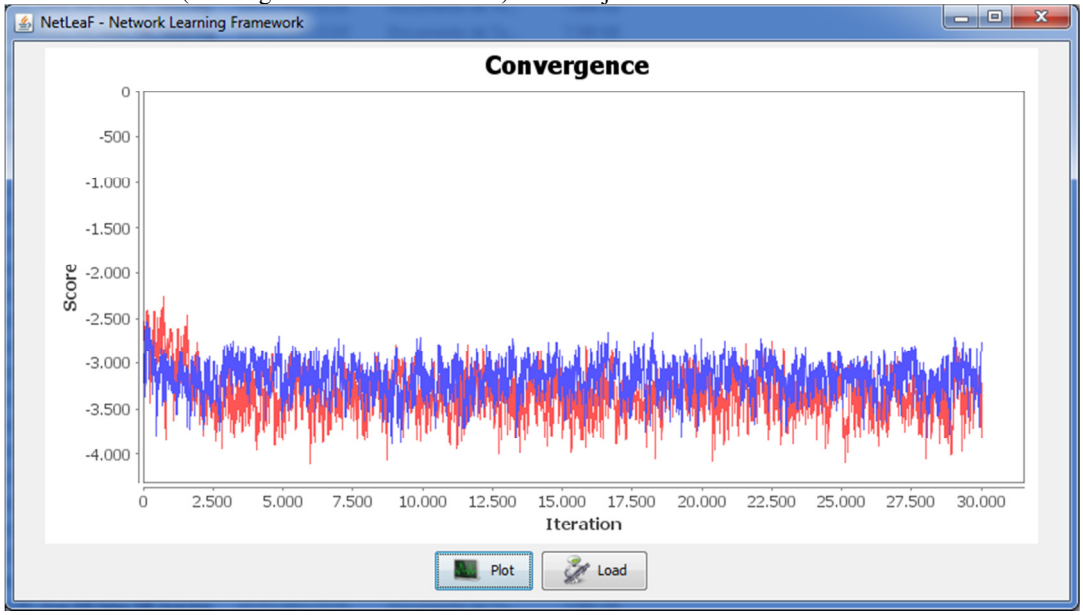


Figura 12 – Gráfico de convergência – Rede Pacientes HU sem testes estatísticos
(rede vazia como rede inicial) com conjuntos de dados de 500 casos.

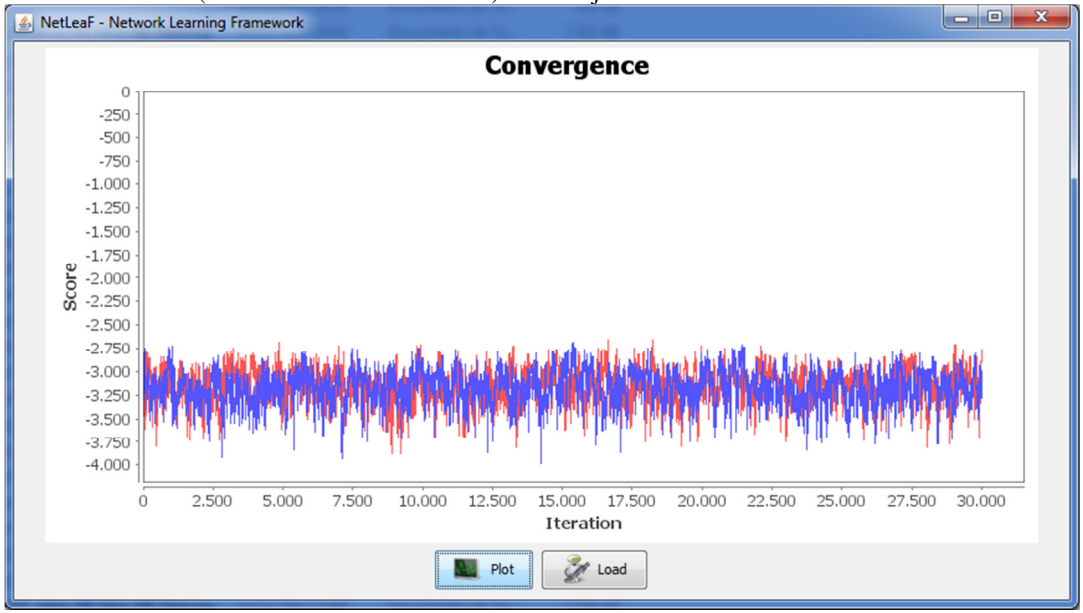


Figura 13 – Gráfico de convergência – Rede Pacientes HU com teste de correlação de Pearson (rede ingênua como rede inicial) com conjuntos de dados de 1000 casos.

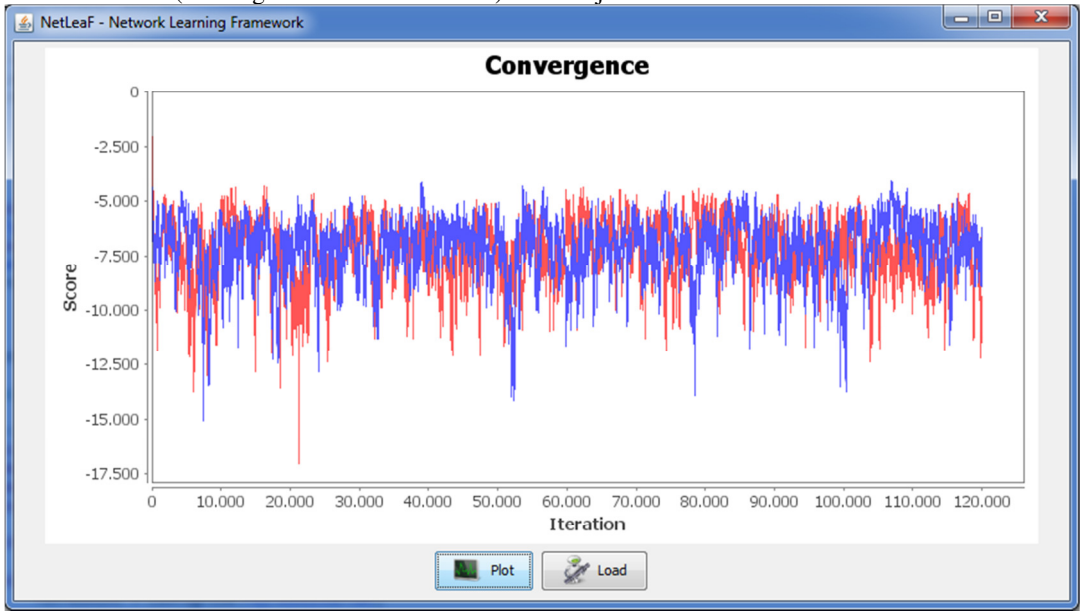


Figura 14 – Gráfico de convergência – Rede *Alarm* com teste de correlação de Pearson (rede vazia como rede inicial) com conjuntos de dados de 500 casos.

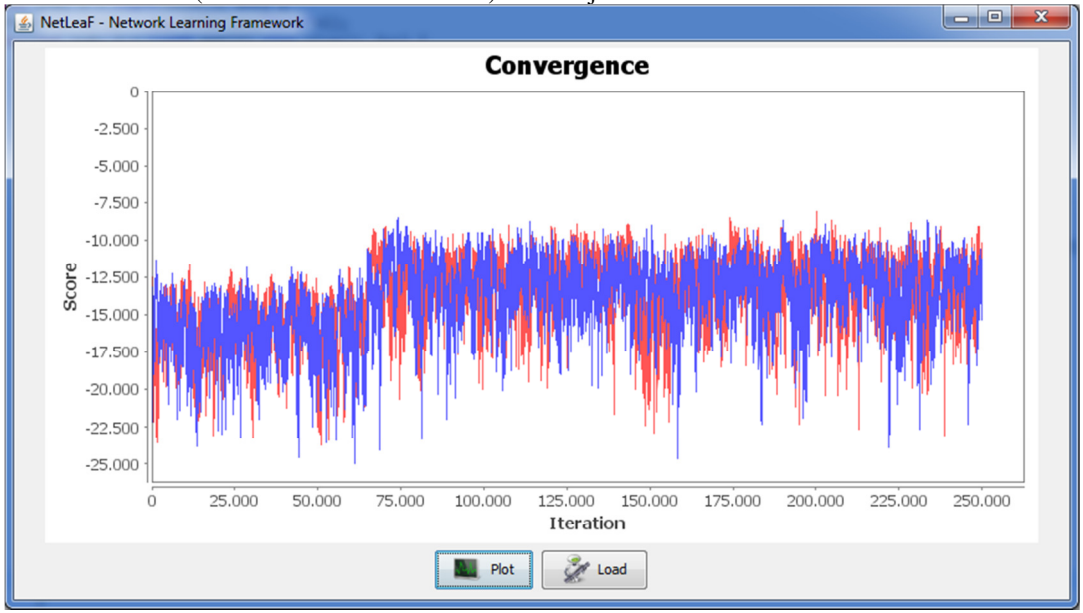
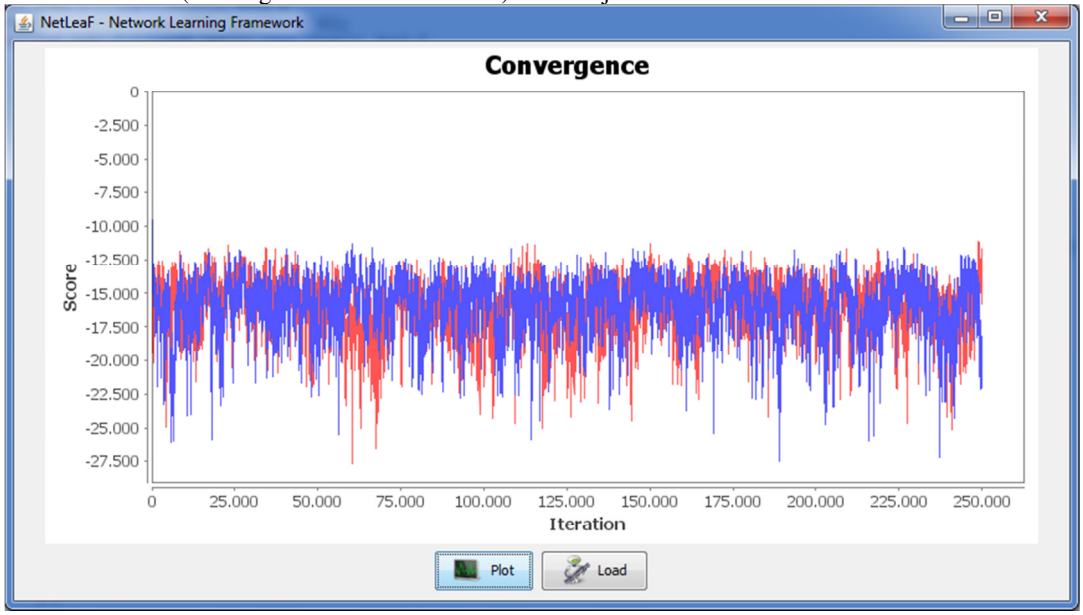


Figura 15 – Gráfico de convergência – Rede *Alarm* com teste de correlação de Pearson (rede ingênua como rede inicial) com conjuntos de dados de 500 casos.



Nas Figuras 14 e 15 estão os dados dos testes, relativos à convergência, da rede *Alarm* para o conjunto de testes de 500 casos. Na execução apresentada na Figura 14, foi utilizada como ponto de partida uma rede inicial totalmente desconectada e na Figura 15, uma rede ingênua. Nos testes com a rede *Alarm* com 1000 casos, as cadeias não apresentaram convergência após 350.000 iterações, mesmo com o uso dos testes estatísticos.

5.3 ANÁLISE FORMAL DA CONVERGÊNCIA

Existem duas formas de análise da convergência através da inspeção visual dos gráficos e a outra baseada em propriedades estatísticas das cadeias geradas durante a execução do algoritmo. Nesta seção são apresentados os resultados da análise formal realizada neste trabalho.

Tabela 15 – Análise formal da convergência.

Conjunto de teste	**Quantidade de Iterações**	**Quantidade de casos**	$\hat{R}$
Pacientes HU	12.000	100	1,005792
Pacientes HU	30.000	500	1,002178
Pacientes HU	60.000	1000	1,000556
Alarm	120.000	100	1,013740
Alarm	250.000	500	1,021639
Alarm	350.000	1000	6,463322

Esta análise foi feita utilizando o método de Gelman e Rubin (1992). Este método recomenda aceitar como garantia de convergência valores de $\hat{R} \leq 1,2$. Os resultados podem ser vistos na Tabela 15.

Pode-se ver na tabela que os testes com o conjunto de dados *Alarm* com 1000 casos não atingiram a convergência.

6 CONCLUSÕES

O desenvolvimento desta pesquisa teve como foco principal a criação de um processo automático de aprendizagem dos relacionamentos entre variáveis de determinado domínio de dados, além de permitir a criação de modelos de redes Bayesianas que pudessem ser utilizados para responder consultas sobre problemas específicos deste domínio. Adotou-se no método de construção do algoritmo a aprendizagem não supervisionada para que não houvesse a necessidade de nenhum tipo de interação do especialista do domínio no processo de extração de conhecimento executado pelo algoritmo. Como forma de implementar essa solução, e tendo em vista a grande dificuldade do problema a ser resolvido, decidiu-se utilizar os métodos de simulação de Monte Carlo via Cadeias de Markov, por serem reconhecidamente métodos extremamente eficientes na solução deste tipo de problema. Apesar dessas características, esses métodos ainda assim lidam com problemas classificados como NP-Completos, ou seja, impossíveis de calcular no pior caso. Isso acontece porque o tamanho do conjunto das possíveis estruturas de uma rede (dentre as quais será escolhido o que melhor representa os dados de acordo com alguma métrica escolhida) cresce exponencialmente conforme o número de variáveis aumenta.

Para minimizar essa característica do problema, optou-se por utilizar testes que permitissem diminuir o máximo possível o tamanho do espaço de busca, o que na prática significou reduzir o número de conexões entre os nós da rede. A justificativa para essa abordagem reside no fato de que se dois nós são fortemente correlacionados, provavelmente estarão presentes em todos os modelos de redes com *scores* altos. Através do uso desses testes, algumas conexões foram previamente fixadas na estrutura da rede e outras foram impostas como restrições, ou seja, durante a execução do algoritmo essas conexões não puderam ser adicionadas na função de geração de amostras de estruturas de redes.

Analisando as redes geradas durante as execuções do algoritmo, juntamente com os dados capturados na submissão dos casos de teste, pôde-se concluir que os resultados mostraram-se muito satisfatórios, principalmente devido às taxas de erros apresentadas nas matrizes de classificação. Pode-se supor que os bons resultados obtidos devem-se ao processo utilizado na coleta de modelos de rede, nos quais foram salvos os melhores modelos durante toda a execução do programa.

No que diz respeito à análise dos gráficos de convergência, nem todos os resultados apresentaram o retorno esperado. Nota-se em alguns

gráficos a presença de picos e vales nas linhas com os valores do *score* das redes. Provavelmente essa característica se deve à forma de geração de amostras, na qual os nós envolvidos em cada mudança no modelo de rede atual da cadeia foram selecionados aleatoriamente. Isto ocorre porque uma simples alteração em uma conexão da rede pode alterar drasticamente o valor do *score* desta rede. Esta constatação também pode ser confirmada através da análise formal da convergência, na qual o conjunto de dados *Alarm* com 1000 casos não atingiu a convergência.

Pode-se dizer que o ponto que mais comprometeu os resultados do algoritmo construído, no que diz respeito a acelerar a convergência da cadeia e que como consequência irá gerar modelos de rede que representem bem os dados utilizados no treinamento destas redes, foi o método utilizado na geração de amostras. Por escolher de forma totalmente aleatória os nós envolvidos na operação da rede, gerou, em várias iterações, modelos de rede totalmente diferentes e, como consequência, com *scores* totalmente distintos. Um dos requisitos dos algoritmos de Cadeias de Markov via Monte Carlo, para que a cadeia se misture o mais rápido possível, é que o processo de alteração no estado anterior ocorra de forma adequada. Isto quer dizer que é preciso encontrar uma forma de gerar movimentos nem tão pequenos e nem tão grandes nas amostras geradas. Uma possível solução para esse problema poderia ser implementada através do uso do conceito de cobertura de Markov, na qual apenas as conexões dos nós dentro de cada cobertura fossem alteradas. Com esta alteração é muito provável que os picos e vales registrados na convergência da cadeia fossem reduzidos drasticamente.

Outra possível alteração no algoritmo diz respeito à forma de cálculo utilizada na comparação do *score* atual com o *score* anterior (modelo de rede atual com modelo de rede anterior). Como não se tem disponível uma constante para normalizar os *scores* gerados, já que não é possível obter todos os *scores* de todas as redes existentes, poderia ser utilizada uma técnica que durante um determinado tempo da execução do algoritmo coletasse *scores* com valores altos e, a partir da média desses valores, fosse possível determinar a distribuição aproximada de cada novo *score* gerado.

REFERÊNCIAS

ANTON, Howard; BUSBY, Robert C. **Álgebra Linear Contemporânea.** Porto Alegre: Bookman, 2006.

BARBER, David. **Bayesian Reasoning and Machine Learning**. Cambridge: Cambridge University Press, 2012.

BARBETTA, Pedro Antonio; REIS, Marcelo Menezes; BORNIA, Antonio Cezar. **Estatística para Cursos de Engenharia e Informática**. São Paulo: Atlas, 2004.

BEINLICH, Ingo A.; SUERMONDT, G.; CHAVEZ, R. Martin; COOPER, Gregory. **The ALARM monitoring system:** a case study with two probabilistic inference techniques for belief networks. Second European Conference on Artificial Intelligence in Medicine. Berlin : Springer-Verlag, 1989.

BINDER, J.; KOLLER, D.; RUSSELL, S. J.; KANAZAWA, K. (1997). **Adaptive probabilistic networks with hidden variables.** Netherlands: Kluwer Academic Publishers, 1997. Machine Learning, 29, p213–244

BORGES, Lívia Costa. **Análise Bayesiana do modelo fatorial dinâmico para um vetor de séries temporais usando distribuições elípticas.** São Paulo: Universidade de São Paulo, 2008.

BROOKS, Steve; GELMAN, Andrew; JONES, Galin L.; MENG, Xiao-Li. **Handbook of Markov Chain Monte Carlo.** New York: CRC Press, 2011.

CALDEIRAS, André Machado et al. **Inteligência Computacional Aplicada à Administração, Economia e Engenharia em Matlab.** São Paulo: Thomson Learning, 2007.

CANO, Andrés; MASEGOSA, Andrés R.; MORAL, Serafin. **A Method for Integrating Expert Knowledge When Learning Bayesian Networks From Data.** Granada: IEEE Transactions on Systems, Man, and Cybernetics, - Part B: Cybernetics, Vol. 41, No. 5 p1382-1394, 2011.

CASELLA, George; BERGER, Roger L.. **Inferência Estatística.** São Paulo: Cengage Learning, 2010.

CHEESEMAN, Peter; STUTZ, John. **Bayesian classification (AutoClass):** theory and results. [S. l.]: Advances in knowledge discovery and data mining, Pages 153 – 180, 1996.

CHENG, Jie; GREINER, Russel. **Learning Bayesian Belief Network Classifiers:** Algorithms and System. Alberta: Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence, 2001.

CHENG, Jie; GREINER, Russel. **Comparing Bayesian Network Classifiers.** Alberta: Department of Computing Science, University of Alberta, Proceedings of the fifteenth international conference on uncertainty in artificial intelligence, 1999.

CHICKERING, David Maxwell; HECKERMAN, David, MEEK, Christopher. **Large-Sample Learning of Bayesian Networks is NP-Hard.** Redmond: Microsoft Research, Journal of Machine Learning Research 5, p1287–1330, 2004

COPPIN, Ben. **Inteligência Artificial.** Rio de Janeiro: LTC, 2010.

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. **Algoritmos: teoria e prática.** Rio de Janeiro: Campus, 2002.

COSTA, Felipe Schneider; PIRES, Maria Marlene de Souza; NASSAR, Silvia Modesto. **Analysis of Bayesian classifier accuracy.** New York: Journal of Computer Science 9, p1487-1495, 2013.

DARWICHE, Adnan. **Modeling and Reasoning with Bayesian Networks.** Cambridge: Cambridge University Press, 2009.

DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B.. **Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm.** [S. l.]: Journal of the Royal Statistical Society V39 p1-38, 1977.

DUDA, Richard. O.; HART, Peter E.; STORK, David G.. **Pattern Classification**. Nova Jersey: Wiley-Interscience, 2000.

EVANS, James R.; OLSON, David L.. **Introduction to simulation and risk analysis**. Nova Jersey: Prentice Hall, 1998.

FREITAS FILHO, Paulo José. **Introdução à Modelagem e Simulação de Sistemas com Aplicações Arena**. Florianópolis: Visual Books, 2008.

FRIEDMAN, Nir; GEIGER, Dan; GOLDSZMIDT, Moises. **Bayesian network classifiers**. Netherlands: Machine Learning, 29:131–163, 1997.

FRIEDMAN, Nir; GOLDSZMIDT, Moises; HECKERMAN, Dan. **Challenge:** Where is the Impact of Bayesian Networks in Learning?. Venue: In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, 1997.

GALVANIN, E. A. S.; **Extração Automática de Contornos de Telhados de Edifícios em um Modelo Digital de Elevação**, Utilizando Inferência Bayesiana e Campos Aleatórios de Markov. Presidente Prudente: Universidade Estadual Paulista, 2007.

GAMERMAN, D.; LOPES, H. F.; **Markov Chain Monte Carlo**: Stochastic Simulation for Bayesian Inference. 2nd. ed. Londres: Chapman & Hall CRC, 2006.

GELMAN, A.; RUBIN, D.; **Inference from iterative simulation using multiple sequences**. [S. l.]: Statistical science 7, 1992.

GILKS, W. R.; RICHARDSON, S.; SPIEGELHALTER, D. J.. **Markov Chain Monte Carlo in Practice**. New York: Chapman & Hall/CRC, 1996.

GOGATE, Vibahv; DECHTER, Rina. **SampleSearch:** Importance sampling in presence of determinism. Essex: Elsevier Science Publishers, Journal Artificial Intelligence V 175 Issue 2, February, 2011 p694-729.

GUO, Peng. **An EM-MCMC algorithm for Bayesian structure learning**. Computer Science and Information Technology. ICCSIT. Beijing, 2009. Disponível em

http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5234973. Acesso em: 05 novembro 2011.

HASTINGS, W. K. **Monte Carlo Sampling Methods Using Markov Chains and Their Applications**. Oxford: Biometrica p57, 97-109, 1970.

HUANG, Yuguang; LI, Lei. **Naive Bayes classification algorithm based on small sample set**. Beijing: IEEE International Conference on Cloud Computing and Intelligence Systems – CCIS, 2011.

JANZURA, Martin; NIELSEN, Jan. **A Simulated Annealing-Based Method for Learning Bayesian Networks from Statistical Data**. New York: International Journal of Intelligent Systems - Uncertainty Processing p21, 335-348, 2006.

JUN, Xie., LI, Wang. **PCMHS-Based Algorithm for Bayesian Networks Online Structure Learning**. Chongqing: International Forum on Computer Science-Technology and Applications p310-314, 2009.

JUNG, Fernando Carlos. **Metodologia Científica Ênfase em Pesquisa Tecnológica**. 2003. Disponível em <http://www.jung.pro.br>. Acesso em: 02 outubro 2011.

LANGLEY, Pat; IBA, Wayne; THOMPSON, Kevin. **An analysis of Bayesian classifiers**. Menlo Park, CA: AAAI Press,Tenth National Conference on Artificial Intelligence p223–228, (1992).

LARRANAGA, Pedro; POZA, Mikel; YURRAMENDI, Yosu; MURGA, Roberto H., KUIJPERS, Cindy M. H. **Structure Learning of Bayesian Networks by Genetic Algorithms:** Performance Analysis of Control Parameters. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 9, September 1996.

LEE, Chang-Hwan; GUTIERREZ, Fernando; DOU, Dejing. **Calculating Feature Weights in Naive Bayes with Kullback-Leibler Measure**. Vancouver: IEEE 11th International Conference on Data Mining – ICDM p1146-1151, 2011.

LIAO, Qin; QIU, Zhicong; ZENG, Jiepeng. **Fuzzy Bayesian Networks and its application in pressure equipment's security alerts**. Shanghai: Natural Computation (ICNC) p1507-1511, 2011.

LUGER, George F.. **Inteligência Artificial:** Estruturas e estratégias para a solução de problemas complexos. Porto Alegre: Bookman, 4 ed., 2004.

LUNA, José Eduardo Ochoa Lima. **Algoritmos EM para Aprendizagem de Redes Bayesianas a partir de Dados Incompletos**. Mato Grosso do Sul: UFMS, 2004.

LUSTOSA, Paulo Roberto Barbosa; GARCIA, Solange; BARROS, Nara Rosa. **Aplicabilidade do método de simulação de monte carlo na previsão dos custos de produção de companhias industriais:** o caso da companhia vale do rio doce. [S. l.]: Revista de Contabilidade e Organizações - OPEN JOURNAL SYSTEMS v. 4, n. 10, 2010.

MARSLAND, Stephen. **Machine Learning: an algorithmic perspective**. New York: Chapman & Hall, 2009.

MASEGOSA, Andrés R., MORAL, Serafín. **New skeleton-based approaches for Bayesian structure learning of Bayesian networks**. Department of Computer Science and Artificial Intelligence, University of Granada, Spain, 2013.

MAYER, Helídia C. **Validação de uma Base de Conhecimento para um Sistema Especialista Bayesiano de Apoio ao Diagnóstico do Risco Metabólico**. Florianópolis: UFSC, 2012

NAEEM, Muhammad; ASGHAR, Sohail. **A novel mutual dependence measure in structure learning**. Pakistan: Department of Computer Science, Mohammad Ali Jinnah University, 2013.

NEAL, Radford M.. **Annealed importance sampling**. Toronto: University of Toronto, Dept. of Statistics and Computing, 11:125–139, 2001.

NEWELL, Allen; SIMON, Herbert A. **Computer Science as Empirical Inquiry:** [S. l.]: Symbols and Search. Turing Award Lecture, 1975.

NIINIMÄKI, Teppo; KOIVISTO, Mikko. **Annealed Importance Sampling for Structure Learning in Bayesian Networks**. Helsinki: HIIT & Department of Computer Science, University of Helsinki, 2013.

NORSYS, **Netica Bayesian Networks Software from Norsys**. Disponível na Web em: <http://www.norsys.com/netlib/alarm.htm>. Acesso em: 22 julho 2012.

PEARL, Judea. **Probabilistic Reasoning in Inteligent Systems**. New York: Morgan Kaufmann, 1988.

POCOCK, Adam. **JavaMI Toolbox**, University of Manchester. Disponível em < http://www.cs.man.ac.uk/~gbrown/software/>. Acesso em 05 maio 2013.

PRESS, William H.; TEUKOLSKY, Saul A.; VETTERLING, William T.; FLANNERY, Brian P.. **Métodos Numéricos Aplicados:** Rotinas em C++, 3ed. Porto Alegre: Bookman, 2011.

R C. T.; **R**: A language and environment for statistical computing. Vienna: R Foundation for Statistical Computing. Disponível em http://www.R-project.org/. Acesso em 20 junho 2013.

RIGGELSEN, Carsten. **MCMC Learning of Bayesian Network Models by Markov Blanket Decomposition**. Netherlands: ECML, 329-340, 2005.

RISH, Irina. **An empirical study of the naive Bayes classifier**. Wroclaw: IJCAI-01 workshop on Empirical Methods in AI, 2001.

RODRIGUES, Fabiene Silva. **Métodos de agrupamento na análise de dados de expressão gênica**. São Carlos: UFSCar, CCET, Mestrado em Estatística, 2009.

RUSSEL, Stewart; NORVIG, Peter. **Inteligência Artificial**. Rio de Janeiro: Elsevier, 2.ed., 2004.

SALAMA, Khalid M.; FREITAS, Alex A.. **Learning Bayesian network classifiers using ant colony optimization**. New York: Springer Science and Business Media, 2013.

SANCHES, Marcelo Kaminski. **Aprendizado de máquina semi-supervisionado: proposta de um algoritmo para rotular exemplos a partir de poucos exemplos rotulados**. São Paulo: ICMC–USP, 2003.

SHI, Hui-Feng; XING, Milan. **The improved MC^3 algorithm and Bayesian network learning**. Machine Learning and Cybernetics, 2008, (July), 12-15. Disponível em http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4620692. Acesso em 25 maio 2012.

SILVA, Edna Lúcia da Silva; MENEZES, Estera Muszkat. **Metodologia da Pesquisa e Elaboração de Dissertação**. Florianópolis: Universidade Federal de Santa Catarina – UFSC, 2001.

SIMON, Herbert E.. **Search and reasoning in problem solving**. Essex: Elsevier Science Publishers, Journal Artificial Intelligence V21(1-2), 1983 p7-29, 1983.

SMITH, B. J.; **boa**: An R Package for MCMC Output Convergence Assessment and Posterior Inference. [S. l.]: Journal of Statistical Software, 21(11), 1-37, 2007.

TURING, Alan M.. **Computing Machinery and Intelligence**. Oxford University Press, Mind, New Series, Vol. 59, No. 236 (Oct., 1950), pp. 433-460, 1950. Disponível em http://mind.oxfordjournals.org/cgi/doi/10.1093/mind/LIX.236.433. Acesso em 18 de junho 2012.

ZADEH, Lotfi A. **Fuzzy sets**. [S. l.]: Information and Control, 8, 338–353, 1965.

ZHANG, Harry. **The Optimality of Naive Bayes**. American Association for Artificial Intelligence, 2004.

ZHANG, Harry; SHENG, Shengli. **Learning Weighted Naive Bayes with Accurate Ranking.** Washington, DC: ICDM, pp.567-570, Fourth IEEE International Conference on Data Mining (ICDM'04), 2004.

ZHANG, Shao-Zhong; LIU, Lu. **MCMC Samples selecting for online Bayesian network**. Kunming: Machine Learning, (July), 12-15, 2008.

WAZLAWICK, Raul S. **Metodologia de pesquisa para ciência da computação**. Rio de Janeiro: Elsevier, 2008.

WEISS, Gerhard. **Multiagent Systems - A Modern Approach to Distributed Modern Approach to Artificial Intelligence**. Massachusetts: The MIT Press, Cambridge, 1999.

WITTEN, Ian H.; FRANK, Eibe; HALL, Mark A.. **Data Mining:** pratical machine learning tools and techniques. Massachusetts: British Library, 2011.

APÊNDICE A – Nós da rede *Alarm*

A rede *Alarm* possui os seguintes nós:

1. CVP (central venous pressure): classes LOW, NORMAL e HIGH.
2. PCWP (pulmonary capillary wedge pressure): classes LOW, NORMAL e HIGH.
3. HIST (history): classes TRUE e FALSE.
4. TPR (total peripheral resistance): classes LOW, NORMAL e HIGH.
5. BP (blood pressure): classes LOW, NORMAL e HIGH.
6. CO (cardiac output): classes LOW, NORMAL e HIGH.
7. HRBP (heart rate / blood pressure): classes LOW, NORMAL e HIGH.
8. HREK (heart rate measured by an EKG monitor): classes LOW, NORMAL e HIGH.
9. HRSA (heart rate / oxygen saturation): classes LOW, NORMAL e HIGH.
10. PAP (pulmonary artery pressure): classes LOW, NORMAL e HIGH.
11. SAO2 (arterial oxygen saturation): classes LOW, NORMAL e HIGH.
12. FIO2 (fraction of inspired oxygen): classes LOW e NORMAL.
13. PRSS (breathing pressure): classes ZERO, LOW, NORMAL e HIGH.10 alarm
14. ECO2 (expelled CO2): classes ZERO, LOW, NORMAL e HIGH.
15. MINV (minimum volume): classes ZERO, LOW, NORMAL e HIGH.
16. MVS (minimum volume set): classes LOW, NORMAL e HIGH.
17. HYP (hypovolemia): classes TRUE e FALSE.
18. LVF (left ventricular failure): classes TRUE e FALSE.
19. APL (anaphylaxis): classes TRUE e FALSE.
20. ANES (insufficient anesthesia/analgesia): classes TRUE e FALSE.
21. PMB (pulmonary embolus): classes TRUE e FALSE.
22. INT (intubation): classes NORMAL, ESOPHAGEAL e ONESIDED.
23. KINK (kinked tube): classes TRUE e FALSE.
24. DISC (disconnection): classes TRUE e FALSE.

25. LVV (left ventricular end-diastolic volume): classes LOW, NORMAL e HIGH.
26. STKV (stroke volume): classes LOW, NORMAL e HIGH.
27. CCHL (catecholamine): classes NORMAL e HIGH.
28. ERLO (error low output): classes TRUE e FALSE.
29. HR (heart rate): classes LOW, NORMAL e HIGH.
30. ERCA (electrocauter): classes TRUE e FALSE.
31. SHNT (shunt): classes NORMAL e HIGH.
32. PVS (pulmonary venous oxygen saturation): classes LOW, NORMAL e HIGH.
33. ACO2 (arterial CO2): classes LOW, NORMAL e HIGH.
34. VALV (pulmonary alveoli ventilation): classes ZERO, LOW, NORMAL e HIGH.
35. VLNG (lung ventilation): classes ZERO, LOW, NORMAL e HIGH.
36. VTUB (ventilation tube): classes ZERO, LOW, NORMAL e HIGH.
37. VMCH (ventilation machine): classes ZERO, LOW, NORMAL e HIGH.