Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Departamento de Computação

Programa de Pós-Graduação em Ciência da Computação

A Seleção de Atributos e o Aprendizado Supervisionado de Redes Bayesianas no Contexto da Mineração de Dados

SEBASTIAN DAVID CARVALHO DE OLIVEIRA GALVÃO

.

SÃO CARLOS - SP

Outubro/2007

A Seleção de Atributos e o Aprendizado Supervisionado de Redes Bayesianas no Contexto da Mineração de Dados

Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Departamento de Computação
Programa de Pós-Graduação em Ciência da Computação

A Seleção de Atributos e o Aprendizado Supervisionado de Redes Bayesianas no Contexto da Mineração de Dados

Sebastian David Carvalho de Oliveira Galvão

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Departamento de Computação, da Universidade Federal de São Carlos, como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Inteligência Artificial – IA.

SÃO CARLOS - SP

Outubro/2007

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

G182sa

Galvão, Sebastian David Carvalho de Oliveira.
A seleção de atributos e o aprendizado supervisionado de redes bayesianas no contexto da mineração de dados / Sebastian David Carvalho de Oliveira Galvão. -- São Carlos : UFSCar, 2008.
109 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2007.

1. Aprendizado de computador. 2. Data mining (Mineração de dados). 3. Redes Bayesianas. I. Título.

CDD: 006.31(20ª)

Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Programa de Pós-Graduação em Ciência da Computação

"A Seleção de Atributos e o Aprendizado Supervisionado de Redes Bayesianas no Contexto da Mineração de Dados"

SEBASTIAN DAVID CARVALHO DE OLIVEIRA GALVÃO

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Membros da Banca:

Prof. Dr. Estevam Rafael Hruschka Junior
(DC/UFSCar)

Profa. Dra. Maria do Carmo Nicoletti
(DC/UFSCar)

Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho
(ICMC/USP)

Prof. Dr. Fábio Gagliardi Cozman
(POLI/USP)

São Carlos
Outubro/2007

Aquilo que se sabe, saber que se sabe;
aquilo que não se sabe, saber que não se sabe;
na verdade é este o saber.
Confúcio

A Deus e a toda sua obra.

Agradecimentos

Primeiramente a Deus, por tudo que me proporcionou, incluindo a presença das pessoas aqui citadas e tantas outras que me auxiliaram de diversos modos.

Parafraseando o grande inventor Alexander Graham Bell, qualquer conhecimento adquirido envolve a cooperação de várias mentes. Pode-se receber o crédito de se concluir uma jornada, mas ao se olhar para trás vê-se que o crédito é tanto quanto de outros na verdade. Agradeço, primeiramente, ao meu orientador, o Prof. Dr. Estevam Rafael Hruschka Junior. Agradecer não somente pela excepcional orientação recebida neste trabalho, mas por toda sua generosidade e por todo conhecimento que compartilhou comigo desde os tempos na graduação, ainda em Curitiba/PR. É uma pessoa que admiro muito e me incentivou e me despertou à vida científica e serviu, e sempre servirá para mim, como um exemplo.

À minha mãe, Telma, a quem agradeço por tudo e sem quem nunca teria chegado a lugar algum e ao meu irmão, Igor, que sempre me apoiou.

À minha namorada, Cristiane, por todo carinho, compreensão e ajuda.

Aos meus amigos e colegas que conviveram comigo e me auxiliaram de alguma forma durante o tempo da realização deste trabalho: Bruno Kimura, Edimilson Santos, Danilo Sanches, e todos os demais colegas.

Aos professores que contribuíram para o trabalho, incluindo a Prof. Dra. Maria do Carmo Nicoletti, o Prof. Dr. André Ponce de Leon F. de Carvalho e aos professores do Departamento de Computação da UFSCar.

A todos os funcionários, pelos serviços prestados e todo auxílio concedido.

À FAPESP, pela concessão da bolsa de estudos e pelo apoio e auxílio recebidos.

RESUMO

As técnicas de Descoberta de Conhecimento em Bancos de Dados (KDD), também chamadas de Mineração de Dados, surgiram da grande necessidade de se obter mais informação sobre os dados armazenados por organizações, como empresas, grandes corporações e instituições de pesquisa. As Redes Bayesianas (RBs) podem ser consideradas como uma forma de representação do conhecimento baseada no raciocínio probabilístico e possuem características que as tornam muito adequadas para tarefas de descoberta de conhecimento em bancos de dados. Por isso, este é um campo de aplicação efervescente nos últimos anos. O aprendizado automático de RBs e Classificadores Bayesianos (CBs) busca identificar uma RB (ou CB) que represente o relacionamento entre as variáveis de um determinado conjunto de dados, mas como este é um problema NP-completo o espaço de busca se torna muito amplo na maioria das aplicações. Por este motivo, muitos algoritmos exploram alguma forma de redução do espaço de busca para tornar o processo de aprendizado computacionalmente viável. Esta dissertação de mestrado apresenta um método (MarkovPC) de aprendizado de CBs que visa exatamente reduzir o espaço de busca durante a indução de um classificador a partir de dados. Para tanto, toma-se como base algoritmos de aprendizado de RB da classe IC (Independência Condicional) e o conceito de Markov Blanket. Resultados obtidos através de experimentos realizados com 10 conjuntos de dados mostram que o MarkovPC é capaz de reduzir o esforço computacional do processo de indução de um classificador Bayesiano e manter a qualidade do classificador induzido (em termos de taxa de classificação correta).

ABSTRACT

The Knowledge Discovery in Databases (KDD) techniques have grown from the need for obtain more information about the data stored by organizations, such as, enterprise companies and research institutes. Bayesian Networks (BNs) can be considered as a probabilistic reasoning based model to represent knowledge and are very adequate to KDD tasks. In the last years, Bayesian Networks (BNs) have been applied in many supervised and unsupervised learning successful applications. The process to induce BNs and Bayesian Classifiers (BCs) from data tries do identify a BN (or a BC) able to represent the relationship among the variables of a certain data set. However, this is a NP-complete problem and, thus, its search space may become very large in most applications. That is the reason why many algorithms explore some way to reduce the search space in order to make the learning process computationally viable. In this master's thesis a new Conditional Independence based approach to induce BCs from data is proposed and implemented. Such approach is based on the Markov Blanket concept in order to impose some constraints and optimize the traditional PC learning algorithm. Experiments performed with ten data sets revealed that the proposed approach tends to execute fewer comparisons than the traditional PC. The experiments also show that the implemented algorithm produce competitive classification rates when compared with both, PC and NaiveBayes.

SUMÁRIO

1 INTRODUÇÃO 14

1.1 OBJETIVOS 15

1.2 METODOLOGIA 16

1.3 ORGANIZAÇÃO DO TRABALHO 16

2 REDES BAYESIANAS 18

2.1 APRENDIZADO DE REDES BAYESIANAS 21

2.1.1 Independência Condicional 22

2.1.1.1 Assertivas de dependência 24

2.1.1.2 Testes de dependência 25

2.1.1.3 Separação orientada 25

2.1.1.4 Critério m-separação 27

2.1.1.5 Separador desconhecido 27

2.1.1.6 Mapeamento 27

2.1.1.7 Orientação e independência 29

2.1.1.8 Indução de representante 29

2.1.2 Aprendizado por Independência Condicional 30

2.1.2.1 O algoritmo IC 32

2.1.2.2 O algoritmo PC 34

2.1.2.3 Exemplo de execução do algoritmo PC 37

2.2 CONCLUSÕES 39

3 DESCOBERTA DE CONHECIMENTO EM BANCOS DE DADOS - KDD 40

3.1 SELEÇÃO DE ATRIBUTOS 41

3.1.1 Abordagem wrapper 42

3.1.2 Abordagem filtro 43

3.1.3 Abordagem embutida 45

3.2 CLASSIFICAÇÃO 46

3.3 CONCLUSÕES 48

4 O ALGORITMO MARKOVPC 49

4.1 REDE ALARM 54

4.1.1 Descrição dos experimentos 55

4.1.2 Análises dos experimentos 57

4.1.3 Resultados de experimentos na rede ALARM para diferentes conjuntos de dados 63

4.2 REDES SYNTHETIC 65
4.2.1 Synthetic1 ..... *66*
4.2.2 Synthetic2 ..... *69*
4.2.3 Synthetic3 ..... *72*
4.3 CAR 75
4.4 KR-VS-KP 77
4.5 LUNG-CANCER 79
4.6 PATIENT 81
4.7 SOLAR-FLARE 1 82
4.8 SOLAR-FLARE 2 84
4.9 ANÁLISE GERAL DOS RESULTADOS 86
4.10 CONCLUSÕES 88

5 TESTE DE INDEPENDÊNCIA CONDICIONAL UTILIZANDO UMA ESTRATÉGIA BASEADA EM MARKOV BLANKET 89

5.1 TESTE DE INDEPENDÊNCIA POR PROBABILIDADE CONJUNTA 94
5.2 TESTE DE INDEPENDÊNCIA POR PONTUAÇÃO 94
5.3 EXPERIMENTOS E RESULTADOS 95
5.4 CONCLUSÕES 97

6 CONCLUSÕES 99

6.1 TRABALHOS FUTUROS 100
6.2 PUBLICAÇÕES 100

7 REFERÊNCIAS 102

LISTA DE FIGURAS

Figura 2.1. Um exemplo de Rede Bayesiana 19
Figura 2.2. O Markov Blanket do nó X 20
Figura 2.3. Um grafo direcionado 26
Figura 2.4. Rede Bayesiana em construção 29
Figura 2.5. Estruturas equivalentes 30
Figura 2.6. Algoritmo IC, adaptado de [53] 33
Figura 2.7. Pseudocódigo do algoritmo PC, adaptado de [17,60]. 36
Figura 2.8a. Exemplo de aplicação do algoritmo PC [44]. 37
Figura 2.8b. Exemplo de aplicação do algoritmo PC [44] (continuação) 38
Figura 2.9. Padrão final produzido pelo PC para o exemplo anterior [44]. 38
Figura 3.1. Fases do processo de extração de conhecimento de bancos de dados. 40
Figura 3.2. Fluxograma da abordagem wrapper 43
Figura 3.3. A abordagem filtro 44
Figura 3.4. A abordagem embutida 45
Figura 3.5. Classificador NaiveBayes 48
Figura 4.1. O Algoritmo MarkovPC 51
Figura 4.2. Metodologia dos testes 53
Figura 4.3. Rede Bayesiana original ALARM [56] 54
Figura 4.4. Rede ALARM gerada pelo PC tradicional 56
Figura 4.5. Hypovolemia gerada por MarkovPC 57
Figura 4.6. Anaphylaxis gerada por MarkovPC 57
Figura 4.7. LVFailure gerada por MarkovPC 58
Figura 4.8. Insufficient Anesthesia (InsuffAnesth) gerada por MarkovPC 59
Figura 4.9. PulmEmboulus gerada por MarkovPC 60
Figura 4.10. Intubation gerada por MarkovPC 60
Figura 4.11. KinkedTube gerada por MarkovPC 62
Figura 4.12. Disconnect gerada por MarkovPC 62
Figura 4.13. Rede original Synthetic1 66
Figura 4.14. Synthetic1 gerada pelo PC tradicional 67
Figura 4.15. Synthetic1 gerada pelo MarkovPC 67
Figura 4.16. Rede Synthetic2 69
Figura 4.17. Synthetic2 gerada pelo PC tradicional 70
Figura 4.18. Synthetic2 gerada pelo MarkovPC 70
Figura 4.19. Rede Synthetic3 72
Figura 4.20. Synthetic3 gerada pelo PC tradicional 73
Figura 4.21. Synthetic3 gerada pelo MarkovPC 73

Figura 4.22. Rede Car gerada pelo PC tradicional....75
Figura 4.23. Rede Car gerada pelo MarkovPC 75
Figura 4.24. Rede Kr-vs-Kp gerada pelo algoritmo PC tradicional 77
Figura 4.25. Rede Kr-vs-Kp gerada pelo algoritmo MarkovPC 78
Figura 4.26. Rede Lung-Cancer gerada pelo PC tradicional....79
Figura 4.27. Rede Lung-Cancer gerada pelo MarkovPC 80
Figura 4.28. Rede Patient gerada pelos algoritmos PC e MarkovPC 81
Figura 4.29. Rede Solar-Flare 1 gerada pelo PC tradicional....83
Figura 4.30. Rede Solar-Flare 1 gerada pelo MarkovPC....83
Figura 4.31. Rede Solar-Flare 2 gerada pelo PC tradicional....85
Figura 4.32. Rede Solar-Flare 2 gerada pelo MarkovPC....85
Figura 5.1. Um algoritmo de Independência Condicional genérico....90
Figura 5.2. Pseudo-código do algoritmo PC, adaptado de [17,60]. 91
Figura 5.3. Um algoritmo genérico de independência condicional usando MB....92

LISTA DE TABELAS

Tabela 2.1. Símbolos usados neste trabalho 23
Tabela 2.2. Vantagens e desvantagens da classe IC 31
Tabela 4.1. Resultados de experimentos na rede ALARM 63
Tabela 4.2. Resultados da ALARM com 10.000 registros 64
Tabela 4.3. Resultados da ALARM com 1.000 registros 64
Tabela 4.4. Resultados da ALARM com 100 registros 65
Tabela 4.5. Resultados da ALARM com 50 registros 65
Tabela 4.6. Resultados de Synthetic1 68
Tabela 4.7. Resultados de Synthetic2 71
Tabela 4.8. Resultados de Synthetic3 74
Tabela 4.9. Resultados da base Car 76
Tabela 4.10. Resultados da base Kr-vs-Kp 78
Tabela 4.11. Resultados da base Lung-Cancer 80
Tabela 4.12. Resultados da base Patient 81
Tabela 4.13. Domínio das variáveis da base Solar-Flare 1 82
Tabela 4.14. Resultados da base Solar-Flare 1 84
Tabela 4.15. Resultados da base Solar-Flare 2 85
Tabela 4.16. Resultados gerais 87
Tabela 5.1. Testes de Independência Condicional 89
Tabela 5.2. Resultados do método de IC usando uma estratégia baseada em MB 98

1 INTRODUÇÃO

No contexto da tecnologia da informação, uma das principais motivações para o crescente interesse em mineração de dados é a extensa disponibilidade de grandes quantidades de dados e a necessidade de transformar esses dados em informação útil e em conhecimento [1]. Pode-se dizer, portanto, que as técnicas de descoberta de conhecimento em bancos de dados [2,3], conhecidas também como mineração de dados, surgiram da necessidade de se obter mais informação sobre os dados armazenados por empresas e grandes corporações [3]. Esses dados têm sido coletados das mais diferentes formas devido ao avanço da tecnologia, especialmente transações eletrônicas como compras em supermercados, pela Internet, movimentações bancárias, entre diversos outros exemplos presentes no cotidiano. A capacidade de armazenamento das organizações evoluiu muito também, facilitando a manutenção de quantidades de dados cada vez maiores.

A análise de tantos dados guardados, torna-se uma tarefa muito difícil para seres humanos, sendo necessário então um processo automático. O ser humano tem dificuldades para tratar o volume de dados normalmente encontrados nas organizações, bem como em compreendê-los, ainda mais quando se leva em conta, a forma como as informações são armazenadas, que é uma maneira eficiente para os computadores, mas não para as pessoas.

Tanta informação e dados armazenados nos bancos de dados corporativos podem gerar um conhecimento útil para subsidiar as decisões humanas nos processos de tomada de decisão. Esse conhecimento oriundo dos dados deve ser representado de alguma forma, tal como uma Rede Bayesiana (RB), por exemplo. As RBs são baseadas no raciocínio probabilístico e possuem características interessantes que as tornam adequadas em tarefas de descoberta de conhecimento em banco de dados [3]. Somente ao final da década de 90, as RBs ganharam mais força em aplicações de mineração de dados, e, devido à relativa novidade que isso representa, muitas propostas ainda encontram-se em aberto.

Na mineração de dados, tanto o aprendizado automático quanto o processo de inferência em uma estrutura de representação do conhecimento são

fundamentais. Atualmente, na área das RBs, os estudos sobre aprendizado automático vêm recebendo bastante atenção dos pesquisadores por ser um tema ainda com resultados menos consolidados que os já obtidos com os mecanismos de inferência. O aprendizado automático de RBs e de Classificadores Bayesianos (CBs) busca identificar uma RB (ou CB) que melhor represente um determinado conjunto de dados; como este é um problema NP-completo [18], o espaço de busca se torna muito amplo na maioria das aplicações. Por este motivo, muitos algoritmos exploram alguma forma de redução do espaço de busca para tornar o processo de aprendizado computacionalmente viável. Neste sentido, utilizando-se Redes Bayesianas e seleção de atributos, pode-se otimizar o processo de descoberta de conhecimento em bancos de dados através da proposta e implementação de algoritmos capazes de oferecer ganhos em termos de qualidade, medida através de taxas de classificação, de esforço computacional e de poder de representação, uma vez que uma adequada seleção de atributos representa o conhecimento de forma mais objetiva.

1.1 Objetivos

O objetivo deste trabalho é apresentar, discutir e avaliar empiricamente a otimização do aprendizado de Redes Bayesianas, utilizando-se do conceito de seleção de atributos, para a aplicação em tarefas de descoberta de conhecimento em bancos de dados. Para tanto, utiliza-se o conceito dos métodos de aprendizado da classe de Independência Condicional (IC)[4], para definir uma estrutura de Rede Bayesiana adequada e que tenha um número reduzido de atributos. Desta forma, o algoritmo de aprendizado pode ser otimizado em dois aspectos, a saber: tempo de aprendizado e qualidade do classificador. A rapidez do aprendizado pode ser aumentada pelo fato de se trabalhar com um número menor de atributos; a qualidade do classificador tende a aumentar dado que apenas os atributos relevantes para a classificação deverão ser utilizados. Tem-se então como objetivos específicos:

- A proposta de um método embutido de seleção de atributos (com base no conceito de independência condicional) que será aplicado na otimização de métodos de aprendizado de Redes Bayesianas a partir de dados;

- A avaliação da adequação do método desenvolvido para o algoritmo PC [23] de aprendizado de Redes Bayesianas baseado em independência condicional;
- A aplicação da mesma estratégia de seleção de atributos também ao processo de teste de independência condicional usado pelos algoritmos de aprendizado baseados em independência condicional.

1.2 Metodologia

O conceito de Markov Blanket (MB) é uma forma de se identificar um subconjunto dos atributos relevantes em uma base de dados; a tarefa de se identificar um MB consistente, entretanto, não é trivial [39]. Uma RB pode ser utilizada como modelo de identificação de Markov Blanket [17]. Desta forma, algoritmos de aprendizado automático de RBs podem ser aplicados diretamente na tarefa de seleção de atributos de bases de dados. Resultados obtidos nos trabalhos [16, 17 e 40], se utilizam de métodos clássicos de aprendizado de Redes Bayesianas para a identificação de MBs a serem utilizados na seleção de atributos. A idéia central do presente projeto é utilizar o conceito de independência condicional para identificar o MB da variável classe em um problema de aprendizado supervisionado. Assim, não é necessário que um algoritmo de aprendizado percorra todo o espaço de busca, definido pelo método de aprendizado, para a identificação do MB e o MB encontrado já é, em si, o subconjunto de atributos relevantes. Pode-se dizer então que se pretende propor um algoritmo que aprenda uma Rede Bayesiana reduzida, contendo apenas as variáveis mais relevantes para o problema de classificação.

1.3 Organização do Trabalho

O restante deste texto busca mostrar os conceitos básicos necessários para a compreensão do método proposto, bem como apresentar o método de indução de classificadores Bayesianos MarkovPC, os experimentos realizados e discutir os resultados obtidos. A organização do documento está definida da seguinte forma: o Capítulo 2 apresenta os conceitos fundamentais da teoria de redes *Bayesianas*, incluindo a definição de inferência, a descrição do processo de aprendizado de estrutura e

parâmetros, apresentando algoritmos de aprendizado de redes *Bayesianas* e de classificadores.

O Capítulo 3 aborda o processo de descoberta de conhecimento em bancos de dados (KDD), com ênfase nas fases de seleção de atributos e classificação. Essas fases são as mais beneficiadas pelos métodos aqui propostos.

No Capítulo 4 é apresentado, discutido e analisado o algoritmo MarkovPC, que é o algoritmo de aprendizado de CBs proposto neste trabalho para otimização da indução de Classificadores Bayesianos usando o conceito de Markov Blanket.

O Capítulo 5 apresenta uma contribuição adicional deste trabalho, que é a utilização do conceito de Markov Blanket, conforme utilizado pelo MarkovPC, no processo de identificação de independências condicionais.

E, por fim, o Capítulo 6 encerra o documento com as conclusões, destacando-se as contribuições, perspectivas atuais e sugestões de trabalhos futuros.

2 REDES BAYESIANAS

As Redes Bayesianas (RBs) podem ser consideradas como uma forma de representação do conhecimento baseada no raciocínio probabilístico. Segundo [3], as RBs possuem características bastante importantes que as tornam muito adequadas para tarefas de descoberta de conhecimento em bancos de dados. Como o objetivo deste trabalho foca diretamente as Redes Bayesianas e algoritmos de aprendizado destas redes, o restante deste Capítulo busca definir uma RB e descrever alguns de seus algoritmos clássicos de aprendizado.

Formalmente, uma Rede Bayesiana é um grafo direcionado acíclico (*Directed Acyclic Graphic – DAG)* no qual os nós representam variáveis aleatórias com medidas de incerteza associadas. Os arcos representam a existência de uma influência direta entre as variáveis que eles conectam, e o peso destas influências é quantificado por probabilidades condicionais.

A Rede Bayesiana da Figura 2.1 mostra a relação existente entre suas variáveis ligadas por arcos orientados. A ocorrência de um arco da variável X_i para a variável X_j representa uma relação direta entre X_i e X_j. Neste caso X_i é chamado pai de X_j e consequentemente X_j é filho de X_i.

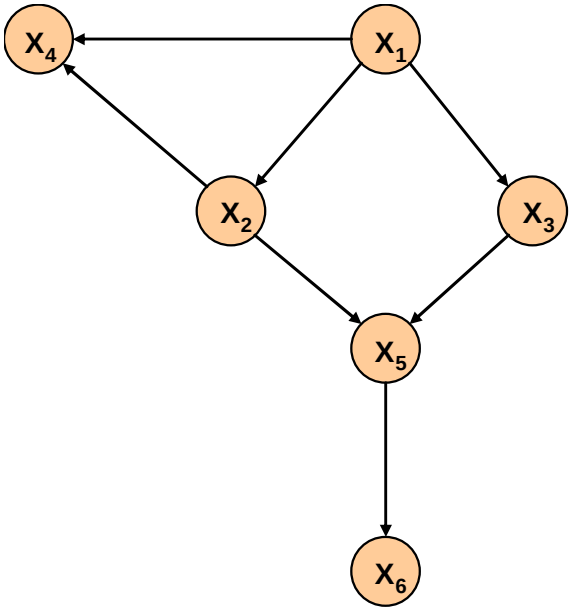


Figura 2.1. Um exemplo de Rede Bayesiana

A quantificação da relação entre as variáveis é atualizada por uma distribuição condicional, a qual condiciona a variável filha a seu(s) pai(s). Representa-se a distribuição conjunta como o produto das distribuições condicionais. Assim, para a rede da Figura 2.1 tem-se (1):

$$P(x_1, x_2, ..., x_6) = P(x_6|x_5)\, P(x_5|x_2, x_3)\, P(x_4|x_1, x_2)\, P(x_3|x_1)\, P(x_2|x_1)\, P(x_1) \qquad (1)$$

Pode-se observar que as relações de independência reduzem o esforço no cálculo da distribuição de probabilidade conjunta. A estrutura de cálculos terá uma complexidade proporcional ao número de variáveis dependentes e não ao número de variáveis do problema [5].

Partindo-se do conceito de independência, pode-se definir um conjunto de nós da Rede Bayesiana que têm influência sobre uma determinada variável do problema. Para tanto, considere uma Rede Bayesiana onde Λ_X é o conjunto de filhos do nó (variável) X e $\prod_X$ é o conjunto de pais da variável X. O conjunto de nós formado pela união dos conjuntos Λ_X e $\prod_X$ e ainda os pais das variáveis contidas em Λ_X é chamado de Markov Blanket da variável X. Um exemplo genérico pode ser visto na Figura 2.2, no qual os nós circundados representam o Markov Blanket do nó X.

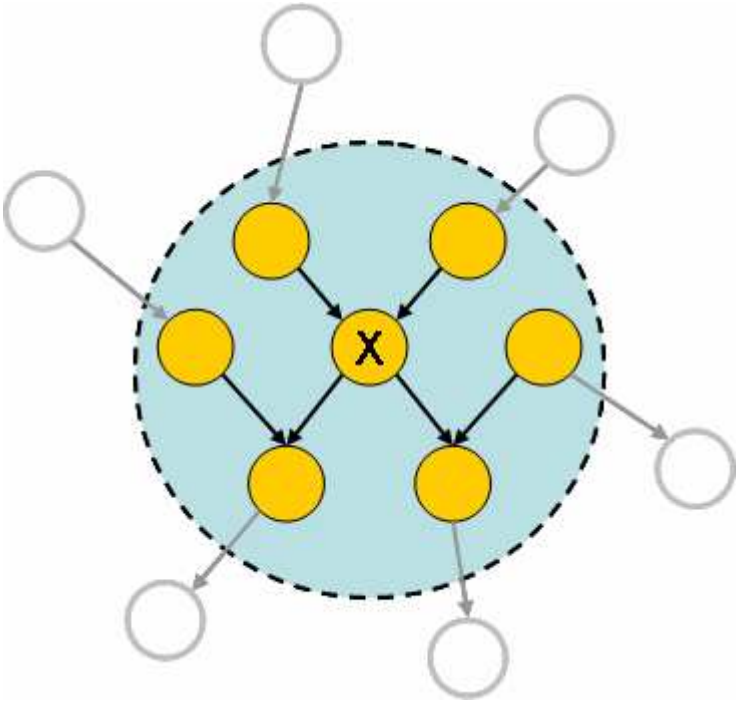


Figura 2.2. O Markov Blanket do nó X

Como pode ser visto em [4], os únicos nós da rede que influenciam o cálculo da distribuição de probabilidade condicional do nó X (dados os estados de todos os outros nós da rede) são os contidos no Markov Blanket de X. Esta propriedade pode ser muito útil no processo de seleção de atributos utilizando-se Redes Bayesianas e será mais explorada no Capítulo 3.

Tendo-se uma Rede Bayesiana definida, pode-se extrair o conhecimento nela representado através de um processo chamado inferência. De uma maneira geral, a inferência em Redes Bayesianas tem como base um processo chamado propagação de evidências. A inferência pode ser realizada apenas para se extrair um conhecimento existente na rede (a priori) ou, então, para verificar o que ocorre quando determinadas situações acontecem. Quando o objetivo é a obtenção do conhecimento a priori, basta consultar a probabilidade a priori da variável alvo. Já quando o objetivo é a obtenção de valores em uma situação específica, deve-se fornecer à rede os fatos necessários, propagar estes fatos por toda a rede e, em seguida, consultar o estado da variável alvo. A propagação dos fatos (ou propagação de evidências) é o processo responsável pela atualização do conhecimento a partir de informações fornecidas à rede.

Existem vários métodos para a realização da propagação de evidências em uma Rede Bayesiana, como os descritos em [6,4,7]. Uma vez que o objetivo deste trabalho focaliza o aprendizado de Redes Bayesianas, e não inferência, os métodos de inferência devem ser consultados nas referências citadas ou, para uma visão mais geral, em [41].

2.1 Aprendizado de Redes Bayesianas

O aprendizado de redes de conhecimento pode ser visto como um processo que gera uma representação interna (na máquina) das características que definem um dado domínio de conhecimento de modo a facilitar a recuperação de informações. Ou seja, criar-se um modelo que represente um determinado conhecimento, modelo este, no caso, uma rede de conhecimento. Uma Rede Bayesiana pode ser considerada como um tipo de uma rede de conhecimento da qual se pode extrair informações após induzida. Outro possível tipo de rede de conhecimento são as estruturas de árvores.

Em 1968, Chow e Liu [9] mostraram o primeiro resultado usando uma estrutura de aprendizado em redes de conhecimento. Foi utilizada uma estrutura de árvore, gerada a partir do método que ficou conhecido como o método de Chow e Liu, que trabalha com uma estrutura de árvore para k variáveis. Este método estima uma distribuição de probabilidade conjunta P e procura a árvore de conhecimento geradora de uma distribuição de probabilidade P' que seja a mais próxima de P.

Depois em 1987, Rebane e Pearl [10] criaram um algoritmo, para ser utilizado juntamente com o método de Chow e Liu, chamado "algoritmo de recuperação de poliárvores" destinado aos casos em que a representação gráfica da distribuição P é dada em forma de uma poliárvore. Uma poliárvore é um DAG onde um nó pode ter vários pais, porém, não existe mais de um caminho entre dois nós (veja [23]). Usando o método de Chow e Liu, é gerada a estrutura básica da árvore e em seguida, aplicando-se a esta estrutura o algoritmo de recuperação de poliárvores, obtém-se a representação gráfica da distribuição. Este algoritmo só recupera a poliárvore caso ela seja um mapeamento perfeito da distribuição a ser representada [4].

O uso de árvores, como mencionado acima, para representar conhecimento foi generalizado ao uso de grafos, como uma Rede Bayesiana, por exemplo. O processo de aprendizado em Redes Bayesianas - a partir de dados, que é o único método a ser considerado daqui em diante - deve: (1) identificar a sua estrutura, ou seja, identificar as relações de interdependência dadas pelos arcos e (2) "aprender" as distribuições de probabilidades (parâmetros numéricos) de uma rede. Normalmente o processo de aprendizado é dividido em dois subprocessos: "aprendizado da estrutura" e "aprendizado dos parâmetros numéricos" [8].

De uma maneira geral, pode-se dizer que os métodos bayesianos de aprendizado de Redes Bayesianas dividem-se em duas classes principais. A primeira é classe dos algoritmos que geram a rede através de uma busca heurística em uma base de dados, e alguns exemplos podem ser encontrados em [11 - 18], entre eles o clássico algoritmo K2 proposto por Cooper e Herskovitz [11]. Esta classe de algoritmos trata o problema de aprendizado como um problema de busca pela estrutura que melhor represente os dados. Essa estrutura vai se alterando, por adições ou remoções de arcos, usando algum tipo de busca, como uma busca heurística (por exemplo, Hill Climbing, Simulated Annealing, etc). Em seguida, os algoritmos de busca utilizam um método de pontuação para identificar se a nova estrutura é melhor do que a antiga. Este processo se repete até que a melhor estrutura seja encontrada.

Já na segunda classe estão os algoritmos que utilizam o conceito de independência condicional [4] para a construção da rede; trabalhos que aplicam esta classe de algoritmos podem ser encontrados em [19 - 26]. Esta é a classe de algoritmos utilizada neste trabalho e será mais detalhada ao longo deste Capítulo. Vale ressaltar ainda que existem propostas híbridas que combinam algoritmos das duas classes (de busca heurística e de independência condicional) [27 - 30].

2.1.1 Independência Condicional

Primeiramente será definida a notação matemática usada neste trabalho daqui em diante. A notação é apresentada na Tabela 2.1.

Tabela 2.1. Símbolos usados neste trabalho

X, Y	variáveis aleatórias
D_X, D_Y	domínio das variáveis X e Y respectivamente. São os valores que X e Y podem assumir
x,y	possíveis valores de X e Y respectivamente

Suponha duas variáveis X e Y. Elas são consideradas independentes se:

$P(x|y)=P(x)$, sempre que $P(y)>0$, $\forall x \in D_x$ e $y \in D_y$ 2)

Se X e Y são independentes, então o conhecimento do valor de Y não colabora na determinação do valor de X. Ou seja, saber Y não altera a probabilidade de X.

Outra forma de se determinar uma independência é através da medida de informação mútua:

$$I(X,Y) = \sum_x \sum_y P(x,y) \log \frac{P(x,y)}{P(x)P(y)} \quad (3)$$

É uma relação reflexiva, I(X,Y) = I(Y,X). Se I(Y,X) > 0, então X e Y são informativos um para o outro, isto é, são dependentes; caso contrário, X e Y são independentes.

Seja dado um conjunto de variáveis aleatórias Z, que não contenha X nem Y, é possível usar a medida de informação mútua condicional a fim de verificar se X e Y são condicionalmente independentes, conforme a Equação (4).

$$I(X,Y,Z) = \sum_{z}\sum_{x}\sum_{y} P(x,y,z)\log\frac{P(x,y/z)}{P(x/z)P(y/z)} \qquad (4)$$

Como no caso anterior, se I(X,Y|Z)=I(Y,X|Z)>0, então X e Y são condicionalmente dependentes, e são condicionalmente independentes em caso contrário. Em termos probabilísticos, se P(X|Y,Z)=P(X|Z), então conhecido Z, conhecer Y não informa nada para X. Isto é, X é independente de Y, dado Z. Vale lembrar que, para serem independentes, todos os valores de X, Y e Z devem satisfazer à equação P(X|Y,Z)=P(X|Z).

No ponto de vista da independência condicional, quando se deseja induzir uma rede Bayesiana a partir de uma amostra de dados, o objetivo é encontrar uma rede que melhor represente a distribuição conjunta de acordo com a amostra aleatória. Tal rede deve representar todas as assertivas de dependência e independência presentes na distribuição conjunta na amostra.

A seguir serão abordados de maneira geral os elementos necessários para o desenvolvimento e compreensão do enfoque de independência condicional. Para uma visão mais detalhada do tema consulte o trabalho [53], no qual esta Seção foi baseada.

2.1.1.1 Assertivas de dependência

Entende-se por assertiva de dependência definições acerca das relações de independência, ou dependência, entre as variáveis, considerando uma certa distribuição de probabilidade. Considere o conjunto de variáveis $X=\{X_1,X_2,...,X_n\}$, P(X) a distribuição de probabilidade conjunta de X, três subconjuntos disjuntos de X notados por A, B e C, e que *a, b* e *c* representem valores que A, B e C podem assumir respectivamente. Pode-se obter as assertivas de independência pela relação de equivalência descrita em (5):

$$(A,B \mid C)_p \Leftrightarrow P(a \mid b,c) = P(a \mid c), \forall b,c, em\ que\ P(b,c) > 0 \quad (5)$$

Ainda pode-se usar a informação mútua condicional (Equação (3)) para verificar essa relação de independência.

Uma rede Bayesiana é capaz de mostrar essas relações de independência através de sua estrutura S, a qual expressa a distribuição P(x).

Diz-se que A é independente de B dado C, $(A,B \mid C)_p$, quando C d-separa A de B no DAG S. Em $(A,B \mid C)_{p,}$ P refere-se a assertivas de dependência em relação à distribuição conjunta P.

A d-separação é discutida a seguir.

2.1.1.2 Testes de dependência

Uma Rede Bayesiana será compatível com uma distribuição de probabilidade induzida de uma amostra qualquer quando toda variável na rede for condicionalmente independente de seus não-descendentes, dado seus pais.

Essa compatibilidade entre a estrutura S da rede e a distribuição P é capaz de fazer com que S represente um conjunto de dados demonstrados em P [42]. Pode-se testar se o DAG S é compatível com distribuições P listando-se as dependências condicionais que as distribuições expressam. Pode-se obter essas dependências a partir do DAG S usando o critério gráfico “d-separação” [4], conforme será abordado a seguir.

2.1.1.3 Separação orientada

Seja G um DAG, onde V representa os nós, assume-se que $A \subseteq V$ e que V_i e V_j são nós distintos em V. Um caminho entre V_i e V_j é dito bloqueado por A se uma das seguintes condições for válida:

- Existe um nó $Z \in A$ no caminho e os arcos incidentes em Z no caminho formam uma seqüência $V_i \rightarrow Z \rightarrow V_{j.}$

- Existe um nó Z∈A no caminho e os arcos incidentes em Z no caminho formam uma seqüência V_i <– *Z* –> V_j.
- Existe um nó Z, tal que Z e, no caminho considerado, todos os descendentes de Z não estão em A, e os arcos incidentes em Z formam uma seqüência V_i –> Z <– V_j.

Considera-se V_i *d-separado* de Vj por A em G se todo caminho entre V_i e V_j for bloqueado por A.

Na Figura 2.3, X_2 e X_3 são d-separados por $\{X_1\}$. Será usada a notação $(X_2,X_3|\{X_1\})_S$ para indicar essa separação na estrutura S da rede R. Porém, X_2 e X_3 não são d-separados por $\{X_4\}$, por $\{X_5\}$ e nem pela união desses, $\{X_4,X_5\}$. De um modo geral, Z separa X de Y, $(X,Y|Z)_S$, se todos os caminhos que ligam os nós X e Y no DAG S forem d-separados por nós em Z. Na Figura 2.3, Z=$\{X_2,X_3\}$ d-separa X1 de X4, isto é $(X_1,X_4|\{X_2,X_3\})_S$. Nesse caso, todos os caminhos em S, entre X_1 e X_4, são bloqueados por nós em Z.

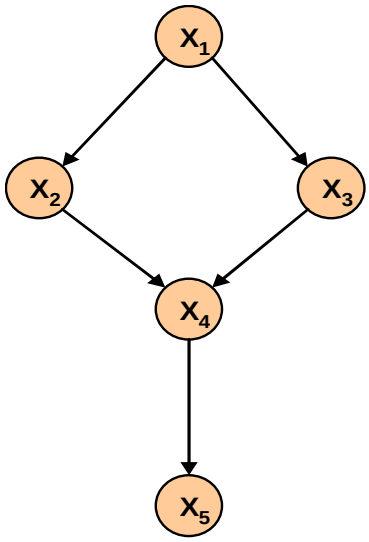


Figura 2.3. Um grafo direcionado

A d-separação é uma assertiva de independência verificada na estrutura S da rede Bayesiana. Se todos os caminhos entre X e Y são d-separados por Z na rede, então X e Y são independentes, dado Z.

2.1.1.4 Critério m-separação

Os critérios de d-separação podem ser estendidos de forma a serem aplicados em grafos ancestrais[1], e é chamado de m-separação nesse contexto. Em linhas gerais pode-se dizer que a m-separação é o equivalente à d-separação em grafos direcionados, portanto, ao efetuar a m-separação em um grafo não-direcionado tem-se um grafo direcionado contendo as d-separações. Como esse trabalho não se utiliza de grafos ancestrais, o leitor mais interessado neste assunto pode consultar [49,54].

2.1.1.5 Separador desconhecido

Os critérios de d-separação, assim como m-separação, pressupõem o conhecimento de um conjunto separador, $S_{x,y}$. Entretanto, $S_{x,y}$ não é sempre conhecido. Algumas propostas surgiram, como em [50], que descreve um algoritmo para encontrar conjuntos mínimos de d-separação e em [51], onde é apresentado um algoritmo exato, baseado em conectividade em rede [53], que é capaz de encontrar o menor subconjunto separador em grafos não orientados.

2.1.1.6 Mapeamento

A partir de um conjunto de dados, como uma amostra aleatória, deseja-se obter uma estrutura de uma rede Bayesiana que represente fielmente o conhecimento contido nesses dados. Para isso, são feitas estimativas de distribuição de probabilidade conjunta nas variáveis da amostra, de forma que a estrutura encontrada seja compatível com P. Essa compatibilidade entre uma estrutura, um DAG S, e uma distribuição de probabilidade P, é verificada caso cada valor x_i de X_i em S seja independente dos seus não descendentes, dado o conjunto de seus pais.

Afirmar que uma distribuição P é compatível com um DAG S não significa necessariamente que todas as assertivas de independência em S sejam existentes em P e vice-versa. A correspondência existente entre as independências de P e S podem ser tratadas como mapeamentos, e demonstram qual a relação de

[1] Do termo em inglês *Ancestral Graph*: um grafo com três tipos de arcos: direcionados, bidirecionados e não-direcionados. Pode ser decomposto em subgrafos de acordo com essas características dos arcos.

equivalência entre a distribuição e o DAG. Esses mapeamentos podem ser assim definidos:

a) Mapa de Dependência (D-map)

Para uma distribuição de probabilidade P, um grafo S é um mapa de dependência (D-map) de P se toda independência em P pode ser expressa em S:

$(X,Y|Z)_P \Rightarrow (X,Y|Z)_S$ (6)

Ou seja, todas as independências expressas em P estão expressas em S também.

b) Mapa de Independência (I-map)

Para uma distribuição de probabilidade P, um grafo S é um mapa de independência (I-map) de P se toda independência em S é verdadeira em P:

$(X,Y|Z)_S \Rightarrow (X,Y|Z)_P$ (7)

Ou seja, todas as independências contidas em S estão contidas em P também.

c) Mapa Perfeito (P-map)

S é um P-map de P se S for ambos, um D-map e um I-map de P, isto é,

$(X,Y|Z)_P \Leftrightarrow (X,Y|Z)_S$ (8)

Esse é o mapeamento perfeito, onde todas as relações de independência em P estão expressas em S, e todas as independências em S existem em P também.

O ideal seria sempre obter-se um mapa-perfeito, mas ainda não é possível afirmar-se com certeza que a rede encontrada contenha todas as relações causais. Existe a possibilidade de que arcos fiquem sem direcionamento, por exemplo.

2.1.1.7 Orientação e independência

Ao se determinar as relações de independência entre as variáveis ainda não é possível saber o direcionamento entre os arcos. Por exemplo: suponha-se um grafo G, onde X, Y e Z são nós de G, e sabe-se a partir da distribuição P que $(X,Y|Z)_{P.}$ Essa independência condicional não consegue determinar a orientação dos arcos entre X e Y, mas somente que eles são independentes. Na Figura 2.4 tem-se uma rede em construção e que $(X_2,X_3|\{X_1\})_{P.}$ Assim, qualquer forma de orientação entre X_1, X_2, X_3 é possível: (a) X2 ←X1→X3, (b) X2 ←X1 ←X3 (c) X2 →X1 →X3. Afinal, qualquer uma dessas orientações é compatível com a relação de independência $(X_2,X_3|\{X_1\})_{P.}$ Entretanto, se $(X,Y|Z)_P$ for falso – X e Y são dependentes, dado Z – então Z contém uma variável que é filha comum de X e Y. Aí fica completamente determinada a orientação dos arcos ao identificar qual é essa variável. Na Figura 2.4, X_2 e X_3 são dependentes dado $\{X_4,X_5\}$, particularmente X_2 e X_3 são dependentes dado $\{X_4\}$.

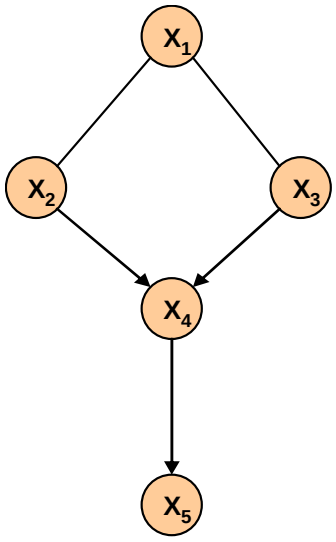


Figura 2.4. Rede Bayesiana em construção

2.1.1.8 Indução de representante

O esqueleto de um DAG qualquer é o grafo não-direcionado obtido ignorando-se a direção de cada arco. Uma estrutura-v em um DAG G é uma tripla ordenada de nós (X, Y, Z) tal que (1) G contém os arcos X -> Y e Z -> Y, e (2) X e Z não são adjacentes em G. Na Figura 2.4, os arcos e nós X_2 –> X_4 <– X_3 formam um estrutura-v.

Em [42], Pearl demonstra um teorema que garante que: dois DAGs são equivalentes se e somente se eles têm o mesmo esqueleto e as mesmas estruturas-v. Desta forma, eles também têm o mesmo conjunto de distribuições compatíveis.

O DAG da Figura 2.3 e os DAGs da Figura 2.5 são, então, equivalentes. Daí pode-se concluir que somente saber uma distribuição P não basta para se encontrar uma estrutura S compatível com todas as relações de causalidade, pois só as estruturas-v que sempre podem ser determinadas.

Portanto, pode-se dizer que ao se induzir um DAG S, nem sempre será determinada a estrutura S que corresponda à relação de causalidade real entre as variáveis do modelo. Essa limitação decorre do fato que existem vários DAGs equivalentes a uma mesma distribuição P.

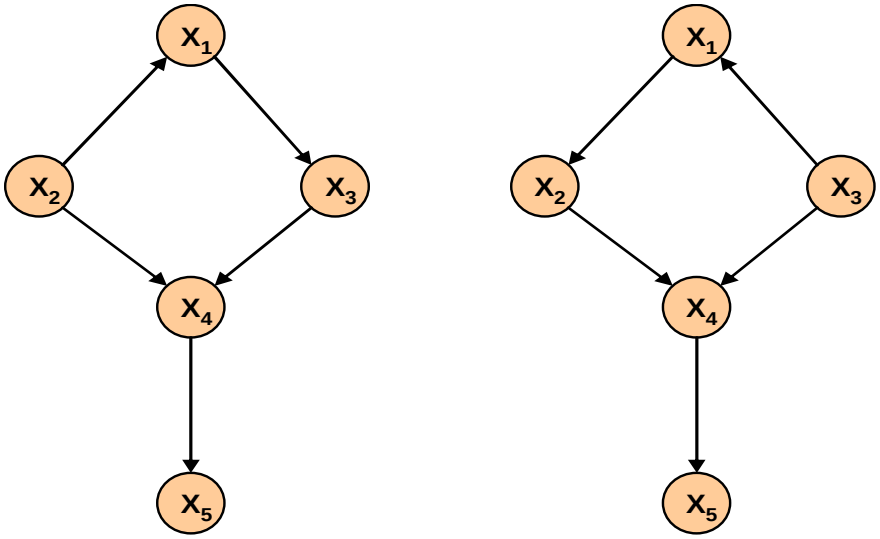


Figura 2.5. Estruturas equivalentes

2.1.2 Aprendizado por Independência Condicional

Como já citado anteriormente, existem basicamente duas classes de algoritmos de aprendizado de Redes Bayesianas a partir de dados. Uma delas agrupa algoritmos que se baseiam no uso de alguma métrica de pontuação que avalia a aptidão de uma dada rede candidata àqueles dados e um método para explorar o espaço de busca, que geralmente é o conjunto de DAGs. Os mais eficientes métodos nesse

contexto são buscas de *hill climbing*, seja determinística ou estocástica, inclusive *hill climbing* com reinícios aleatórios [32]. A outra classe de algoritmos é a de independência condicional (também chamado de *constraint-based learning*, como utilizado em [41]) conforme visto anteriormente.

Pode-se resumir então que métodos de independência condicional trabalham com testes de independência condicional em vez de funções de avaliação de modelos, como os de busca heurística. Dado um conjunto de independências condicionais em uma distribuição de probabilidade, tenta-se encontrar um DAG para o qual a condição de Markov[2] envolve todas as independências dadas somente. Para que esta classe de método alcance seu maior potencial as seguintes condições devem ser verificadas:

- é possível determinar-se, ou pelo menos estimar, as independências condicionais na distribuição de probabilidade;
- os relacionamentos de independência são perfeitamente representáveis por um DAG;
- uma extensa base de dados está disponível;
- os testes estatísticos não apresentam erros.

Algumas vantagens e desvantagens desta classe (IC) estão descritas na Tabela 2.2.

Um algoritmo é dito basear-se em independência condicional quando ele segue as idéias básicas dos três passos do Algoritmo IC (Independência Condicional) [80]. Alguns trabalhos que aplicam esta classe de algoritmos podem ser encontrados em [20 – 27,80], entre eles os clássicos algoritmos PC (Peter & Clark) e o IC. Ambos serão vistos em mais detalhes a seguir.

[2] Do termo em inglês Markov condition: qualquer nó numa RB é independente condicionalmente de seus não-descendentes, dados seus parentes. Um nó é independente condicionalmente da rede inteira dado seu Markov Blanket.

Tabela 2.2. Vantagens e desvantagens da classe IC

Vantagens	Desvantagens
Mais decisões globais: não se prende em máximos locais	Testes de independência são menos confiáveis em amostragens pequenas
Fornece exatamente as informações contidas nos dados	Uma independência incorreta pode ser longamente propagada durante a indução da rede (logo menos robusta a erros)

Tanto o PC como o algoritmo IC dependem de uma lista de independências bem consistente. Uma vez que os dados são representativos e só pode-se ter um resultado perfeito com um conjunto de dados infinitos, algumas das independências podem ser consideradas plausíveis enquanto na verdade são falsas. A questão então é como os algoritmos de IC usam a informação sobre as dependências para derivar as estruturas causais (como a estrutura de uma Rede Bayesiana) e como os erros nas informações sobre as dependências alteram os resultados desses algoritmos.

2.1.2.1 O algoritmo IC

O IC faz uma busca global exaustiva por todos os subconjuntos de variáveis para cada par de variáveis a fim de determinar quando podem ser consideradas independentes no subconjunto. Spirtes[24] comenta que este é um processo teoricamente estável, pois uma independência não identificada ainda irá, usualmente, produzir a estrutura correta porque outro conjunto de variáveis tende a implicar a mesma independência. Uma independência adicional irá resultar em um relacionamento ausente no grafo inicial.

O algoritmo descrito na Figura 2.6 foi uma proposta de Verma and Pearl em 1990. Este é o algoritmo de base para métodos de independência condicional e descreve em três passos principais como se obter um DAG parcialmente orientado a

partir de uma distribuição de probabilidade estável. Considere-se X, W e Y nós em um grafo e $S_{x,y}$ o conjunto de separação entre X e Y. Os passos são:

1. Encontrar a estrutura não direcionada: adicionar um arco (não direcionado) entre X e Y, se não há um conjunto $S_{x,y}$ tal que $(X,Y|\{S_{x,y}\})_P$
2. Determinar as estruturas-v. Para cada tripla X – W – Y, direcionar os arcos e criar uma estrutura-v X –> W <–*Y* se W $\notin$ $S_{x,y}$, ou seja, se X e Y são dependentes dado W.
3. Direcionar os demais arcos. Podem-se seguir diferentes regras a fim de se obter esse direcionamento remanescente, mas evitando-se a criação de novas estruturas-v e de ciclos.

Este algoritmo não é apresentado com preocupações de implementação, pois inclusive tem esforço computacional de ordem exponencial. Porém os demais algoritmos desta classe, independência condicional, baseiam-se nele. Na Figura 2.6 é apresentado o algoritmo completo.

ConditionalIndependence Algorithm
Problem: Given a set IND of d-separations, determine the DAG pattern faithful to IND if there is one.
Inputs: a P stable distribution (P-map); S_0 = (V,E); V={a,b,c,…} and E = {}.
Outputs: a S_0 DAG partially oriented.

1. for each pair *a* and *b* in V do
   if NOT EXISTS a separator S_{ab} such that (a,b|S_{ab})P,
   then add {*a,b*} to E;

2. for each pair *a* and *b* in V
   a and *b* not adjacent to E,
   a and *b* adjacent to c, do
      if EXISTS a S_{ab} and c $\notin$ S_{ab}, then
         Orient the edges in E: (a → c and b → c);

3. In the resulting graph, orient as many edges as possible, such that:
   i. The orientation does not create a new v-structure.
   ii. The orientation doest not create an oriented cycle.

Figura 2.6. Algoritmo IC, adaptado de [53]

De acordo com [53], a entrada P representa uma distribuição de probabilidade induzida de um representante do grafo que se pretende reconstruir ou induzida por uma amostra aleatória. Nesse último caso se pretende gerar um (grafo de saída) DAG S_0 representante de uma família de grafos compatíveis com P e que seja um P-map de P. S_0 é um grafo parcialmente orientado.

O passo 1 inclui no grafo de saída os arco não orientados {a,b} para os quais não há separador entre os nós a e b, na distribuição P, isto é, apenas os arcos cujos nós extremos a e b sejam condicionalmente dependentes. Um subconjunto de nós S_{ab} de V é um separador dos nós a e b, pela distribuição P, se a informação mútua condicional $I(a,b| S_{ab}) = 0$ (veja a Equação (4)). Esse passo, se executado tal qual é apregoado tem esforço computacional de ordem exponencial, pois S_{ab} teria de ser testado para cada elemento do conjunto das partes de V-{a,b}, nesse caso com 2n-2 subconjuntos. Uma forma de contornar o problema é começar com um grafo completo e ir tirando os arcos entre nós independentes, fazendo a escolha do separador a testar entre os vizinhos de a e b, tomando sempre o conjunto de vizinhos menor. Os vizinhos de a e b são seus separadores naturais.

O passo 2 cria as estruturas-v no DAG S_0 comuns a todos os DAGs observacionalmente equivalentes. Nesse passo o esqueleto do DAG S_0 já foi criado. Aqui o separador S_{ab}, se existir, é um dos conjuntos de vizinhos de a, de b ou um outro conjunto entre a e b que pode ser achado em tempo de ordem proporcional ao número de nós n multiplicado pelo o número de arcos do grafo S_0. Uma vez identificado um candidato ele pode ser testado usando a informação mútua condicional. Finalizado este passo, o objeto de pesquisas e propostas tem sido os passos 1, definição do esqueleto do DAG S_0, e o passo 3, orientação dos arcos que não participam de estruturas-v.

O passo 3 orienta os demais arcos de S_0. Essa orientação não pode criar outras estruturas-v e nem pode criar ciclos orientados. Em [42,43] apresentam-se propostas para o detalhamento desse passo, ambas são propostas heurísticas.

2.1.2.2 O algoritmo PC

Assume-se um conjunto de variáveis $X = \{X_1, ..., X_n\}$ com uma distribuição de probabilidade P sobre eles. Considere A representando um subconjunto

de variáveis de X. O algoritmo PC assume fidelidade (*faithfulness*). Isso significa que há um DAG G tal que as relações de independência entre as variáveis de X são exatamente aquelas representadas por G através de critérios de d-separação. O algoritmo PC é baseado na existência de um procedimento capaz de expressar quando I(A,B|C) é verificada no grafo G. O PC tenta primeiramente achar a estrutura (um grafo não-direcionado) e, em um passo posterior, faz o direcionamento dos arcos. Teoricamente, se o conjunto de independências é fiel ao grafo e há um jeito ideal de se determinar quando I(A,B|C), então o algoritmo garante a produção de um grafo equivalente, que representa o mesmo conjunto de independências, ao original.

A Figura 2.7, elaborada a partir do algoritmo PC apresentado em [17] e [60], apresenta o algoritmo PC de uma forma mais didática. O conjunto de variáveis condicionadas precisa pertencer ao conjunto de variáveis adjacentes, assim, suponha que *Adjacencies*(B_s,*A)* seja o conjunto de vértices adjacentes à *A* no grafo direcionado B_s que representa a estrutura de uma RB. Este algoritmo possui como entradas uma base de dados e um conjunto de d-separações *IND* de ordens (cardinalidades) 0, 1, 2, e subseqüentes.

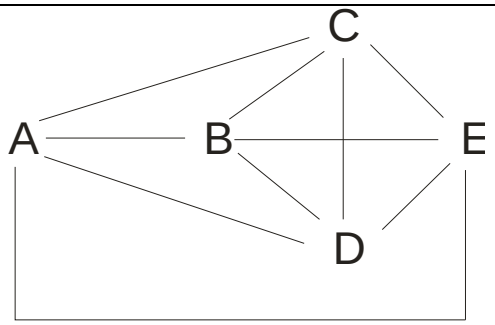
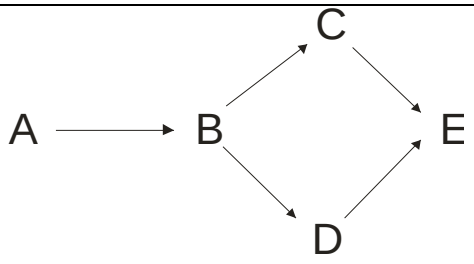
A seguir, o exemplo 2.1.2.3 exemplifica como o algoritmo define B_s. O símbolo ├ é usado para representar d-separação. Por exemplo, para n=1, **A ├ C \B,** significando que o nó A é d-separado do nó C pelo nó B.

```
PC Algorithm
Problem: Given a set IND of d-separations, determine the DAG
pattern faithful to IND if there is one.
Inputs: a set V of nodes and a set IND of d-separations among
subsets of the nodes.
Outputs: If IND admits a faithful DAG representation, the DAG
pattern gp containing the d-separations in this set.
```

```
1  begin
2  A.) form the complete undirected graph gp over V;
3  B.)
4  i = 0;      //initial cardinality
5  steps = 0; //helps to measure the algorithm effort
6  repeat
7  for (each X ∈ V) {
8     for (each Y ∈ ADJX) { //ADJX = nodes neighbors to X
9        determine if there is a set S ⊆ ADJX−{Y} such
10       that |S| = i and I({X},{Y}|S) ∈ IND;
11       steps = steps + 1;
12       if such a set S is found {
13          Add S to S_XY; // S_XY is the separator set
14          remove the edge X − Y from gp;
15       }
16    }
17 }
18 i = i + 1;
19 until (|ADJX| < i ∀X ∈ V);
20 C.)for each triple of vertices X, Y, Z such that the pair
21    X, Y and the pair Y, Z are each adjacent in C but the pair
22    X, Z are not adjacent in C, orient X–Y–Z as X->Y<-Z if and
23    only if Y is not in S_XY
24 D.) repeat
25        If A -> B, B and C are adjacent, A and C are not
26        adjacent, and there is no arrowhead at B, then
27        orient B–C as B->C.
28        If there is a directed path from A to B, and edge
29        between A and B, then orient A-B as A->B.
30     until no more edges can be oriented.
31 end.
```

Figura 2.7. Pseudocódigo do algoritmo PC, adaptado de [17,60].

2.1.2.3 Exemplo de execução do algoritmo PC

	
Grafo Completo não direcionado: Este é o passo inicial do algoritmo PC, que liga todos os nós uns aos outros.	**Grafo Verdadeiro:** Este é o grafo que deve ser a saída do algoritmo PC. Pode não ser conhecido de antemão.

i = 0 Sem independências de ordem zero (sem d-separações).

A lista de independências (que é gerada por um processo separado do algoritmo PC) não inclui independências de cardinalidade zero neste caso, ou seja, uma variável separada de outra dado um conjunto vazio. Assim, no próximo passo o grafo ainda estará completamente ligado.

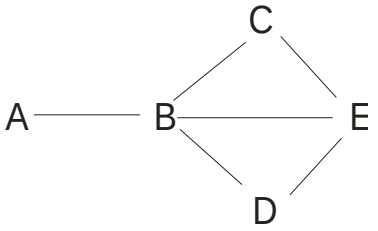
i = 1 Independências de primeira ordem São lidas as independências de cardinalidade um (variável separada de outra dada uma outra variável)	**Adjacências resultantes**
d-separações: **A ⊢ C \B (removeu o arco entre A e C)** **A ⊢ D \B (removeu o arco entre A e D)** **A ⊢ E \B (removeu o arco entre A e E)** **C ⊢ D \B (removeu o arco entre C e D)**	 Grafo resultante das independências lidas

Figura 2.8a. Exemplo de aplicação do algoritmo PC [44].

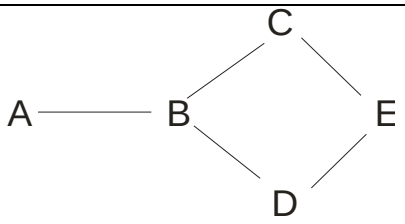
i = 2 Independência de segunda ordem	**Adjacências resultantes**
d-separações: **B ⊢ E \ {C,D}**	

Figura 2.8b. Exemplo de aplicação do algoritmo PC [44] (continuação).

Embora não esteja presente neste caso, o estágio *B)* do algoritmo pode continuar testando depois que o conjunto de adjacências do grafo verdadeiro foi identificado. O grafo não direcionado, na Figura 2.8, agora está parcialmente orientado no passo *C)*. A tripla de variáveis com apenas duas adjacências entre elas são:

$A - B - C;$ $A - B - D;$

$C - B - D;$ $B - C - E;$

$B - D - E;$ $C - E - D$

O nó *E* não está no $S_{x,y}(C, D)$, assim $C - E$ e $E - D$ colidem em E. Nenhuma das outras triplas formam colisões. O padrão final produzido pelo algoritmo é mostrado na Figura 2.9. Nota-se que alguns arcos não foram direcionados, pois o algoritmo nem sempre consegue direcionar todos os arcos. Neste caso, seria necessária outra maneira de direcioná-los, como, por exemplo, consultar um especialista, ou utilizar uma ordenação das variáveis.

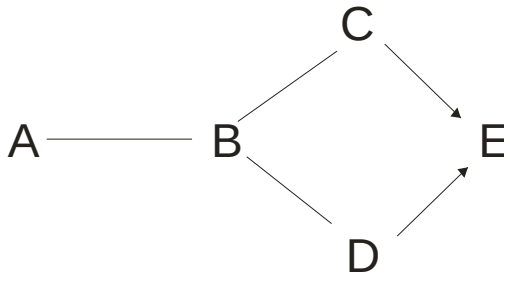


Figura 2.9. Padrão final produzido pelo PC para o exemplo anterior [44].

Conforme [4], “se a população, da qual a amostra de entrada foi retirada, se ajusta perfeitamente a um DAG C no qual todas variáveis foram estimadas, e a distribuição de população P não contém independências condicionais exceto aquelas de acordo com C, então em uma amostra suficientemente grande o algoritmo PC produz o padrão real” (contido nos dados).

2.2 Conclusões

Neste Capítulo foi abordado o aprendizado automático de Redes Bayesianas, focando principalmente no método de aprendizado de Independência Condicional. As Redes Bayesianas, por serem uma forma de modelar o conhecimento, podem ser usadas para representar o conhecimento aprendido em uma base de dados. No próximo Capítulo será exposto o processo de aprendizado em bases de dados.

3 DESCOBERTA DE CONHECIMENTO EM BANCOS DE DADOS - KDD

O processo de KDD tem como objetivo principal encontrar informações interessantes em grandes bancos de dados. Para tanto, o processo subdivide-se em algumas fases, as quais podem ser resumidas conforme a Figura 3.1.

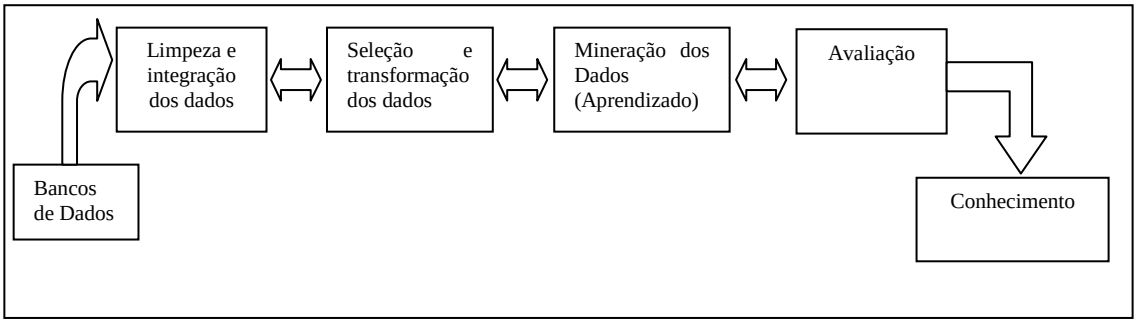


Figura 3.1. Fases do processo de extração de conhecimento de bancos de dados.

A limpeza e integração dos dados são importantes para a eliminação de "ruídos" e inconsistências presentes nos dados originais. A seleção e transformação dos dados são necessárias pelo fato de que nem todos os dados presentes em um banco de dados são informativos ou úteis em um processo de aquisição de conhecimento, e nem sempre estão em um formato adequado para os algoritmos os utilizarem. Muitas vezes é necessário selecionar os atributos relevantes para que o processo possa ter maior qualidade. A seleção de atributos é uma técnica muito importante para todo o processo de aquisição de conhecimento a partir de bancos de dados, pois, com ela pode-se selecionar atributos mais informativos, eliminar-se os irrelevantes e melhorar a qualidade dos dados fazendo assim com que o desempenho computacional dos algoritmos se torne maior. Além disso, os resultados encontrados, por serem mais simples, podem ser mais fáceis de serem compreendidos [31].

A fase de mineração de dados (Aprendizado) é essencial, pois é nesta fase que algoritmos, baseados em técnicas ditas inteligentes ou até mesmo estatísticas, são aplicados aos dados para se extrair padrões existentes. Segundo Han & Kamber [2], uma tarefa de mineração de dados pode ser, de maneira geral, preditiva ou descritiva. É importante que esta tarefa seja capaz de identificar padrões de várias granularidades e

que permita que o usuário guie o processo quando desejar. Para que a mineração de dados obtenha resultados de qualidade é necessária a identificação dos algoritmos mais adequados para a tarefa a ser realizada, pois não há um algoritmo de aprendizado considerado ótimo para qualquer tarefa de mineração de dados.

A fase de avaliação e apresentação dos dados se faz necessária pois, nem todos os resultados gerados pela fase de mineração de dados são importantes e interessantes. Por este motivo é necessário que o processo de aquisição de conhecimento seja capaz de avaliar quais resultados são relevantes e quais devem ser desprezados. Alguns padrões são utilizados para se definir o quanto um padrão é interessante e alguns critérios são considerados importantes [2] para esta definição:

- facilidade para a compreensão humana;
- grau de certeza que se tem sobre o padrão encontrado;
- facilidade ou potencial para a utilização do padrão; e
- característica inovadora, ou seja, o padrão descoberto deve revelar aspectos que ainda não haviam sido verificados.

Pode-se dizer que um padrão realmente interessante é um conhecimento importante para o usuário. E a forma como este padrão é apresentado ao usuário também é de extrema relevância para que a aquisição de conhecimento de bancos de dados seja considerada bem sucedida.

3.1 Seleção de atributos

A seleção de atributos tem se desenvolvido constantemente desde a década de 1970, principalmente nas áreas de reconhecimento de padrões, aprendizado de máquina e mineração de dados [44], tendo um papel muito importante na etapa de seleção e preparação dos dados no processo de KDD [45]. A seleção de atributos permite, por exemplo, a ordenação de atributos segundo algum critério de importância, a redução da dimensionalidade do espaço de busca de atributos e a remoção de atributos contendo ruídos ou outras características indesejadas, conforme estudado em [46].

De uma maneira geral, pode-se pensar que quanto maior a quantidade de dados disponível, mais refinado será o processo de aprendizado e, conseqüentemente,

mais precisa e representativa será a expressão obtida do conceito aprendido, mas isso nem sempre acontece. Reunanen [32] mostra que existem muitas razões para a realização da seleção de atributos em um processo de mineração de dados, dentre elas pode-se citar: (i) a manipulação de um conjunto menor de atributos é menos custosa; (ii) a qualidade do processo de aprendizado pode ser aumentada quando se trabalha apenas com o subconjunto de atributos mais relevantes de uma base de dados; (iii) o classificador gerado é mais simples e assim, tende a ser mais rápido e com menor complexidade computacional; (iv) a identificação dos atributos mais relevantes pode auxiliar no entendimento e no estudo do problema que está sendo analisado. As vantagens em se utilizar a seleção de atributos se dão, principalmente, porque a grande quantidade de atributos pode trazer informações redundantes, contraditórias ou totalmente irrelevantes.

Existem basicamente três abordagens para o problema de seleção de atributos, para o aprendizado de máquina (ou mineração de dados): *wrappers,* filtros e embutidas. Existem também diversas propostas híbridas com esses métodos [72,73,74].

3.1.1 Abordagem wrapper

A abordagem *wrapper* utiliza o resultado de algoritmos de mineração de dados (aprendizado) como critério de seleção dos atributos, ou seja, aplica-se um algoritmo de aprendizado utilizando-se todos os subconjuntos possíveis de atributos de uma base de dados. Pode-se usar, em alguns casos, alguma busca heurística de forma a não utilizar necessariamente todos os subconjuntos possíveis. Em seguida, o subconjunto que trouxer o melhor desempenho de aprendizado é definido como o melhor subconjunto de atributos. Pode-se notar que para bases de dados com um número elevado de atributos, a abordagem *wrapper* gera um custo computacional bastante elevado, pois o algoritmo de aprendizado será executado, geralmente, para todos os subconjuntos possíveis de atributos. A Figura 3.2, ilustra o funcionamento de um *wrapper*.

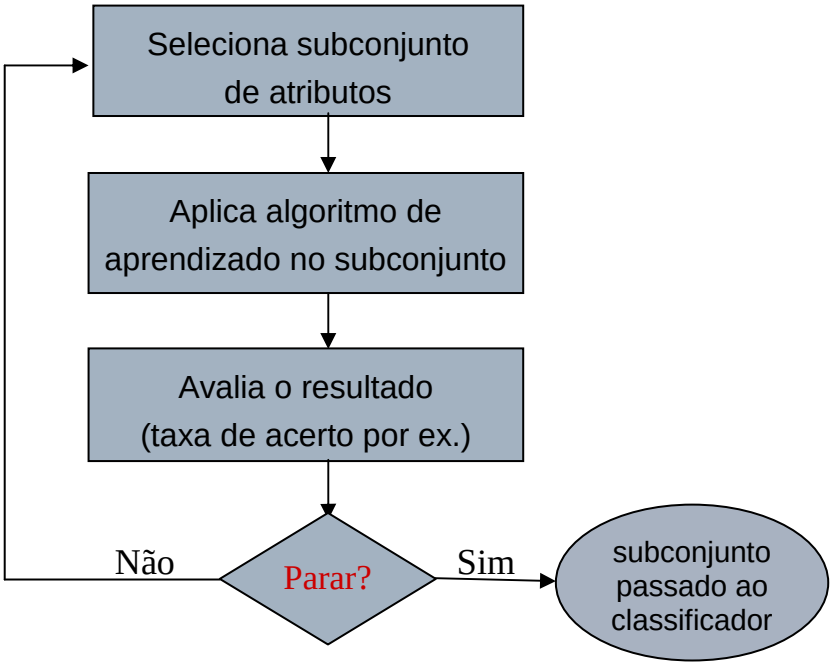


Figura 3.2. Fluxograma da abordagem wrapper

A aplicação da abordagem *wrapper* no aprendizado supervisionado de redes Bayesianas é chamada na literatura de "Redes Bayesianas Seletivas". Este método de aprendizado que integra a seleção de atributos foi descrito pela primeira vez em [33] onde se induzia uma rede Naive-Bayes (conforme descrita na Seção a seguir) em um processo *wrapper*. A idéia foi, em seguida, aplicada em Redes Bayesianas irrestritas e seus resultados comparados com algoritmos de aprendizado clássicos. A análise destes resultados pode ser encontrada em [34].

3.1.2 Abordagem filtro

A abordagem filtro, diferente dos *wrappers*, se utiliza de um processo à parte, que é executado antes da aplicação do algoritmo de aprendizado escolhido para a tarefa de mineração dos dados. Essa abordagem não se utiliza do resultado de um algoritmo de aprendizado para definir o melhor subconjunto de atributos. Os critérios de seleção utilizados nesta abordagem variam bastante e alguns de seus métodos tradicionais estão descritos em [31, 35, 36] e suas referências. A Figura 3.3, adaptada de [75], ilustra o funcionamento desta abordagem.

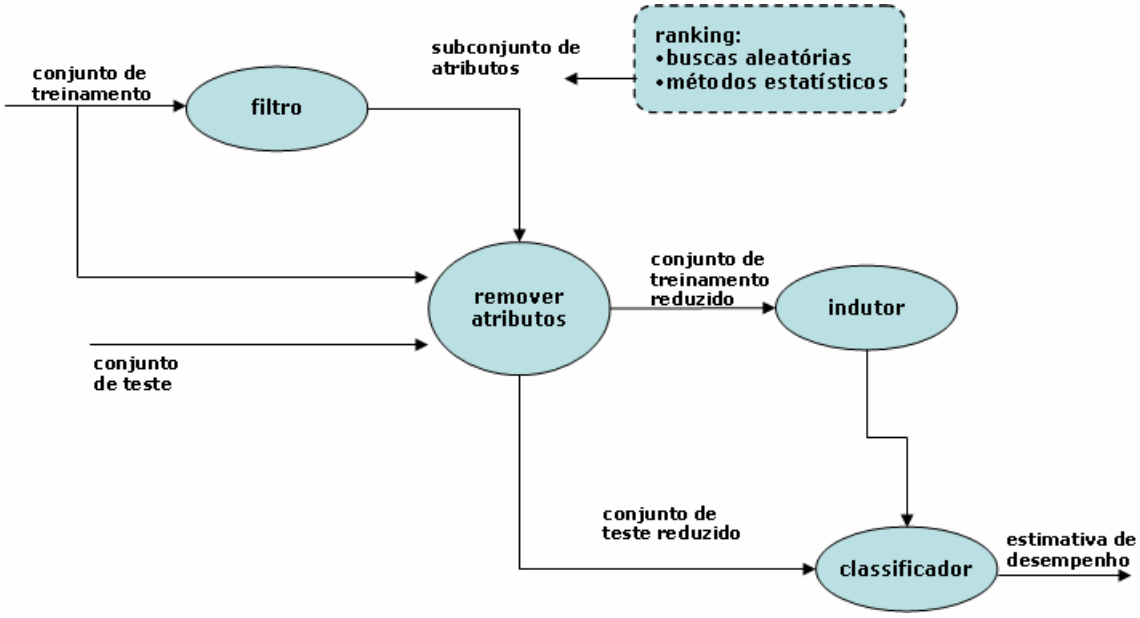


Figura 3.3. A abordagem filtro

Uma característica dos filtros, considerada ruim por alguns autores, é o fato destes métodos ignorarem inteiramente o efeito do subconjunto de atributos selecionados no desempenho do algoritmo de aprendizado [48]. As principais estratégias usadas na abordagem filtro são:

- Seleção do mínimo subconjunto de atributos: pode causar problemas quando aplicado livremente sem considerar a influência no resultado geral da aprendizagem.
- Seleção por *ranking* de atributos: qualificar uma lista de atributos, que são ordenados de acordo com métricas de avaliação como exatidão, consistência, distância, dependência. Em [47] é apresentado o uso de métodos estatísticos para a avaliação de atributos.

Resultados descritos em [17, 37] mostram que as redes Bayesianas podem ser utilizadas como filtros na seleção de atributos em tarefas de aprendizado supervisionado. Nestes trabalhos, as redes Bayesianas são utilizadas para se definir o Markov Blanket do atributo classe. Todos os atributos, que não pertencem ao Markov Blanket definido, são considerados irrelevantes para a tarefa de aprendizado. Uma das características que torna a utilização de redes Bayesianas, como filtro, uma estratégia

promissora é a robustez deste método. Em outras palavras pode-se dizer que, a seleção de atributos através de filtros Bayesianos (como descritos em [17]) tende a definir subconjuntos de atributos que trazem bons resultados em tarefas de classificação mesmo quando os algoritmos de aprendizado utilizados não são bayesianos. Como foi sugerido em [17 e 38] a "qualidade" da Rede Bayesiana influencia na "qualidade" do Markov Blanket definido pela estrutura da rede. Desta forma, obtendo-se um processo de otimização de aprendizado da Rede Bayesiana, pode-se obter também a otimização do processo de seleção de atributos (através de *Markov Blankets*).

3.1.3 Abordagem embutida

Esta abordagem é também conhecida pelo termo em inglês *embedded*, e tem esse nome porque o seletor de atributos está embutido no próprio algoritmo de aprendizado de máquina. Ou seja, a seleção dos atributos e a indução do classificador são feitas simultaneamente pelo algoritmo de aprendizado. A Figura 3.4, adaptado de [46], mostra o funcionamento desta abordagem.

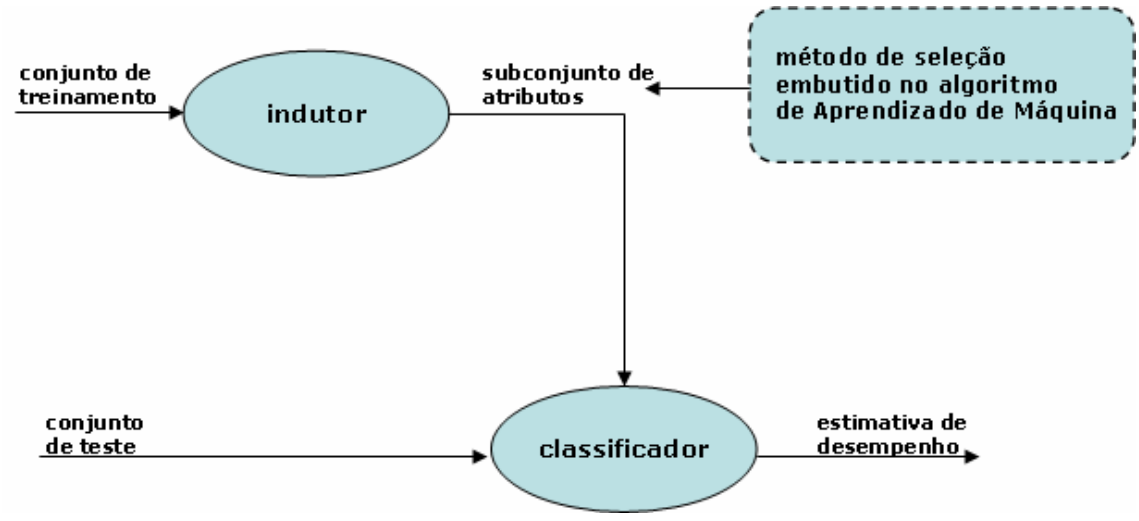


Figura 3.4. A abordagem embutida

Neste trabalho, os conceitos de um método embutido otimizado pelo uso da idéia de *Markov Blanket* são utilizados na busca da otimização da indução de um classificador Bayesiano através de algoritmos de aprendizado com base no conceito de IC. Ou seja, o algoritmo seleciona os atributos enquanto aprende a rede.

3.2 Classificação

Classificação é uma tarefa de análise de dados e reconhecimento de padrões que requer a construção de um classificador, ou seja, uma função que atribua uma *classe* a instâncias descritas por um conjunto de atributos [68]. A indução de classificadores a partir de bases de dados é um problema muito recorrente no campo do aprendizado de máquina. Existem diversas abordagens para este problema baseadas em representações como árvores de decisão, redes neurais, regras entre muitos outros. Outra abordagem é a bayesiana, usando-se Classificadores Bayesianos. Nesta abordagem pode-se usar uma Rede Bayesiana para representar o conhecimento, e é possível se determinar a classe de uma determinada instância a partir de inferências feitas tendo como alvo a variável-classe. Redes bayesianas irrestritas podem ser aprendidas como descrito na Seção 2.1 (Aprendizado de Redes Bayesianas), mas pode-se usar uma abordagem "ingênua" como um Classificador Bayesiano, que é o classificador NaiveBayes [57, 58].

O classificador NaiveBayes aprende através de dados de treinamento a probabilidade condicional de cada atributo A_i dada a classe C. A classificação então ocorre aplicando-se a regra de Bayes [4] para se calcular a probabilidade de C dado uma instância de A_1, ..., A_n, e depois encontrando-se a classe com a maior probabilidade posterior. Este método *naive* é muito eficiente computacionalmente, pois simplesmente assume que todos os atributos são condicionalmente independentes dado o valor da classe (conforme Figura 3.5) em vez de gerar um modelo baseado nas reais relações de independência entre as variáveis. Desta forma não é necessário um complexo processo de aprendizado da rede como ocorre nas redes irrestritas, e a classificação em si também é muito menos custosa. Apesar de assumir essa independência condicional irrealista, o NaiveBayes apresenta bons resultados em diversas tarefas de classificação e ainda a um baixo custo computacional. Por outro lado existem algumas desvantagens no uso do NaiveBayes, e por isso as redes irrestritas são preferíveis em determinados casos:

- Modelo não condizente com a realidade: o NaiveBayes faz uma abordagem ingênua, por isso o nome "naive", onde assume que todos os atributos são condicionalmente independentes dado a classe. Porém isso não faz sentido para o mundo real, desta forma o NaiveBayes não consegue apresentar um modelo que explique

adequadamente o conhecimento do mundo real para justificar suas classificações.

- Estimativas de probabilidade irreais: as probabilidades estimadas pelo NaiveBayes não podem ser usadas como probabilidades reais, ou seja, se o cálculo efetuado pelo NaiveBayes obtiver um valor de 65% de chance para determinada classe frente a outra com 15%, por exemplo, isso não significa que a probabilidade real daquela classe é de 65%, apenas que é maior que a de 15%, e a instância seria classificada de acordo com a classe estimada em 65%. Mais uma vez então o resultado de uma classificação NaiveBayes não tem uma explicação adequada ao usuário, o que é uma desvantagem grande, pois em diversos domínios é muito importante que o resultado de um classificador ofereça um embasamento ao usuário de forma que aumente sua confiança para a tomada de decisão.
- Taxas de classificação: apesar de em muitos domínios reais o NaiveBayes apresentar um bom desempenho, o uso das corretas independências condicionais, mais alinhadas ao mundo real, pode trazer melhores taxas de classificação, mesmo que seja a um custo computacional mais elevado.

Este trabalho tem como objetivo a otimização do processo de indução de uma Rede Bayesiana irrestrita para problemas de classificação, o que pode trazer ganhos de desempenho. Pode-se dizer então que se tende a obter um desempenho melhor para classificação, até mesmo melhor que um NaiveBayes, e a um custo computacional menor que os tradicionais, mesmo que ainda maior que um NaiveBayes, mas sem as desvantagens descritas. Uma extensa análise comparativa do NaiveBayes com outros classificadores é apresentada em [67].

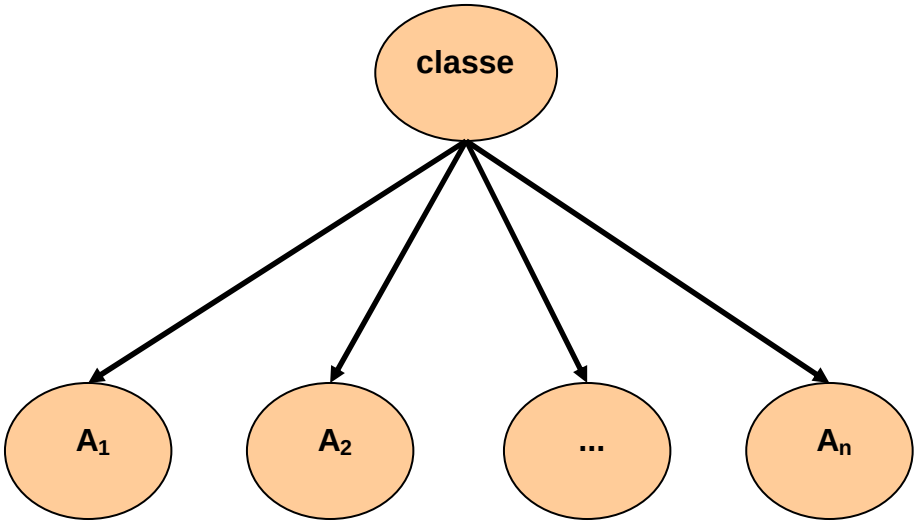


Figura 3.5. Classificador NaiveBayes

3.3 Conclusões

Neste Capítulo foi abordado o que é e como se estrutura a descoberta de conhecimento em bases de dados. Foi dada maior ênfase nas fases de seleção de atributos e classificação uma vez que são as etapas onde o método proposto neste trabalho se contextualiza. Este método, que visa otimizar a seleção e classificação através da abordagem embutida (Seção 3.1.3), é descrito no próximo Capítulo.

4 O ALGORITMO MARKOVPC

O MarkovPC pode ser visto como uma extensão do algoritmo PC tradicional. Seu intuito é explorar o Markov Blanket da Classe (MBC) tentando construir um classificador mais preciso e mais simples. A idéia principal é excluir as possíveis estruturas que contiverem atributos fora do MBC. Ou seja, durante a indução da rede só serão considerados os atributos que possam estar no MBC.

O uso do MBC permite a redução do esforço computacional necessário para construir a estrutura (DAG) do classificador e também favorece a simplificação dessa estrutura. Desta forma, o MarkovPC pode minimizar o esforço requerido tanto na construção do classificador quanto na classificação, pois um número menor de atributos será considerado. Além disso, um classificador menos complexo, como proposto pelo MarkovPC, tem outras vantagens:

- Facilita a visualização do modelo bem como o seu entendimento: um classificador mais simples e menor, com menos atributos, é mais fácil de ser analisado;
- Reduz requisitos de medição e armazenamento: um classificador mais complexo exige que mais atributos sejam considerados, e os valores desses atributos ocupam maior espaço de armazenamento. Além disso, é exigido um maior esforço para a obtenção e medida desses atributos excedentes. Um classificador mais simples diminui os atributos necessários e consequentemente reduz os requisitos para medi-los e para armazená-los.
- Redução de tempo de treinamento e de classificação: o esforço para se induzir o classificador a partir de dados (treinamento) será menor, com menos atributos sendo considerados. O tempo de classificação também tende a diminuir, pois o processamento de um número menor de variáveis é mais rápido.

- Redução da maldição da dimensionalidade[3]: ao se estimar um modelo a partir de dados a complexidade aumenta exponencialmente com a dimensão (a faixa de atributos). Eliminando-se os atributos irrelevantes, diminui-se essa complexidade, beneficiando o uso de recursos computacionais e melhorando-se o desempenho da predição.

O algoritmo MarkovPC foi projetado de forma a obter essas vantagens citadas. Na Figura 4.1 é apresentado o algoritmo MarkovPC baseado no PC tradicional descrito anteriormente.

Observando-se as Figuras 2.7 (algoritmo PC) e 4.1 (MarkovPC) é possível verificar que a principal diferença entre o PC e o MarkovPC estão nas linhas 8 e 9 da Figura 4.1. Essas linhas definem o teste de MBC. Em outras palavras, o procedimento UNDIRECTED_MB(CLASS) representa um conjunto de nós que podem estar contidos no MBC. É importante ressaltar que o MBC criado pelo MarkovPC é uma aproximação do MBC identificado pelo PC. Isso acontece porque neste ponto (linhas 8 e 9 do algoritmo na Figura 4.1) o grafo é não-orientado, então o MBC exato não pode ser definido ainda. Neste sentido, os nós retornados por UNDIRECTED_MB(CLASS) é independente de orientação de grafo, o que significa que este conjunto contém todos os nós que podem ser alcançados a partir da classe em até no máximo dois arcos. Esses nós são possíveis candidatos de serem membros do MBC quando o grafo estiver direcionado (passos C e D da Figura 4.1). Seguindo esta estratégia, todos os nós identificados pelo algoritmo PC como parte do MBC estarão também presentes no MBC definido pelo MarkovPC. Alguns nós presentes no MBC definidos pelo MarkovPC, entretanto, podem não estar presentes no MBC definido pelo algoritmo PC.

O MBC induzido pela estratégia aplicada pelo MarkovPC pode ser sumarizado da seguinte forma: considerando-se uma lista consistente de independências (IND), o MBC induzido pelo MarkovPC não é o MBC mínimo, mas contém pelo menos todos os nós relevantes (presentes no MBC induzido pelo PC tradicional) à variável classe.

[3] Do termo em inglês “curse of dimensionality”, apresentado inicialmente em [61].

```
MarkovPC Algorithm
Problem: Given a set IND of d-separations, determine the DAG
         pattern, optimized as a classifier and faithful to
         IND if there is one.
Inputs:  a set V of nodes, a set IND of d-separations among
         subsets of the nodes and a CLASS class-node ∈ V.
Outputs: If IND admits a faithful DAG representation, the DAG
         pattern gp containing the d-separations in this set
         optimized as a Bayesian classifier.
```

```
1  begin
2  A.) form the complete undirected graph gp over V;
3  B.)
4  i = 0;
5  steps = 0;      //helps to measure the algorithm effort
6  repeat
7  for (each X ∈ V) {    //first X should be the CLASS node
8     if UNDIRECTED_MB(CLASS) ⊄ X {
9        remove X from gp;
10    }
11    else {
12       for (each Y ∈ ADJ_X) {
13          determine if there is a set S ⊆ ADJX−{Y} such
14          that |S| = i and I({X},{Y}|S) ∈ IND;
15          steps = steps + 1;
16          if such a set S is found {
17             Add S to S_XY;
18             remove the edge X − Y from gp;
19          }
20       }
21    }
22    steps = steps + 1;
23 }
24 i = i + 1;
25 until (|ADJX| < i for all X ∈ V OR i<=|S|);
26 C.)for each triple of vertices X, Y, Z such that the pair
27    X, Y and the pair Y, Z are each adjacent in C but the pair
28    X, Z are not adjacent in C, orient X – Y – Z as X -> Y <- Z
29    if and only if Y is not in S_XY
30 D.)repeat
31       If A -> B, B and C are adjacent, A and C are not
32       adjacent, and there is no arrowhead at B, then
33       orient B–C as B->C.
34       If there is a directed path from A to B, and edge
35       between A and B, then orient A-B as A->B.
36    until no more edges can be oriented.
37 end.
```

Figura 4.1. O Algoritmo MarkovPC

Quanto à complexidade do classificador gerado, uma vez que o MarkovPC seleciona as variáveis mais relevantes, tende-se a induzir classificadores contendo um número reduzido de variáveis. É importante notar que a seleção de variáveis é baseada em um MBC aproximado, então o número de variáveis selecionadas

depende do tamanho do MBC. Considerando-se as características descritas no algoritmo MarkovPC, por um lado é possível dizer que, tratando-se de domínios com alto número de atributos e um MBC pequeno (poucas variáveis no MBC), o esforço do MarkovPC tende a ser menor do que o do PC tradicional. Por outro lado, em uma situação onde o domínio possui um MBC contendo um alto número de variáveis, o MarkovPC tende a não eliminar variáveis, e, assim, seu esforço pode ser maior que o requerido pelo PC tradicional.

Considerando que testar se uma variável está presente no MBC tem a mesma complexidade presente na linha 10 do algoritmo PC (Figura 2.7), é possível analisar a situação descrita acima como um caso extremo, no qual um domínio com N variáveis (todas presentes no MBC) e uma lista de independências (IND) tendo cardinalidades de 0 até M (consulte [60] para uma descrição mais detalhada sobre cardinalidade de independências) são dadas. Em uma situação assim, o MarkovPC precisará de $(N * (M+1))$ testes a mais que o PC tradicional. Isso acontece porque, para cada conjunto de independência de uma cardinalidade, o MarkovPC testará se cada variável está presente no MBC ou não. Na maioria dos domínios, entretanto, essa situação extrema não acontece, assim, o MarkovPC tende a necessitar de menos esforço do que o PC tradicional.

Outra diferença entre os algoritmos nas Figuras 2.7 e 4.1 está presente na cláusula *"until"* (linha 19 da Figura 2.7 e linha 25 da Figura 4.1). Essa cláusula no MarkovPC possui um operador "OR" que não está presente no PC original. A motivação de se inserir este operador "or" é de se eliminar testes desnecessários. Essa modificação não é necessária à estratégia baseada no MBC proposta pelo MarkovPC, mas foi implementada a fim de se reduzir a complexidade computacional do algoritmo. É importante notar que essa modificação (inserção do operador "OR") não altera o comportamento do algoritmo no sentido do classificador sendo induzido.

Levando-se em conta a taxa média de acerto de classificação (TMAC), o algoritmo MarkovPC deve produzir resultados consistentes aqueles produzidos pelo PC tradicional.

Foram utilizados testes empíricos para a avaliação do desempenho do MarkovPC. Os experimentos conduzidos se utilizaram de 10 bases de dados ao total:

- 4 bases de dados geradas a partir de Redes Bayesianas: a rede clássica ALARM, e mais 3 bases geradas manualmente batizadas de synthetic1, synthetic2 e synthetic3. foram gerados dados a partir dessas redes usando o software Genie [55].
- 6 bases reais extraídas do repositório UCI [62]: car, kr-vs-kp, lung-cancer, patient (postoperative-patient-data), solar-flare 1, solar-flare 2. A descrição mais completa dessas bases pode ser obtida na internet no repositório da UCI.

Os dados então foram usados como conjunto de treinamento e teste, através de validação cruzada (10-fold), para as redes induzidas pelos algoritmos PC e MarkovPC. Pode-se então resumir a metodologia dos testes nos passos definidos na Figura 4.2.

1. Geração de uma base de dados a partir de uma rede Bayesiana original conhecida ou utilização dos dados originais nas bases reais;

2. Para efeito de comparação visual das estruturas de redes geradas por cada um dos algoritmos com as induzidas pelos algoritmos, indução de uma rede Bayesiana utilizando-se todo o conjunto de dados, gerado no passo 1, utilizando-se o algoritmo PC tradicional e o MarkovPC proposto;

3. Para efeito de comparação do poder preditivo de cada rede gerada, através de uma estratégia de validação cruzadas (com 10 partes), indução de redes Bayesianas em conjuntos de dados de treinamento e classificação em conjuntos de teste.

Figura 4.2. Metodologia dos testes

Deve-se ressaltar que o objetivo do algoritmo MarkovPC não é o de identificar o Markov Blanket exato de uma Rede Bayesiana, portanto, uma rede gerada pelo MarkovPC pode ser diferente tanto do Markov Blanket da variável classe da rede gerada pelo PC tradicional, quanto do MB definido na rede original. No entanto, há uma

tendência de que a rede do MarkovPC seja semelhante ao Markov Blanket da variável classe na rede do PC tradicional e na rede original.

As próximas subseções analisam os resultados gerados pelo MarkovPC nos dez domínios considerados.

4.1 Rede ALARM

Esta é uma rede real para monitoração de pacientes em tratamento intensivo conhecida na comunidade dedicada ao estudo de redes Bayesianas. Foi obtida a partir de conhecimento de um especialista e foi descrita originalmente em [56]. É formada por 37 variáveis com dois, três ou quatro possíveis valores e 46 arcos. A rede representa 8 variáveis de diagnóstico, 16 sintomas, e 13 variáveis intermediárias.

Uma vez que o trabalho proposto visa a redução de atributos para problemas de classificação, a Rede Bayesiana de ALARM foi utilizada como um Classificador Bayesiano. A Figura 4.3 mostra a estrutura da rede ALARM.

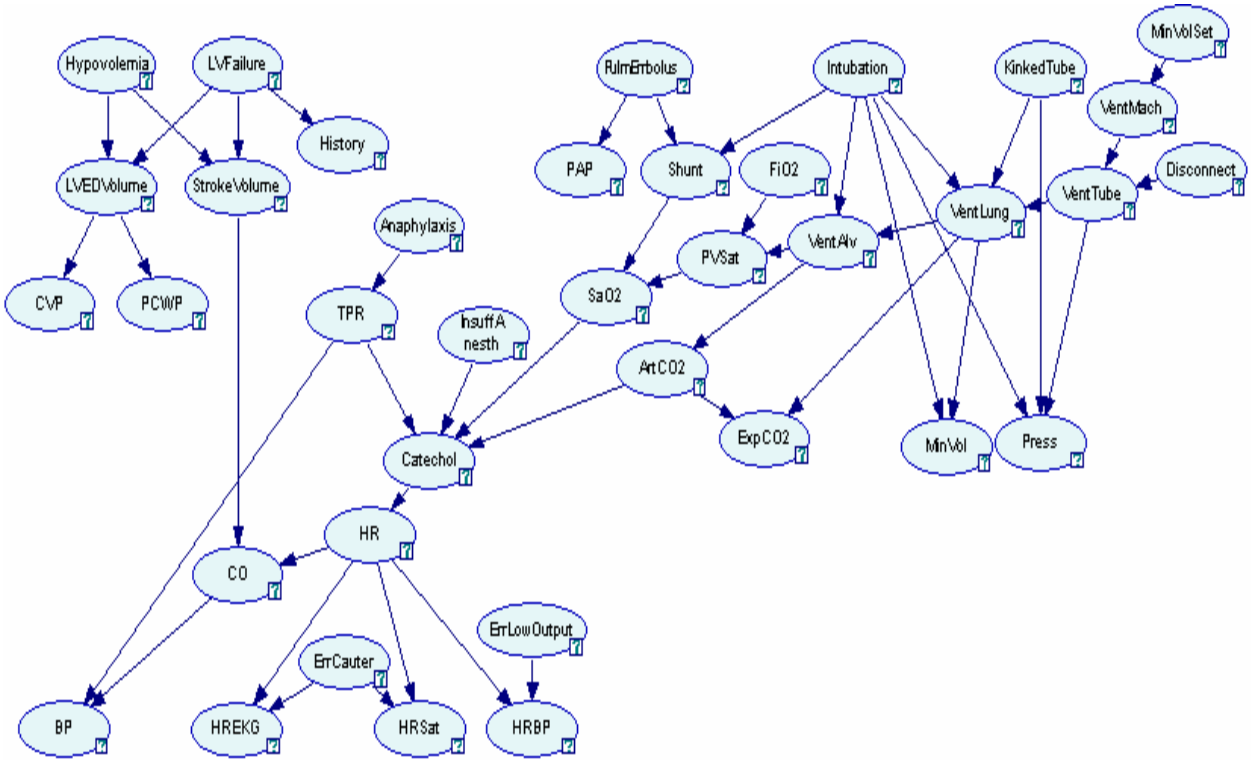


Figura 4.3. Rede Bayesiana original ALARM [56]

Segundo [56], pode-se utilizar como classes as variáveis que expressam diagnósticos, a saber: Hypovolemia, LVFailure, Anaphylaxis, InsuffAnesth, PulmEmboulus, Intubation, KinkedTube e Disconnect.

4.1.1 Descrição dos experimentos

No primeiro passo dos testes, foi gerada uma base de dados com 100.000 registros a partir da rede ALARM original e os passos 2 e 3 (definidos na Figura 4.2) foram executados. A Figura 4.4 representa a rede gerada pelo algoritmo PC. Já as Figuras numeradas de 4.5 a 4.32 representam as redes geradas pelo algoritmo MarkovPC para a rede ALARM tendo como classe as variáveis Hypovolemia, LVFailure, Anaphylaxis, InsuffAnesth, PulmEmbolus, Intubation, KinkedTube e Disconnect, respectivamente. Vale ressaltar que, como o algoritmo PC não distingue a variável classe das outras variáveis do problema, apenas uma rede foi gerada pelo PC para todas as possíveis variáveis classe.

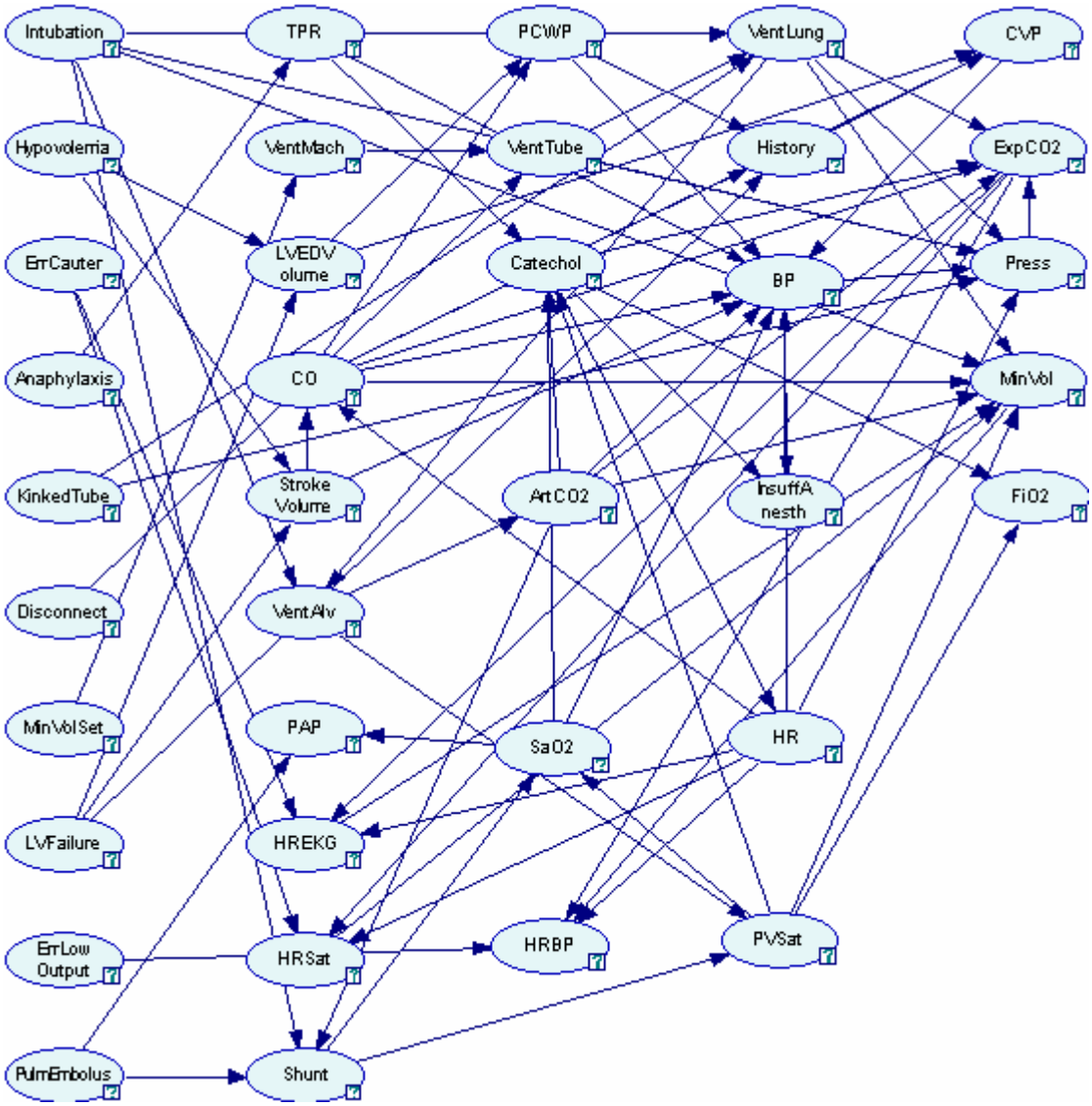


Figura 4.4. Rede ALARM gerada pelo PC tradicional

4.1.2 Análises dos experimentos

A seguir serão mostradas as redes geradas pelo MarkovPC para cada variável classe e, na seqüência. feita uma breve análise.

Hypovolemia

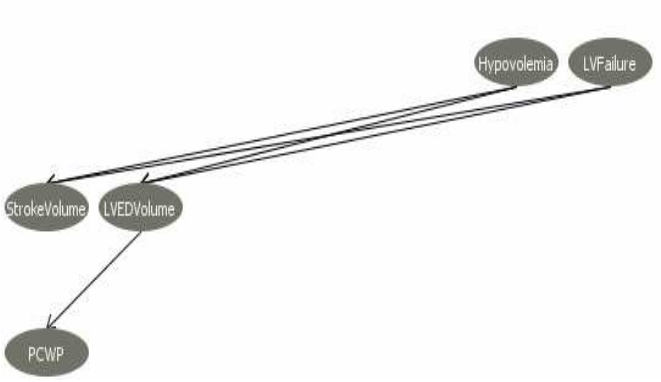


Figura 4.5. Hypovolemia gerada por MarkovPC

Analisando-se a variável Hypovolemia na rede ALARM original (Figura 4.3) é possível identificar que fazem parte do seu MB as variáveis LVFailure, StrokeVolume e LVEDVolume. Observando-se a rede gerada pelo MarkovPC (Figura 4.5) nota-se que todas as variáveis do MB original estão presentes no MB da variável Hypovolemia. Desta forma, pode-se dizer que a estrutura gerada pelo MarkovPC é consistente.

Anaphylaxis

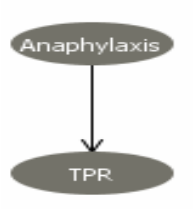


Figura 4.6. Anaphylaxis gerada por MarkovPC

Comparando-se o MB original definido para a variável Anaphylaxis (Figura 4.1)

com o obtido através do MarkovPC (Figura 4.6) é possível notar que são idênticos. Isto confirma a consistência do MarkovPC para esta variável também.

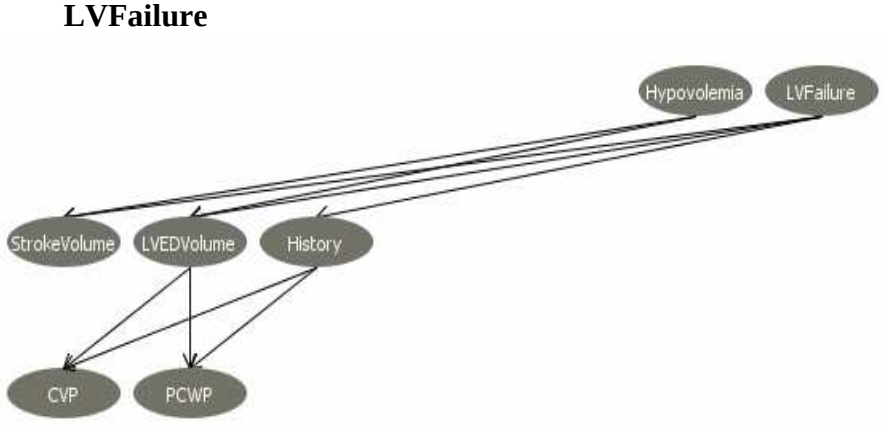


Figura 4.7. LVFailure gerada por MarkovPC

Para a variável LVFailure, o MB gerado pelo MarkovPC (Figura 4.7) é idêntico ao descrito na rede original. Mas a rede induzida pelo MarkovPC possui duas variáveis que não estão no MB e isto ilustra a situação na qual o MarkovPC trabalha com um MB estendido para selecionar as variáveis que devem estar presentes na rede induzida. Vale relembrar que a presença destas duas variáveis (CVP e PCWP) não deve influenciar a qualidade do classificador representado pela rede e por isso elas não precisariam estar presentes. Mas como o MarkovPC identifica as variáveis relevantes antes de direcionar os arcos, tanto a variável CVP quanto a PCWP poderiam fazer parte do MB e por isto foram consideradas.

InsuffAnesth

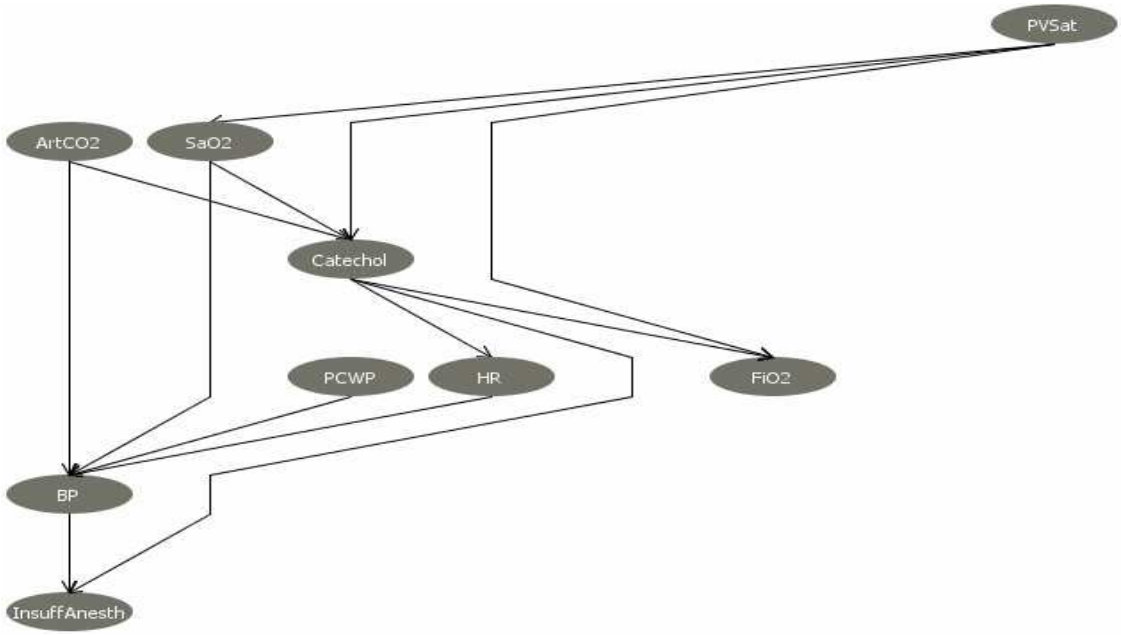


Figura 4.8. Insufficient Anesthesia (InsuffAnesth) gerada por MarkovPC

A rede gerada pelo MarkovPC quando a variável InsuffAnesth foi considerada como classe (Figura 4.8) evidenciou um MB consideravelmente diferente daquele definido na rede original. Mas está muito próximo do MB gerado pelo PC, pois tanto na rede gerada pelo PC como pelo MarkovPC o MB de InsuffAnesth é formado por BP e Catechol. Isto mostra que as independências condicionais não foram bem capturadas para esta variável e, por isso, tanto o PC quanto o MarkovPC podem não apresentar um bom desempenho na classificação desta variável. Este baixo desempenho, comparativamente aos demais, foi comprovado nos experimentos e pode ser observado na Tabela 4.1.

PulmEmbolus

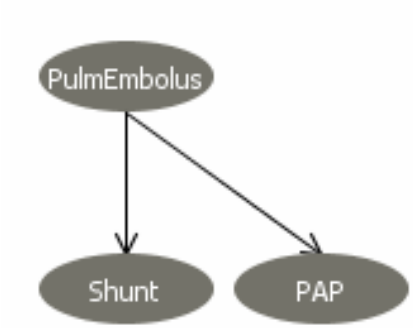


Figura 4.9. PulmEmboulus gerada por MarkovPC

Quando se analisa a Figura 4.9 que mostra a rede gerada pelo MarkovPC quando a variável PumbEmbolus é considerada como a classe, percebe-se que o MB identificado é muito próximo do original, faltando somente a variável Intubation. O MB de PulmEmbolus na rede do PC tradicional inclui 7 variáveis, o que difere bastante do MB original. O MB da rede gerada pelo MarkovPC conseguiu se assemelhar mais do da rede original e obteve uma taxa de classificação levemente melhor, conforme a Tabela 4.1.

Intubation

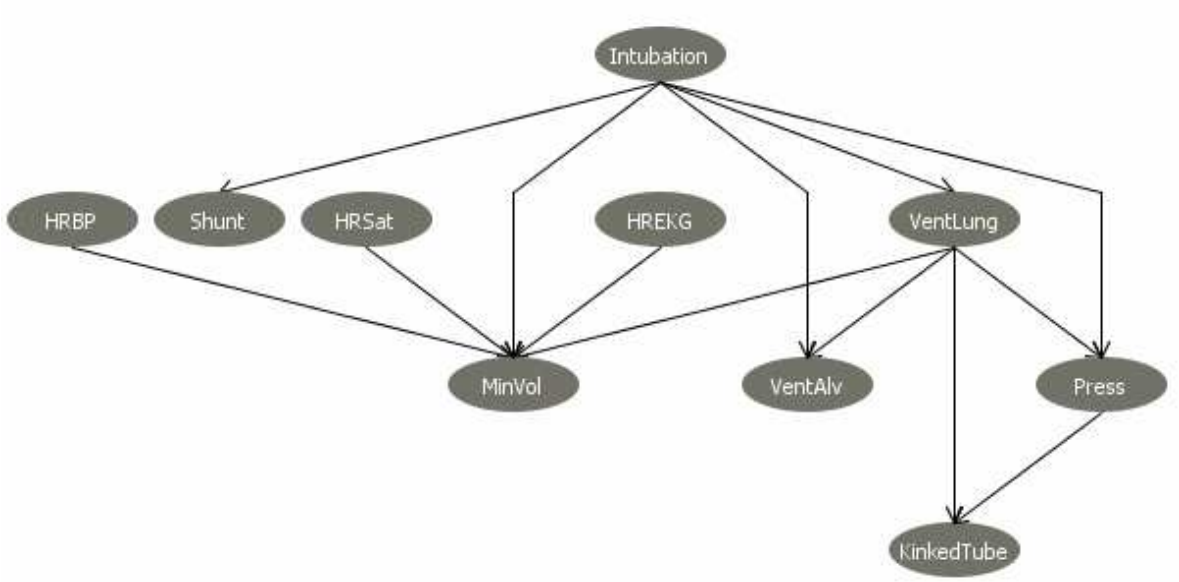


Figura 4.10. Intubation gerada por MarkovPC

Os MBs da variável Intubation são extensos, por isso serão identificados primeiro o MB de cada rede gerada a fim de se facilitar a compreensão:

- MB na rede original da variável Intubation: Shunt, VentAlv, MinVol, VentLung, Press, PulmEmbolus, KinkedTube, VentTube
- MB na rede gerada pelo PC Tradicional da variável Intubation: VentLung, Press, MinVol, VentAlv, Shunt, KinkedTube,VentTube, BP, HR, ArtCO2, CO, HREKG, HRSat,PVSat,PulmEmbolus
- MB na rede gerada pelo MarkovPC na variável Intubation: Shunt, MinVol, VentAlv, VentLung, Press, HRBP, HRSat, HREKG, VentLung,

O MB de Intubation na rede gerada pelo PC inclui todas as variáveis do MB Original, mas adiciona mais variáveis irrelevantes, as quais acabam piorando um pouco a taxa de classificação do PC em relação a taxa do MarkovPC. Este por sua vez não gera um MB que inclui todas as variáveis originais e também adiciona outras não presentes no MB original, mas tem uma taxa de classificação bastante apurada (Tabela 4.1) e gerando uma rede menor.

KinkedTube

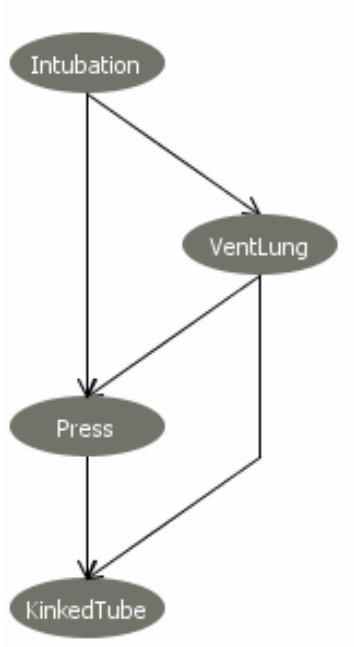


Figura 4.11. KinkedTube gerada por MarkovPC

O MarkovPC conseguiu gerar uma rede com um MB mais compatível com o MB original do que o PC tradicional, o qual induziu uma rede contendo muitas variáveis a mais no MBC comparado ao MBC original. O MarkovPC gerou uma rede menor com praticamente a mesma taxa de classificação (Tabela 4.1) que o PC.

Disconnect

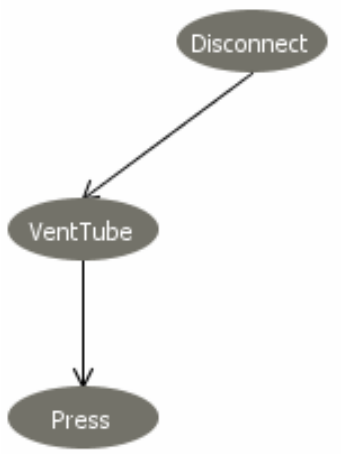


Figura 4.12. Disconnect gerada por MarkovPC

O PC conseguiu gerar uma rede com um MB consistente com o MB original, enquanto o MarkovPC não incluiu uma das variáveis (VentMach). No entanto, a taxa de classificação do MarkovPC foi um pouco superior (Tabela 4.1), indicando que os atributos foram bem selecionados.

Tabela 4.1. Resultados de experimentos na rede ALARM

REDE ALARM variável classe	ESFORÇO (número de passos)		TAXA CLASSIFICAÇÃO		|MB|
	PC	MarkovPC	PC	MarkovPC	
Hypovolemia	2758	1020	98,421	98,421	3
VFailure	2758	1060	99,031	99,043	4
Anaphylaxis	2758	978	98,989	98,989	1
InsuffAnesth	2758	1435	82,476	84,795	4
PulmEmboulus	2758	1049	99,217	99,427	3
Intubation	2758	1434	97,785	98,481	8
KinkedTube	2758	1226	98,884	98,927	4
Disconnect (24)	2758	1034	97,857	98,749	2

Nota-se que o MarkovPC conseguiu taxas de classificação muito semelhantes ao PC tradicional, e onde houve alguma diferença ela foi positiva para o MarkovPC. O ganho de esforço computacional foi em média de 58%, ou seja, o MarkovPC consegue resultados tão bons quanto, ou até melhor, que o PC tradicional efetuando, em média, somente 42% do trabalho. Uma análise geral mais detalhada é apresentada ao final deste Capítulo.

4.1.3 Resultados de experimentos na rede ALARM para diferentes conjuntos de dados

Para a realização dos experimentos apresentados na Seção 4.1.2 foi gerada uma base de dados com 100.000 registros. A fim de se validar a robustez do MarkovPC em bases cuja amostra não é tão extensa, foram realizados experimentos em bases com diversas quantidades de instâncias seguindo a mesma metodologia descrita anteriormente. Foram, então, geradas bases com 10.000, 1.000, 100 e 50 instâncias em cada uma.

De acordo com os resultados mostrados nas Tabelas Tabela 4.2 até Tabela 4.5, pode-se concluir de maneira geral que o MarkovPC não sofre perda de

desempenho tanto em taxa de classificação quanto em redução de esforço computacional. Portanto, a mesma análise feita com os resultados de experimentos com 100.000 registros é válida mesmo para a base ALARM com menor quantidade de instâncias.

Tabela 4.2. Resultados da ALARM com 10.000 registros

REDE ALARM Variável classe	ESFORÇO		TAXA CLASSIFICAÇÃO			|MB|
	PC	MarkovPC	PC	MarkovPC	Naive Bayes	
Hypovolemia	2430	1147	98,430	98,400	96,340	3
LVFailure	2430	1161	99,090	99,060	96,750	4
Anaphylaxis	2430	1098	99,110	99,110	96,260	1
InsuffAnesth	2430	1421	81,490	85,960	63,930	4
PulmEmboulus	2430	1207	99,230	99,520	96,810	3
Intubation	2430	1415	97,020	97,250	84,840	8
KinkedTube	2430	1358	99,130	99,130	85,100	4
Disconnect	2430	1156	98,950	98,950	93,510	2
Média	**2430**	**1245**	**96,556**	**97,173**	**89,193**	

Tabela 4.3. Resultados da ALARM com 1.000 registros

REDE ALARM Variável classe	ESFORÇO		TAXA CLASSIFICAÇÃO			|MB|
	PC	MarkovPC	PC	MarkovPC	Naive Bayes	
Hypovolemia	2295	1106	98,200	98,100	96,100	3
LVFailure	2295	915	98,800	98,900	96,600	4
Anaphylaxis	2295	1102	99,400	99,400	94,800	1
InsuffAnesth	2295	1270	86,900	88,000	63,100	4
PulmEmboulus	2295	1331	99,500	99,800	96,600	3
Intubation	2295	1340	97,000	95,900	84,500	8
KinkedTube	2295	1313	99,300	99,300	86,100	4
Disconnect	2295	1123	98,900	98,900	90,500	2
Média	**2295**	**1188**	**97,250**	**97,288**	**88,538**	

Tabela 4.4. Resultados da ALARM com 100 registros

REDE ALARM Variável classe	ESFORÇO		TAXA CLASSIFICAÇÃO			|MB|
	PC	MarkovPC	PC	MarkovPC	Naive Bayes	
Hypovolemia	2067	1037	99,000	98,000	97,000	3
LVFailure	2067	1166	98,000	98,000	92,000	4
Anaphylaxis	2067	1151	96,000	96,000	89,000	1
InsuffAnesth	2067	1180	83,000	81,000	70,000	4
PulmEmboulus	2067	565	100,000	100,000	99,000	3
Intubation	2067	1060	92,000	90,000	86,000	8
KinkedTube	2067	1165	98,000	95,000	87,000	4
Disconnect	2067	1101	100,000	100,000	90,000	2
Média	**2067**	**1053**	**95,750**	**94,750**	**88,750**	

Tabela 4.5. Resultados da ALARM com 50 registros

REDE ALARM Variável classe	ESFORÇO		TAXA CLASSIFICAÇÃO			|MB|
	PC	MarkovPC	PC	MarkovPC	Naive Bayes	
Hypovolemia	2357	1014	96,000	98,000	96,000	3
LVFailure	2357	1061	96,000	96,000	92,000	4
Anaphylaxis	2357	73	100,000	100,000	94,000	1
InsuffAnesth	2357	1060	72,000	72,000	66,000	4
PulmEmboulus	2357	1217	96,000	98,000	94,000	3
Intubation	2357	1215	94,000	94,000	84,000	8
KinkedTube	2357	1233	72,000	72,000	92,000	4
Disconnect	2357	1112	90,000	90,000	90,000	2
Média	2357	998	89,500	90,000	88,500	

4.2 Redes Synthetic

As redes Synthetic foram redes criadas manualmente no intuito de se observar como os algoritmos PC e MarkovPC podem se comportar em determinados

contextos. As redes criadas levaram em consideração os parâmetros numéricos para não tornar os atributos irrelevantes. Os contextos criados são relativos à quantidade de nós no Markov Blanket de cada variável classe.

Nas Figuras das redes induzidas pelo PC tradicional os nós pertencentes ao MB da classe foram circundados.

4.2.1 Synthetic1

A variável classe, Node1, possui somente um nó no seu Markov Blanket. A rede, conforme Figura 4.13, possui 32 atributos e todos eles podem assumir somente os valores State0 ou State1. Foi gerada uma base de 5000 registros a partir dessa rede. As Figuras 4.14 e 4.15 mostram as redes referentes à base de dados Synthetic1 geradas pelo PC e pelo MarkovPC respectivamente.

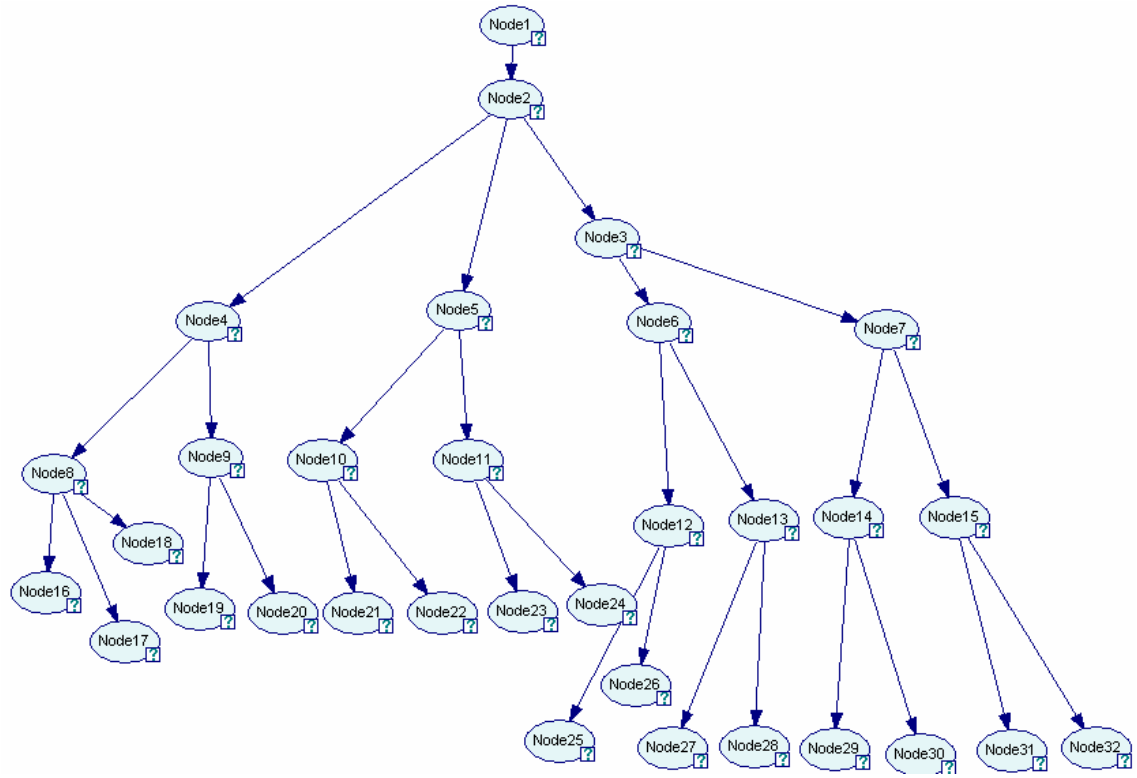


Figura 4.13. Rede original Synthetic1

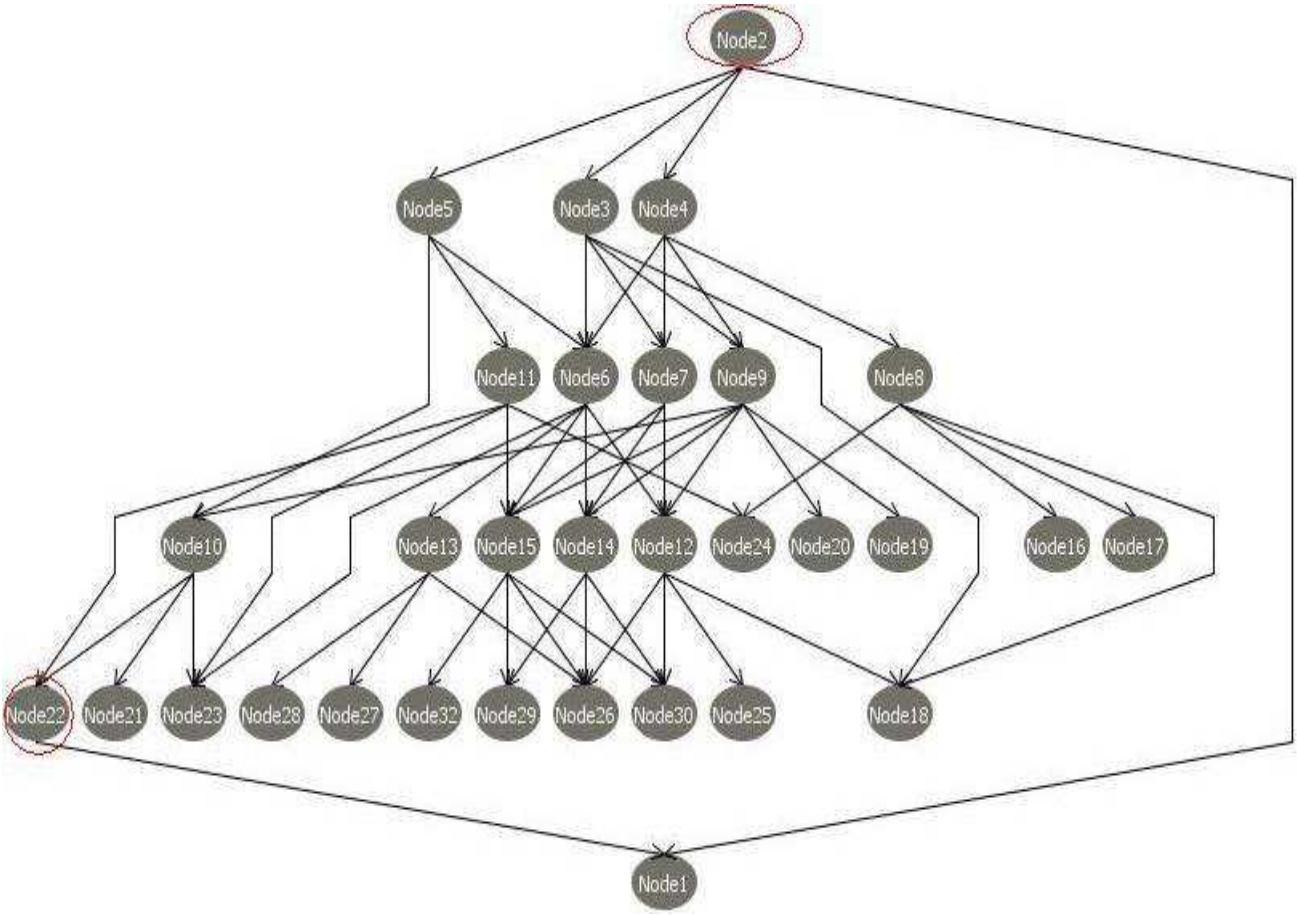


Figura 4.14. Synthetic1 gerada pelo PC tradicional

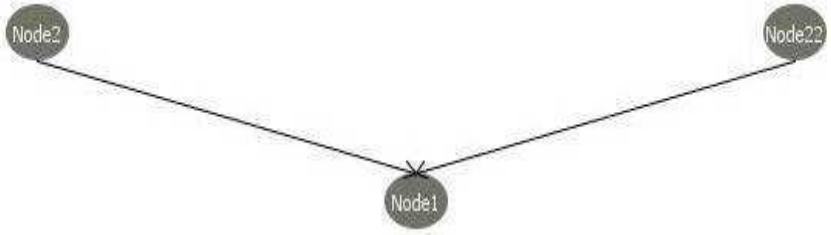


Figura 4.15. Synthetic1 gerada pelo MarkovPC

Os resultados de Synthetic1 estão na Tabela 4.6.

Tabela 4.6. Resultados de Synthetic1

REDE	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Synthetic 1	1663	737	89,160	89,160

Analisando-se a variável Node1 na rede Synthetic1 original (Figura 4.13) é possível identificar que faz parte do seu MB a variável Node2 somente. Observando-se a rede gerada pelo MarkovPC (Figura 4.15) nota-se que a variável do MB original está presente no MB da variável Node1. A variável Node22 aparece no MB de Node1 na rede tanto do PC como do MarkovPC, mas a rede do MarkovPC é muito menor e manteve a mesma taxa de classificação.

4.2.2 Synthetic2

A variável classe, Node1, possui quatorze nós no seu Markov Blanket. A rede, conforme Figura 4.16, possui 32 atributos e todos eles podem assumir somente os valores State0 ou State1. Foi gerada uma base de 5000 registros a partir dessa rede. As Figuras 4.17 e 4.18 mostram as redes referentes à base de dados Synthetic2 geradas pelo PC e pelo MarkovPC respectivamente.

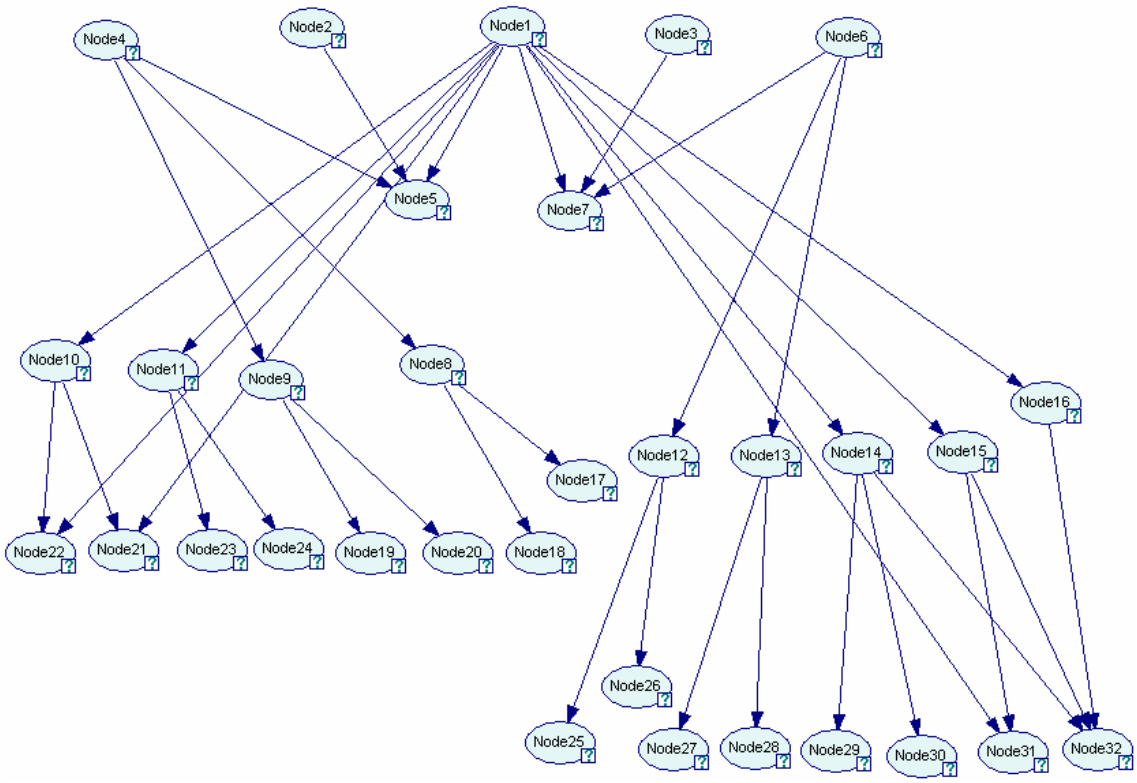


Figura 4.16. Rede Synthetic2

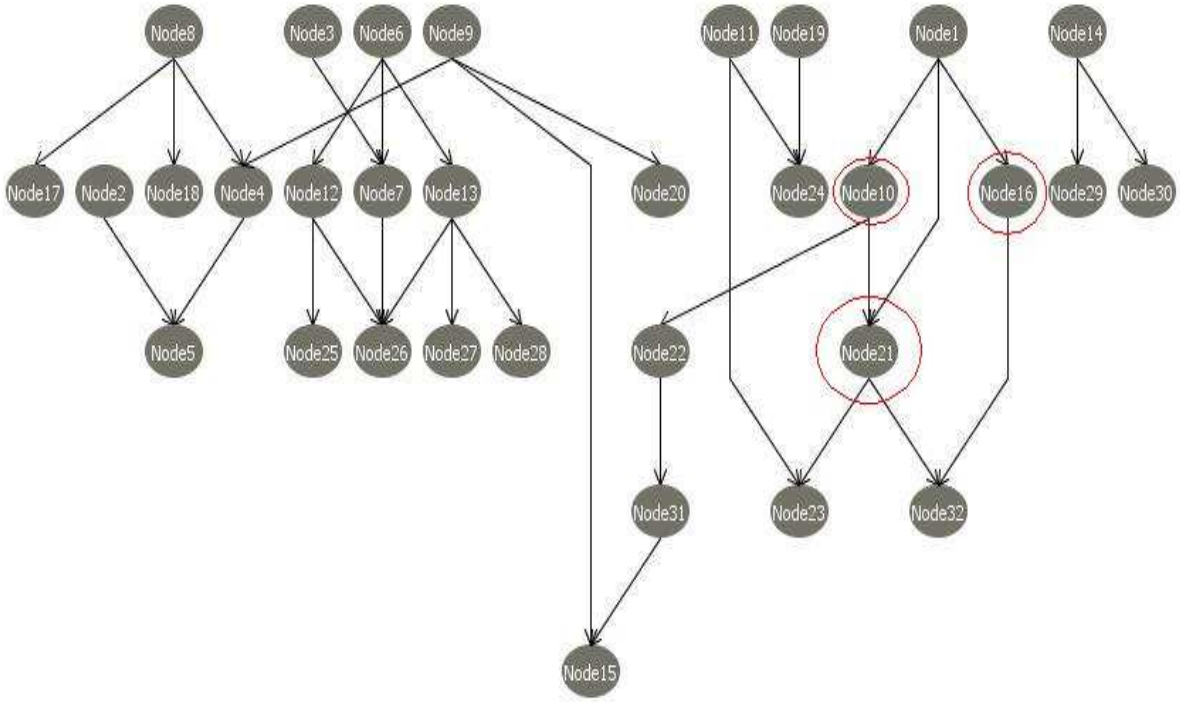


Figura 4.17. Synthetic2 gerada pelo PC tradicional

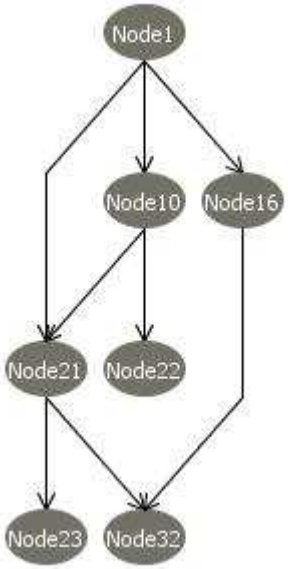


Figura 4.18. Synthetic2 gerada pelo MarkovPC

Os resultados de Synthetic2 estão na Tabela 4.7.

Tabela 4.7. Resultados de Synthetic2

REDE	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Synthetic 2	823	583	93,2	93,2

O Markov Blanket do Node1 nas redes geradas pelo PC tradicional e MarkovPC são idênticos. Porém não são iguais ao MB da rede original, mas todas as variáveis incluídas no MB do PC e MarkovPC estão presentes no original. Isto pode ocorrer devido a questões relativas às independências condicionais encontradas, mas tanto o PC como MarkovPC encontraram o mesmo MB de Node1 e tiveram a mesma taxa de classificação.

4.2.3 Synthetic3

A variável classe, Node1, possui todos os nós no seu Markov Blanket. A rede, conforme Figura 4.19, possui 32 atributos e todos eles podem assumir somente os valores State0 ou State1. Foi gerada uma base de 5000 registros a partir dessa rede. As Figuras 4.20 e 4.21 mostram as redes referentes à base de dados Synthetic3 geradas pelo PC e pelo MarkovPC respectivamente.

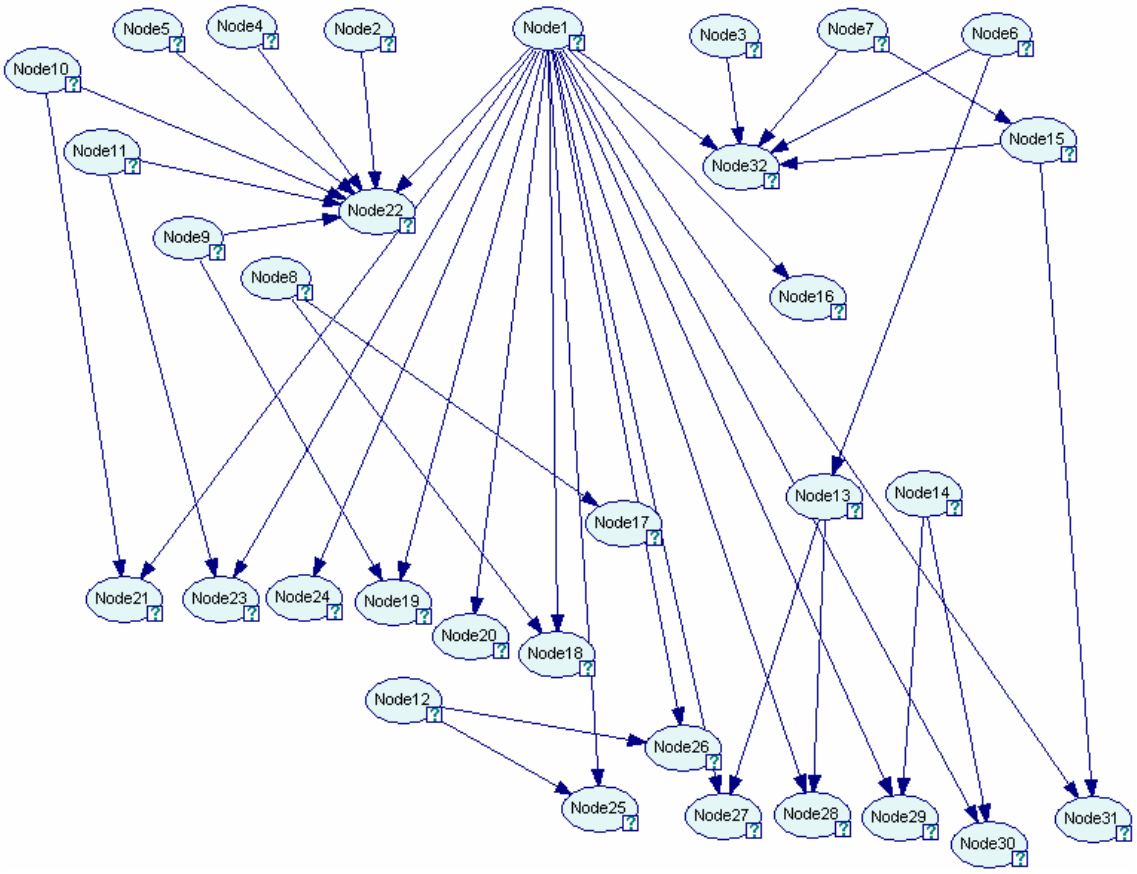


Figura 4.19. Rede Synthetic3

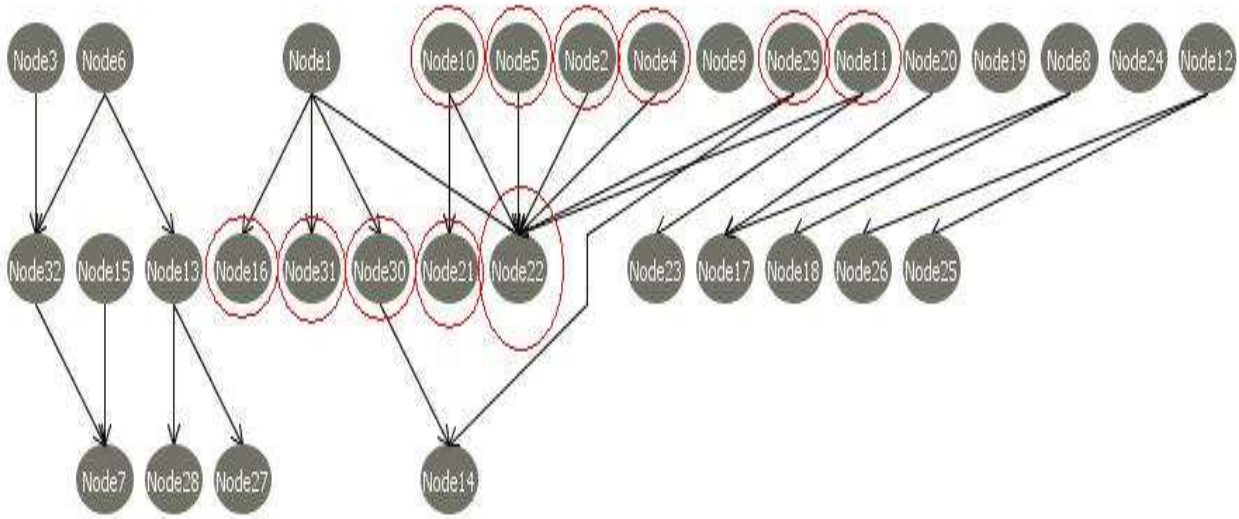


Figura 4.20. Synthetic3 gerada pelo PC tradicional

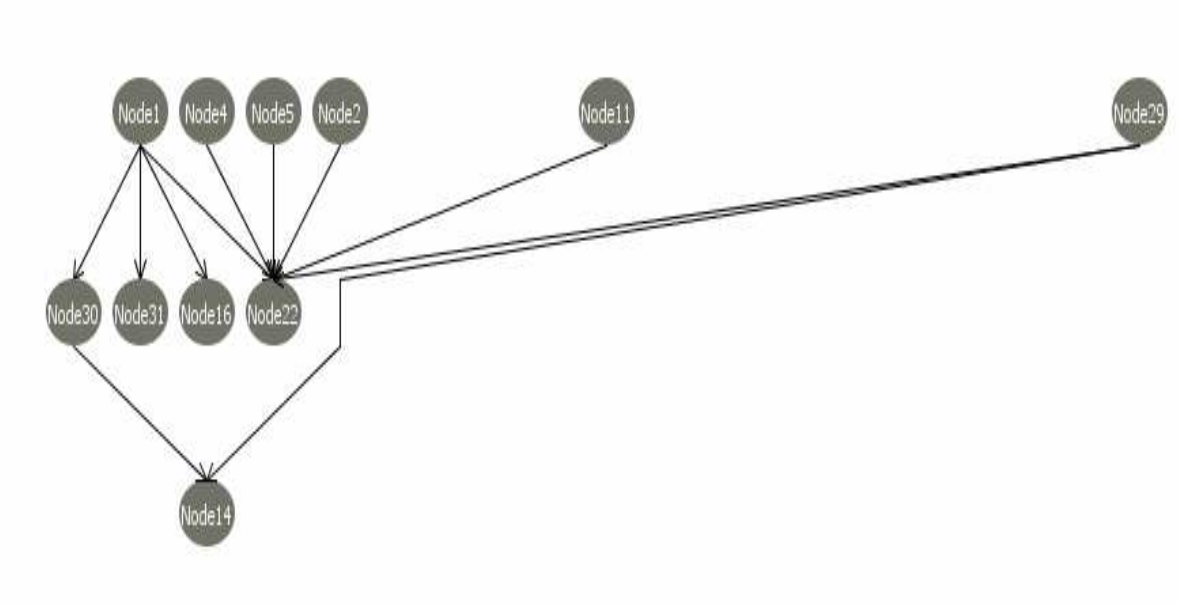


Figura 4.21. Synthetic3 gerada pelo MarkovPC

Os resultados de Synthetic3 estão na Tabela 4.8.

Tabela 4.8. Resultados de Synthetic3

REDE	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Synthetic 3	908	656	85,700	87,240

Na rede original Synthetic3 todos os nós fazem parte do MB da variável classe. Como esperado, foi o menor índice de ganho de esforço computacional que o MarkovPC obteve. Mesmo assim a rede do MarkovPC foi menor e o MB foi semelhante ao MB do PC tradicional, faltando somente 2 variáveis. Com essa redução de atributos ainda obteve pequeno ganho na taxa de classificação.

4.3 Car

A Car database é uma base de dados referente a avaliação de veículos. Foi derivada de um modelo de decisão hierárquico originalmente desenvolvido para a demonstração do sistema DEX [63]. Consiste de 1728 instâncias com 6 atributos nominais, sendo 4 atributos com domínio de 4 valores e os demais com um domínio de 3 valores. As Figuras 4.22 e 4.23 representam as redes induzidas pelo PC tradicional e MarkovPC respectivamente.

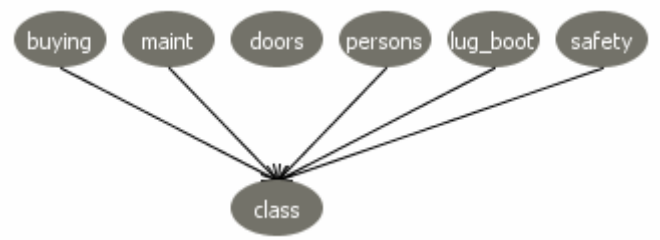


Figura 4.22. Rede Car gerada pelo PC tradicional

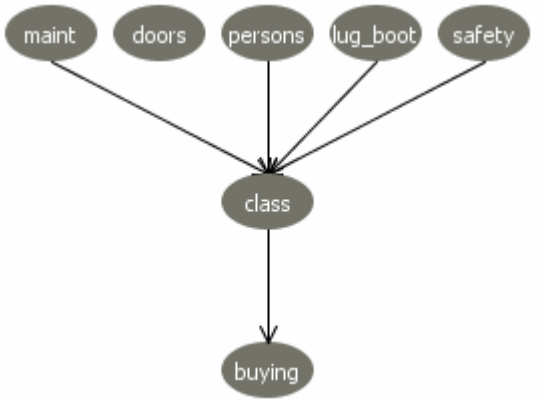


Figura 4.23. Rede Car gerada pelo MarkovPC

Os resultados da base Car estão na Tabela 4.9.

Tabela 4.9. Resultados da base Car

Base	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Car	76	34	89,815	93,576

Analisando-se as redes geradas pelos algoritmos PC e MarkovPC, é possível notar que a estrutura delas é exatamente a mesma. Somente o direcionamento dos arcos que foi diferente, no caso do arco entre as variáveis *class* e *buying*. Esse direcionamento foi o motivo para a diferença entre as taxas de classificação, na qual o MarkovPC se comportou melhor e ainda exigiu menor esforço para a indução. Essa redução de esforço foi devido ao fato de que o MarkovPC desconsiderava a variável *door* que desde os testes iniciais do algoritmo já não podia ser candidata a estar no MBC, pois na lista de independências havia uma independência de cardinalidade 0 entre *door* e *class* e o MarkovPC inicia as comparações (linha 7 da Figura 4.1) com a variável classe.

As regras de direcionamento dos arcos podem ser arbitrárias quando não satisfazem as condições conforme o item D da Figura 2.9 (o algoritmo PC), e o MarkovPC direcionou de forma diferente do PC tradicional.

4.4 Kr-vs-Kp

Base de dados que representa um tabuleiro de xadrez para uma jogada final, conforme descrita em [64]. Esta base consiste de 3196 instâncias e 36 atributos sem valores ausentes. Dois atributos possuem um domínio de 3 valores e todos os demais, incluindo a variável-classe, um domínio de 2 valores. As Figuras 4.24 e 4.25 representam as redes geradas pelos algoritmos PC tradicional e MarkovPC.

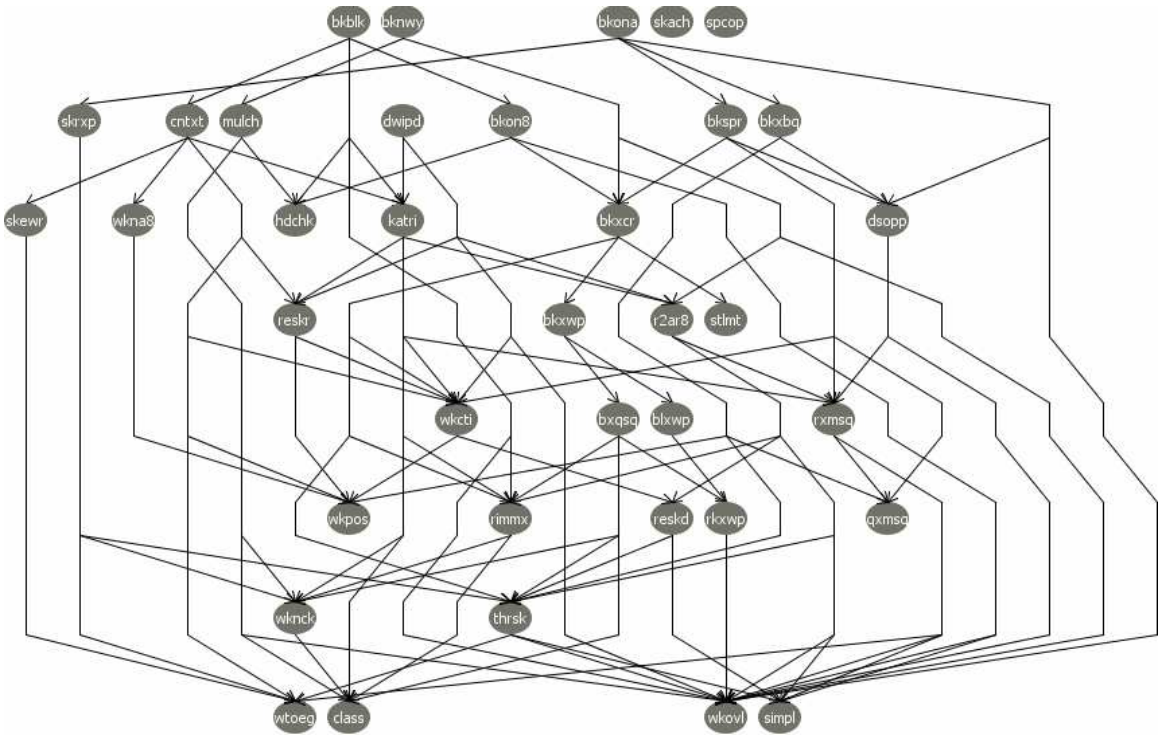


Figura 4.24. Rede Kr-vs-Kp gerada pelo algoritmo PC tradicional

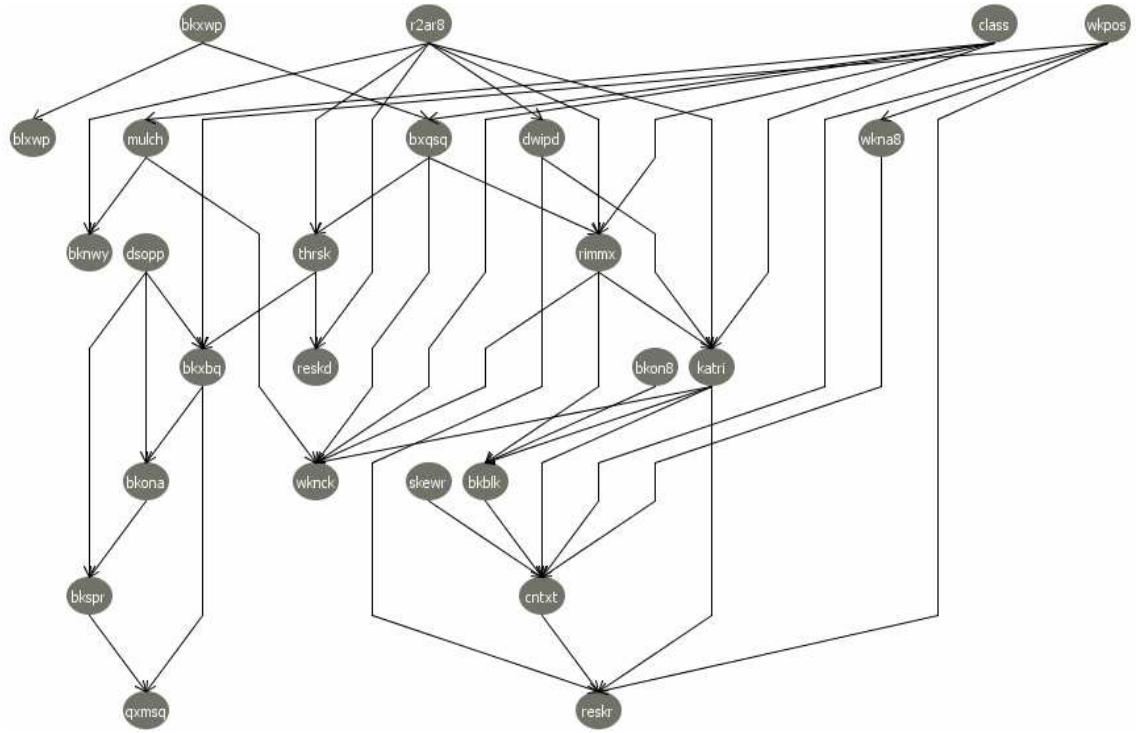


Figura 4.25. Rede Kr-vs-Kp gerada pelo algoritmo MarkovPC

Os resultados da base Kr-vs-Kp estão na Tabela 4.10.

Tabela 4.10. Resultados da base Kr-vs-Kp

Base	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Kr-vs-Kp	6129	2127	96,089	94,086

Observando-se as redes geradas pelos algoritmos pode-se notar que a rede gerada pelo MarkovPC foi visivelmente menor e o seu esforço, consequentemente, bastante menor também. Porém, a taxa de classificação obtida pelo MarkovPC para esta base foi levemente menor, apesar de o MBC ter sido consistente com a rede do PC tradicional. O motivo, mais uma vez, para a diferença nas taxas de classificação foi devido ao direcionamento feito em algumas variáveis ligadas à classe.

4.5 Lung-Cancer

Essa base de dados foi usada em [65] para ilustrar o poder de um plano discriminante ótimo em situações desfavoráveis, como essa em que há somente uma amostra pequena de dados para uma alta quantidade de atributos. Os dados descrevem três tipos patológicos de câncer de pulmão, que é a variável classe. A base consiste de 32 instâncias somente, com 57 atributos nominais ao total. Desses atributos, 2 deles têm um domínio de 4 valores, 13 deles com um domínio de 2 valores, e os demais contam com um domínio de 3 valores. As Figuras 4.26 e 4.27 representam as redes geradas para esta base.

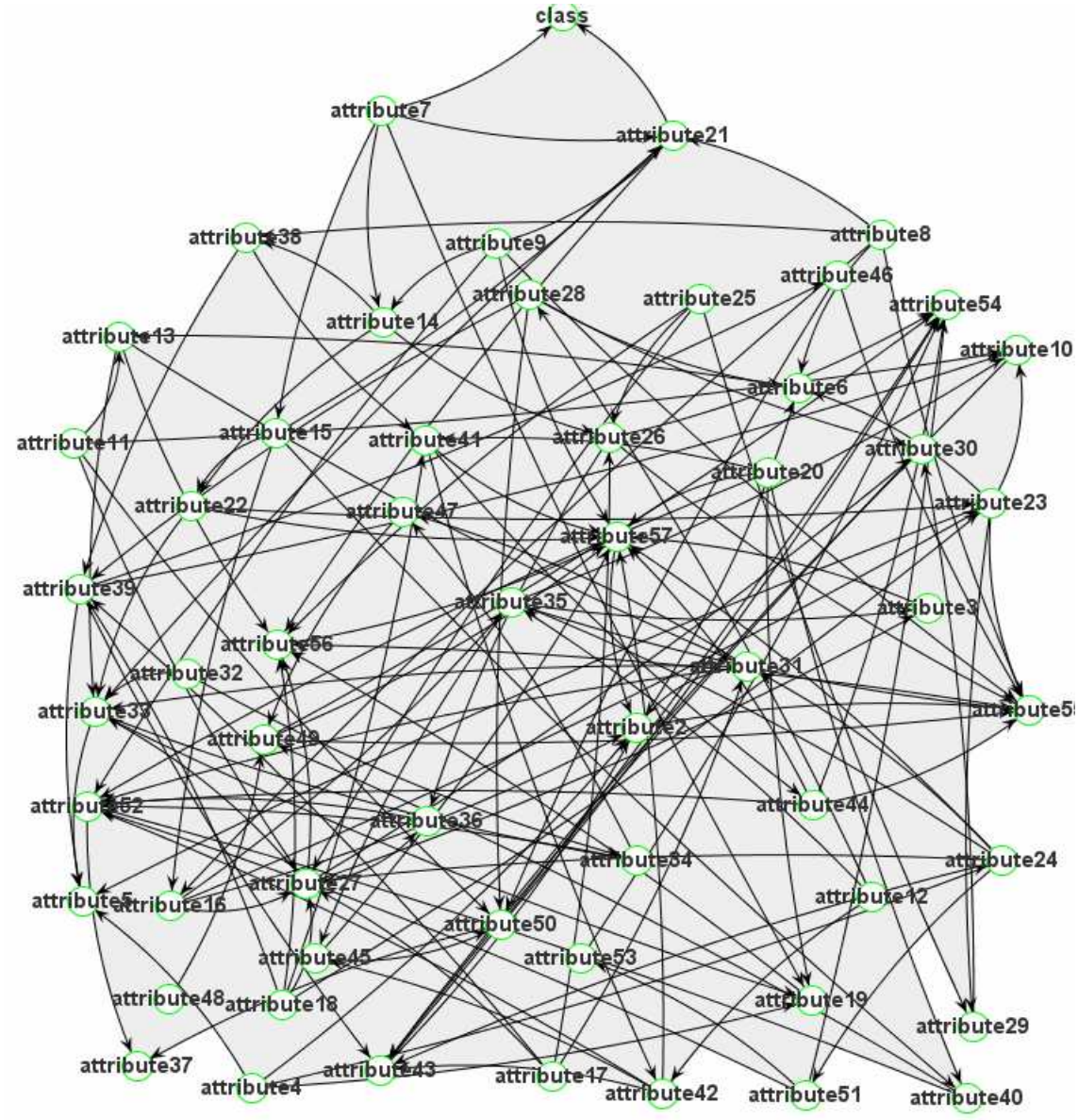


Figura 4.26. Rede Lung-Cancer gerada pelo PC tradicional

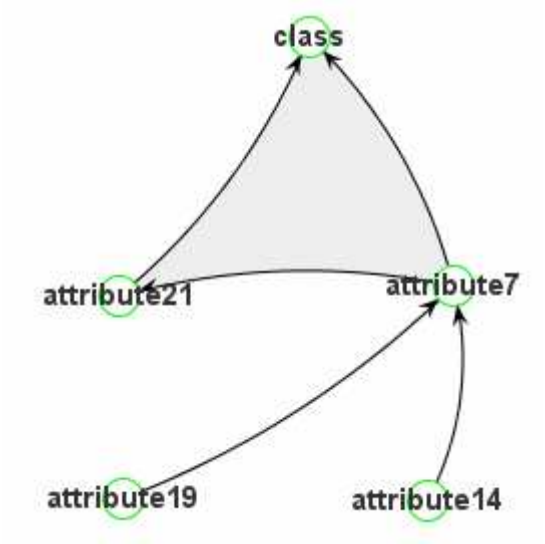


Figura 4.27. Rede Lung-Cancer gerada pelo MarkovPC

Os resultados da base Lung-Cancer estão na Tabela 4.11.

Tabela 4.11. Resultados da base Lung-Cancer

Base	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Lung-Cancer	5881	2151	75,000	75,000

O algoritmo MarkovPC conseguiu um esforço visivelmente menor que o PC tradicional, cerca de 64% de melhora, e manteve a mesma taxa de classificação. Por se tratar de uma base contendo apenas uma pequena amostra de dados, as taxas de classificação não conseguiram ser muito elevadas. Entretanto, como a quantidade de atributos era grande (57 ao todo), o MarkovPC utilizando a estratégia baseada no MBC conseguiu a considerável redução de esforço computacional e simplificou bastante a rede gerada mantendo o MBC consistente. Este teste demonstra, também, que o algoritmo MarkovPC não tende a ser influenciado negativamente por amostras pequenas, tendo o mesmo desempenho que o PC tradicional em termos de classificação para esse contexto.

4.6 Patient

Esta base de dados foi inicialmente usada em [66], e descreve informações sobre pacientes em estado pós-operatório. A variável classe determina se um paciente deve ser enviado à UTI, receber alta ou ser enviado ao quarto de tratamento geral. Esta base consiste de 90 instâncias com 9 atributos nominais. Desses atributos, 6 deles (incluindo a classe) têm um domínio de 3 valores, 2 atributos um domínio de 2 valores, e o outro atributo com um domínio de 4 valores. A Figura 4.28 representa a rede gerada pelo PC tradicional e MarkovPC. Nesta rede todos os nós são independentes uns dos outros, então, é uma rede cujo único nó é o nó que representa a classe.

Figura 4.28. Rede Patient gerada pelos algoritmos PC e MarkovPC

Os resultados da base Patient estão na Tabela 4.12.

Tabela 4.12. Resultados da base Patient

Base	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Patient	36	54	71,111	71,111

A rede gerada tanto pelo PC tradicional como pelo MarkovPC foi exatamente a mesma, pois, segundo a lista de independências utilizada, todas as variáveis eram independentes em cardinalidade 0. Neste caso, o algoritmo MarkovPC caiu no caso extremo descrito no início deste Capítulo: todas as variáveis já não faziam parte do MBC de acordo com as eliminações de arcos em cardinalidade 0, e o MarkovPC continuou testando se os atributos estavam no MBC de forma a eliminá-los. Esses testes adicionais de pertinência de grupo (o MBC) causaram os 9 passos adicionais, para cada cardinalidade, no esforço do MarkovPC. Esses passos são os testes de pertinência para os atributos (a base tem 9 atributos). Nota-se então um contexto

onde o MarkovPC pode ter um esforço maior que o PC tradicional, ainda que mantenha a mesma rede gerada. Porém, vale ressaltar que esses casos são extremos.

4.7 Solar-flare 1

Nesta base cada instância representa características capturadas para uma região ativa no sol. A variável classe indica o tipo de radiação solar na região em questão nas próximas 24 horas. A base consiste de 13 atributos ao todo, com 323 instâncias. Os domínios das variáveis estão descritos na Tabela 4.13.

Tabela 4.13. Domínio das variáveis da base Solar-Flare 1

atributo	#estados
Class	6
Largest_spot_size	6
Spot_distribution	4
Activity	2
Evolution	3
Previous_24_hour_flare_activity_code	2
Historically-complex	2
Did_region_become_historically_complex	2
Area	2
Area_of_the_largest_spot	2
C-class_flares_production_by_this_region	3
M-class_flares_production_by_this_region	4
X-class_flares_production_by_this_region	2

As Figuras 4.29 e 4.30 representam as redes geradas pelos algoritmos PC tradicional e MarkovPC, respectivamente.

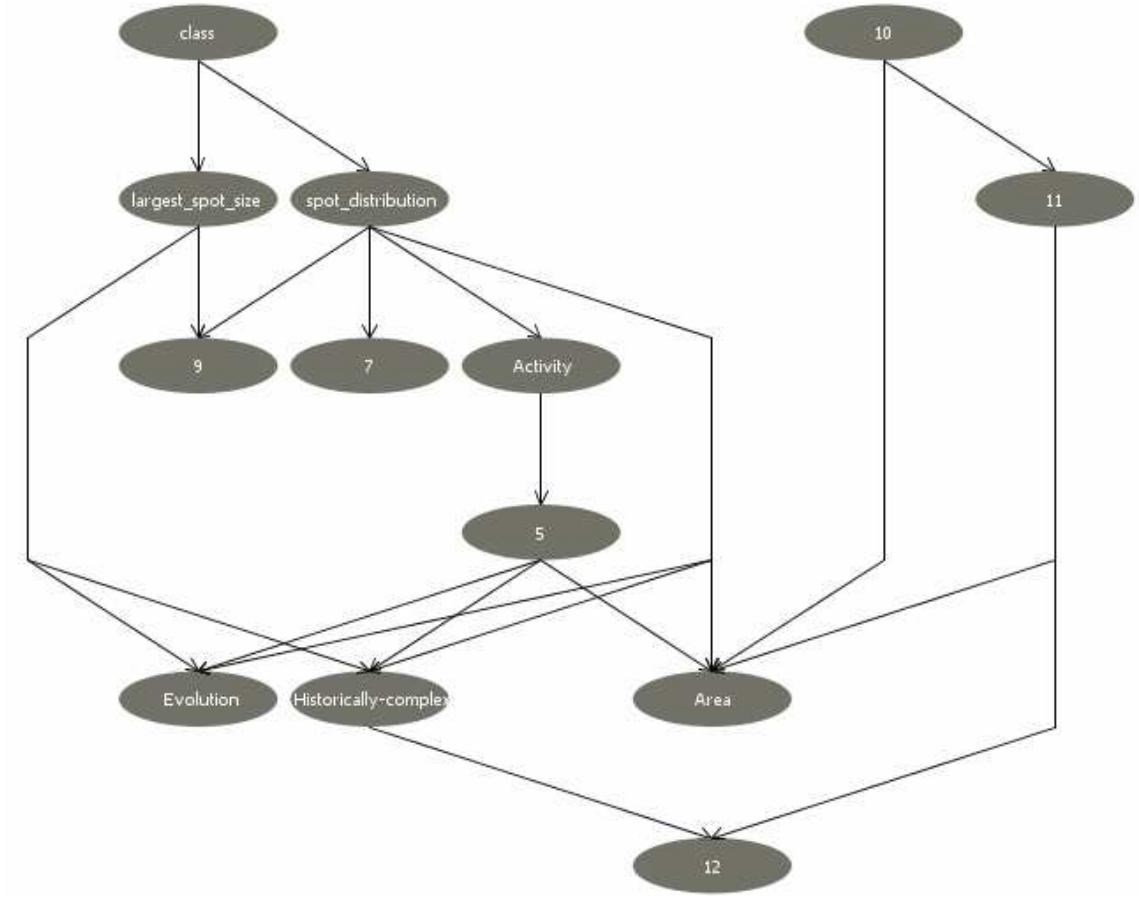


Figura 4.29. Rede Solar-Flare 1 gerada pelo PC tradicional

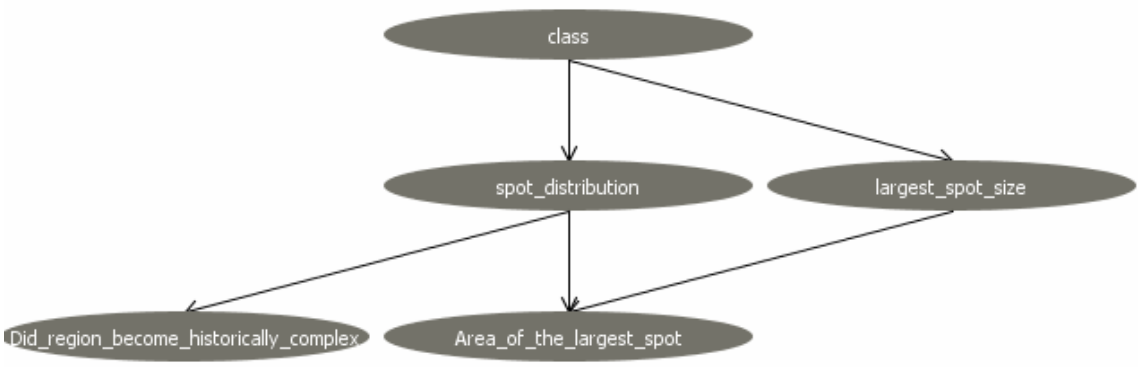


Figura 4.30. Rede Solar-Flare 1 gerada pelo MarkovPC

Os resultados da base Solar-Flare 1 estão na Tabela 4.14.

Tabela 4.14. Resultados da base Solar-Flare 1

Base	ESFORÇO		TAXA CLASSIFICAÇÃO	
	PC	MarkovPC	PC	MarkovPC
Solar-flare 1	418	176	72,755	73,065

O MarkovPC conseguiu induzir uma rede menor, com esforço significativamente menor (cerca de 58%) e ainda manteve o MBC consistente como pode-se observar pelas Figuras X e Y. Além disso, o MarkovPC ainda obteve um leve ganho na taxa de classificação. Isso demonstra a consistência do método em bases de dados reais, como a Solar-Flare 1 por exemplo.

4.8 Solar-Flare 2

A base Solar-Flare 2 tem exatamente a mesma estrutura da base Solar-Flare 1, porém a Solar-Flare 2 contém dados diferentes, os quais sofreram uma maior correção de erros e, consequentemente, é considerada mais confiável.

As Figuras 4.31 e 4.32 representam as redes geradas pelos algoritmos PC tradicional e MarkovPC, respectivamente.

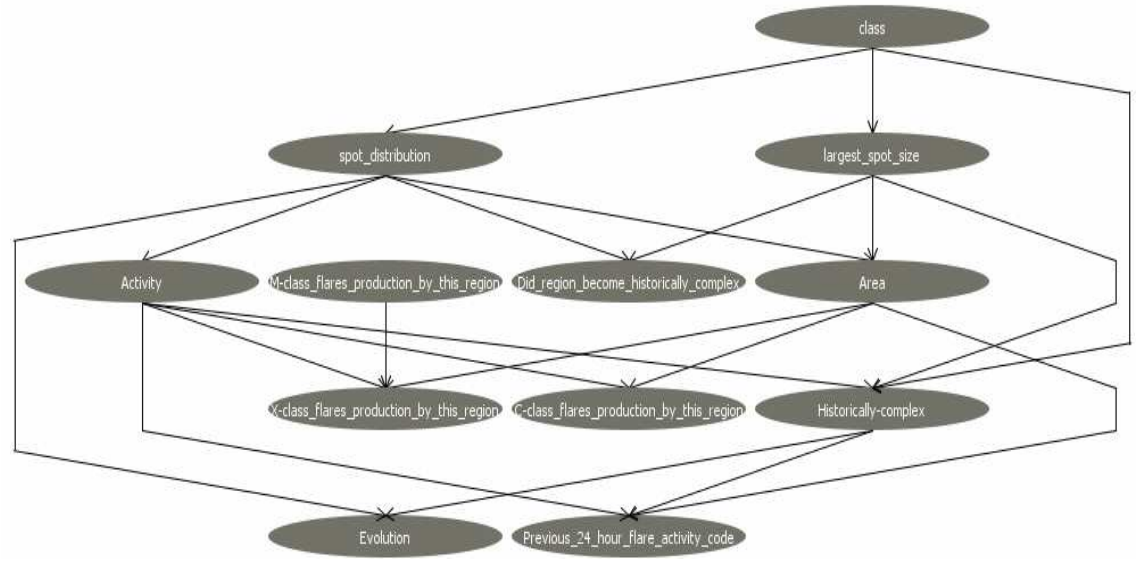


Figura 4.31. Rede Solar-Flare 2 gerada pelo PC tradicional

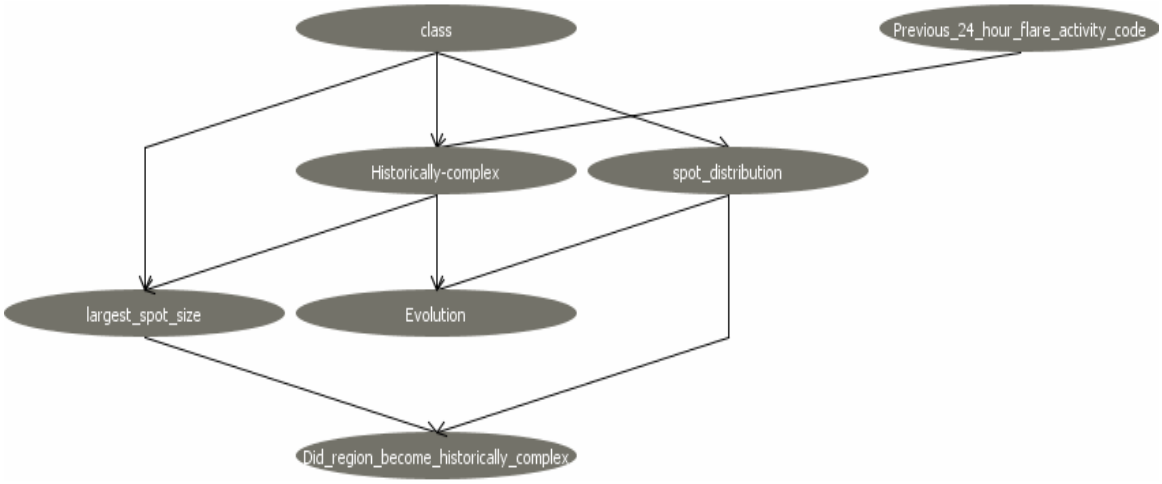


Figura 4.32. Rede Solar-Flare 2 gerada pelo MarkovPC

Os resultados da base Solar-Flare 2 estão na Tabela 4.15.

Tabela 4.15. Resultados da base Solar-Flare 2

	ESFORÇO		TAXA CLASSIFICAÇÃO	
Base	**PC**	**MarkovPC**	**PC**	**MarkovPC**
Solar-Flare 2	342	238	75,235	74,859

A análise dos resultados para a Solar-Flare 2 não diferem muito em relação aos resultados de Solar-Flare 1. O MarkovPC conseguiu induzir uma rede

menor, com esforço menor (cerca de 30%) e manteve o MBC consistente também como pode-se observar pelas Figuras X e Y. Porém, para a Solar-Flare 2 o MarkovPC sofreu uma leve perda na taxa de classificação.

4.9 Análise Geral dos Resultados

A fim de se elaborar uma análise comparativa mais robusta, além de apresentar as taxas de acerto de classificação obtidas utilizando o método MarkovPC proposto e o PC tradicional, será também mostrado o desempenho do classificador NaiveBayes aplicado em todas as bases descritas anteriormente. Para o NaiveBayes (NB), utilizou-se da mesma estratégia de treinamento e testes com uma validação cruzada de 10 *folds*. Na Tabela 4.16 são apresentadas as taxas médias de classificação para todos os métodos usados em todas as bases utilizadas.

A Tabela 4.16 revela que, considerando-se os domínios testados, o esforço computacional (número de testes) realizados pelo MarkovPC é, na média, de apenas 43,36% do esforço total feito pelo PC tradicional. Apenas no domínio Patient o MarkovPC executou mais testes comparativos que o PC tradicional, conforme explicado na Seção 4.6.

Foi aplicado o teste-t [67] para comparação entre os resultados de classificação do PC e do MarkovPC, e entre o MarkovPC e o Naive-Bayes. O teste estatístico conhecido como teste-t avalia se dois grupos são significativamente diferentes um do outro. O valor alpha [67] deste teste foi escolhido como 0.05. Os resultados obtidos por este teste permitem que se conclua que a diferença entre os grupos avaliados (nos dois casos) é significativamente diferente (mesmo considerando-se a variabilidade). Desta forma, pode-se considerar que, além de exigir menor esforço computacional, o MarkovPC se desempenhou melhor que o PC tradicional em termos de resultado de classificação. Ainda sobre as taxas de classificação, o MarkovPC se apresentou sensivelmente melhor (na média) que o NaiveBayes.

Analisando-se a complexidade do classificador, pode-se dizer que o PC e o NaiveBayes geram classificadores contendo todas as variáveis presentes no conjunto de dados do domínio. O MarkovPC, no entanto, tende a reduzir o número de variáveis presentes no classificador induzido. Considerando-se as 17 simulações relatadas na

Tabela 4.16, na média, o PC e o NaiveBayes produziram modelos tendo 31,52 variáveis. O MarkovPC, por outro lado, produziu classificadores com 7,52 variáveis na média. Assim, os classificadores induzidos pelo MarkovPC têm apenas 23,85% das variáveis presentes nos classificadores gerados pelo PC tradicional e pelo NaiveBayes.

Tabela 4.16. Resultados gerais

Domínio	Esforço (Número de passos)		TMAC (%)			|MB|
	PC	MarkovPC	PC	MarkovPC	NB	
ALARM Hypovolemia	**2758**	**1121**	**97.98**	**98.38**	**96.55**	**3**
ALARM LVFailure	**2758**	**1135**	**98.95**	**99.03**	**96.41**	**4**
ALARM Anaphylaxis	**2758**	**1052**	**98.98**	**98.98**	**97.26**	**1**
ALARM InsuffAnesth	**2758**	**1442**	**81.85**	**84.77**	**63.34**	**4**
ALARM PulmEmboulus	**2758**	**1160**	**99.21**	**99.42**	**97.30**	**3**
ALARM Intubation	**2758**	**1458**	**97.55**	**98.46**	**84.96**	**8**
ALARM KinkedTube	**2758**	**1327**	**98.87**	**98.91**	**85.30**	**4**
ALARM Disconnect	**2758**	**1145**	**96.16**	**98.74**	**92.95**	**2**
Synth 1	**1663**	**797**	**89.16**	**89.16**	**83.42**	**1**
Synth 2	**823**	**599**	**93.20**	**93.20**	**93.20**	**14**
Synth 3	**908**	**610**	**85.30**	**86.74**	**83.16**	**31**
Car	**76**	**34**	**89.81**	**93.57**	**85.64**	**N/A**
Kr-vs-kp	**6129**	**2127**	**96.08**	**94.08**	**87.76**	**N/A**
Lung-Cancer	**5881**	**2151**	**75.00**	**75.00**	**46.87**	**N/A**
Patient	**36**	**54**	**71.11**	**71.11**	**68.88**	**N/A**
Solar-flare 1	**418**	**176**	**72.75**	**73.06**	**66.25**	**N/A**
Solar-flare 2	**342**	**238**	**75.23**	**74.85**	**73.92**	**N/A**
Média	**2255.2**	**977.94**	**89.25**	**89.86**	**82.54**	**-**
Desvio Padrão	**1787.1**	**643.45**	**10.33**	**10.38**	**14.28**	**-**

Ainda analisando-se os resultados obtidos, pode-se dizer que o algoritmo MarkovPC conseguiu trazer ganhos em termos de esforço computacional em quase todas as bases (somente na Patient foi pior). Na base de dados com mais variáveis, a ALARM com Anaphylaxis como classe, o ganho no esforço chegou a mais de 70%. Em quase todas as bases (exceto Kr-vs-kp e Solar-flare 2) o MarkovPC não piorou a taxa de classificação apesar da redução da rede e do esforço, e ainda conseguiu até pequenos aumentos nas taxas de classificação de diversas redes.

Há uma tendência de que quanto maior a rede e menor o Markov Blanket da variável classe, melhor se sai o MarkovPC. Isso é muito favorável para redes extensas, onde o desempenho do algoritmo de aprendizado é fundamental para garantir

que o processo de mineração de dados seja viável, principalmente no quesito de tempo de aprendizado e esforço computacional.

Vale lembrar que a rede gerada pelo MarkovPC representa uma possível seleção de atributos, mas não é necessariamente a melhor ou a seleção mais otimizada. É possível, caso seja desejado, que se usem outros classificadores utilizando as variáveis selecionadas pelo MarkovPC, ou ainda seria possível retirar as variáveis que não pertencem ao Markov Blanket da classe na rede final. O MarkovPC tem como intuito gerar uma rede com os atributos mais relevantes e que seja também um Classificador Bayesiano. Conforme os resultados, ele atinge com sucesso esses objetivos e incorpora ganhos computacionais tanto na criação da rede como na inferência (classificação), a qual é feita levando em conta menos atributos e assim é mais rápida.

4.10 Conclusões

Este Capítulo propôs e discutiu uma nova abordagem de aprendizado de classificadores bayesianas por independência condicional baseada no conceito de Markov Blanket. O método aqui proposto, batizado de MarkovPC, utiliza o conceito de MB para impôr algumas restrições e otimizar o PC tradicional durante a indução da rede. Esta abordagem introduz uma novidade frente aos trabalhos descritos na literatura que usam o MB somente após a indução da rede ter sido feita.

Os experimentos demonstraram que o MarkovPC tende a ser mais preciso (em termos de taxas de classificação) do que o PC tradicional e o NaiveBayes. Além disso, o MarkovPC produz classificadores mais simples (com menos variáveis) e demanda menor número de testes de IC durante o aprendizado.

A idéia de usar o conceito de MB durante a indução da rede levou à idéia de se usar esse conceito também durante outra importante fase do aprendizado por IC: os testes de IC para gerar as listas de independências de entrada para os algoritmos de IC. Esta idéia é descrita no Capítulo a seguir.

5 TESTE DE INDEPENDÊNCIA CONDICIONAL UTILIZANDO UMA ESTRATÉGIA BASEADA EM MARKOV BLANKET

Conforme abordado anteriormente, os algoritmos de aprendizado de Redes Bayesianas baseados em independência condicional exigem, como entrada, um conjunto de independências condicionais. Existem diversas abordagens para a obtenção dessas independências, entre elas o cálculo de probabilidade conjunta [4], teste do qui-quadrado [67], *Pearson's conditional independence chi-square* [69] e testes baseados em pontuação (*score based)*[70]. Diversas variações desses métodos são encontradas na literatura, como em [76,77,51], porém, o propósito de todas é encontrar as independências condicionais contidas na amostragem. Esses métodos fazem testes de independência condicional (IC) de acordo com a cardinalidade [60], a qual representa o tamanho do conjunto de variáveis que tornam as variáveis testadas independentes ou não. Por exemplo: considere que uma variável aleatória A é independente de B dado C e D. Neste caso, a cardinalidade é 2, pois há duas variáveis que tornam B não significativo para A: a variável C e a D. Caso A seja independente de B dado um cojunto vazio, diz-se que a cardinalidade é 0 (zero). Uma abordagem mais detalhada sobre métodos de cálculo de independência condicional pode ser encontrada em [41,60].

A maioria dos testes de IC, principalmente os mais tradicionais, realiza os testes entre todas as variáveis para cada cardinalidade. Suponha um conjunto de 4 variáveis (A, B, C e D) e que I(A,B|{S}) representa um teste de independência entre as variáveis A e B dado um conjunto de variáveis contidas em S. Serão realizados os testes de IC como mostrados na Tabela 5.1.

Tabela 5.1. Testes de Independência Condicional

I(A,B|{})	I(A,B|{C})	I(A,B|{D})	I(A,B|{C,D})
I(B,C|{})	I(B,C|{A})	I(B,C|{D})	I(B,C|{A,D})
I(C,D|{})	I(C,D|{A})	I(C,D|{B})	I(C,D|{A,B})

Esses testes de IC são exponenciais em relação à quantidade de variáveis e, portanto, exigem grande esforço computacional para serem calculados. Um básico algoritmo genérico para se encontrar uma lista de independências é apresentado na

Figura 5.1.

```
GenericConditionalIndependence Algorithm
Problem: find an IND list of conditional independence in data.
Input  : a D dataset.
Output : an IND list containing the found independencies
-----------------------------------------------------------
begin
 1 i=0;          //initial cardinality is 0
 2 MAX_CARD = 2; //maximum cardinality to be tested
 2 IND={};       //the independencies set begins empty
 3
 4 while i<=MAX_CARD
 5 {
 6   for each X in D //X represents a variable in D
 7   {
 8       for each Y in D //Y represents a variable in D
 9       {
10         SepSet = NEXT_SEPARATOR_SET(i);
11         while EXISTS SepSet
12         {
13           if CONDITIONALLY_INDEPENDENT(X,Y,SepSet)
14           {
15             add I(X,Y|{SepSet}) to IND;
16           }
17           SepSet = NEXT_SEPARATOR_SET(i);
18          }
19        }
20    }
21    i = i+1;
22 }
23
24 return IND;
end.
```

Figura 5.1. Um algoritmo de Independência Condicional genérico

Sobre o algoritmo apresentado na Figura 5.1, a linha 10 chama o procedimento que retorna o próximo possível conjunto separador para cardinalidade passada. Esse procedimento gera todas as possíveis combinações de variáveis que possam separar, condicionalmente, X e Y na cardinalidade i. Na linha 13 é chamado o procedimento que faz o teste de I(X,Y|{SepSet}). O procedimento CONDITIONALLY_INDEPENDENT(X,Y,SepSet) é o responsável por fazer o teste de IC em si, que pode usar qualquer método de IC disponível, como, por exemplo, os citados no início deste Capítulo. Por isso então o algoritmo na Figura 5.1 pode ser considerado como genérico, pois ilustra somente os passos tradicionais para a geração da lista de independências.

Para tarefas de classificação, verificou-se que é possível aplicar a estratégia baseada em Markov Blanket, conforme utilizada no MarkovPC, para testes de IC também. Esta idéia pode ser considerada independente do método específico de teste de IC e está sumarizada nos macro-passos descritos na Figura 5.2.

Teste de IC baseado em Markov Blanket

Problema: encontrar uma lista de independências (IND) em um conjunto de dados (D) de forma otimizada utilizando-se o conceito de Markov Blanket (MB)

Entrada: um conjunto de dados onde serão realizados os testes e a variável classe

---

Passos:

a) Encontrar as independências de cardinalidade 0 da forma usual
b) Gerar uma estrutura de rede com as independências encontradas no passo a, e definir o Markov Blanket da Classe (MBC)
c) Remover dos testes as variáveis que estiverem fora do MBC
d) Continuar os testes e remover, a cada avanço na cardinalidade, as variáveis que não estiverem no MBC.

Figura 5.2. Pseudo-código do algoritmo PC, adaptado de [17,60].

Os passos descritos na Figura 5.2 ilustram a idéia central da estratégia do uso de MB em testes de IC. Com base no algoritmo genérico de IC mostrado na Figura A, foi induzido um algoritmo de IC genérico também aplicando a idéia do MB. Este algoritmo está descrito na Figura 5.3.

```
MB-based GenericConditionalIndependence Algorithm
Problem: find an IND list of conditional independence in data using
          the MB concept.
Input  : a D dataset and the CLASS class variable.
Output : an IND list containing the found independencies
-------------------------------------------------------------------
begin
 1 i=0;          //initial cardinality is 0
 2 MAX_CARD = 2; //maximum cardinality to be tested
 2 IND={};       //the independencies set begins empty
 3
 4 while i<=MAX_CARD
 5 {
 6   for each X in D //X represents a variable in D
 7   {
 8       for each Y in D //Y represents a variable in D
 9       {
10         network = induce a network structure using the current IND
11         MBC = MARKOV_BLANKET(network,CLASS)
12         for each V in VARS
13         {
14           if MBC ∝ V then
15             remove V from VARS
17         }
18       }
19    for each X in VARS
20    {
21     for each Y in VARS
22     {
23       SepSet = NEXT_SEPARATOR_SET(i);
24       while EXISTS SepSet
25       {
26         if CONDITIONALLY_INDEPENDENT(X,Y,SepSet)
27         {
28           add I(X,Y|{SepSet}) to IND;
29         }
30         SepSet = NEXT_SEPARATOR_SET(i);
31       }
32     }
33    }
34    i = i+1;
35 }
36
37 return IND;
end.
```

Figura 5.3. Um algoritmo genérico de independência condicional usando MB

Comparando-se os algoritmos mostrados nas Figuras 5.1 e 5.3, é possível se observar que a diferença mais relevante está nas linhas de 8 a 18. Na linha 10, é gerada uma estrutura de rede considerando as independências encontradas até então. Essa rede poderia ser gerada usando o algoritmo PC tradicional, por exemplo. Na linha

seguinte (linha 11), a variável MBC representa o conjunto de variáveis que formam o Markov Blanket da CLASSE na rede induzida anteriormente. Vale notar que existem diversas maneiras de se encontrar um MB, e, no algoritmo aqui mostrado, foi optado pelo uso de geração de uma rede Bayesiana (ainda não direcionada) e nela encontrando-se o MB da classe, que seriam os nós pais da classe, os nós filhos e os pais dos filhos. Nos passos seguintes, até a linha 18, são removidos do conjunto de variáveis VARS os nós que não estiverem contidos no MBC. Desta forma, o procedimento PROXIMO_CONJUNTO_SEPARADOR irá considerar uma quantidade reduzida de variáveis para a geração do próximo conjunto. Além dessa diminuição do espaço de busca para o PROXIMO_CONJUNTO_SEPARADOR, os laços de repetição em VARS (linhas 19 e 21) tendem a considerar menos variáveis e serão feitas menos repetições e, consequentemente, menos chamadas ao procedimento INDEPENDENTE_CONDICIONALMENTE(X,Y,SepSet), que é um procedimento custoso computacionalmente na maioria dos métodos de IC.

Essa redução no conjunto de variáveis a serem testadas (testes de IC) pode diminuir significativamente a quantidade de testes de IC realizados. Conforme citado anteriormente, esses testes podem ser exponenciais em relação à quantidade de atributos, e uma redução, como a aqui proposta, pode trazer muitos ganhos em esforço computacional.

Uma vez que o conjunto de variáveis a serem testadas (testes de IC) tende diminuir a cada vez que a cardinalidade é aumentada, pode-se inclusive tornar viável testes de cardinalidades maiores que, com testes tradicionais, são inviáveis muitas vezes.

Foram realizados testes nas bases de dados utilizadas pelo MarkovPC no Capítulo anterior, a fim de se validar a robustez deste método de IC usando o conceito de MB. Para os testes efetuados, o procedimento INDEPENDENTE_CONDICIONALMENTE foi implementado usando-se duas abordagens: probabilidade conjunta e outra baseada em pontuação. Essas abordagens serão descritas a seguir com mais detalhes.

5.1 Teste de Independência por Probabilidade Conjunta

Duas variáveis são consideradas independentes se a sua probabilidade conjunta é igual ao conjunto de suas probabilidades marginais [4]. Resumindo brevemente o que foi abordado na Seção 2.1.1:

- Diz-se que duas variáveis aleatórias A e B são independentes se P(A | B) = P(A), ou seja, a probabilidade de A ocorrer dado B, é a mesma probabilidade marginal de A. Então conclui-se que B não altera A e portanto são independentes. Por simetria, se A é independente de B então B é independente de A também. Esse é um caso de independência de cardinalidade 0, pois o conjunto separador é de tamanho vazio.
- Dado uma variável C, as variáveis A e B são condicionalmente independentes se P(A,B|C) = P(A,B,C)/P(C). Ou seja, dado o valor de C, a informação (valor) de B não altera a probabilidade de A. Esse é um caso de independência de cardinalidade 1, pois o conjunto separador é de tamanho 1 (variável C).
- Para cardinalidade 2, pode-se deduzir que as variáveis A e B são condicionalmente independentes de C e D se: P(A,B|C,D) = P(A,B,C,D)/P(B,C,D)
- Para a prova destes teoremas e mais informações pode-se consultar a Seção 2.1.1 e as referências [4,53,41].

Os testes citados acima são os testes de independência por probabilidade conjunta. Este método, por ser um dos mais tradicionais, foi um dos métodos implementados neste trabalho.

5.2 Teste de Independência por Pontuação

Um dos métodos de obtenção de independência condicional é o baseado em testes de pontuação, como o implementado pela ferramenta Weka [79]. De acordo com [70], para testar se duas variáveis X e Y são independentes, dado um conjunto de variáveis Z, são comparadas a pontuação de duas Redes Bayesianas: uma com e a outra

sem um arco direcionando Y a X, onde todos os nós em Z são parentes de X. Como métrica de avaliação de pontuação, pode ser usada a *Minimum Description Language (MDL)* ou uma métrica bayesiana como a utilizada pelo algoritmo K2 [11]. Para se encontrar a estrutura da rede, pode-se usar algum algoritmo de aprendizado de Redes Bayesianas. Na implementação usada pelo Weka, foi utilizado o algoritmo IC[80] para geração da rede.

Este método pode apresentar um desempenho computacional melhor que os testes estatísticos clássicos e apresenta resultados (as independências encontradas) consistentes. Este foi o método utilizado para a geração de listas de independência usadas nos testes de avaliação dos algoritmos PC e MarkovPC no Capítulo 4. Para mais detalhes sobre esse método deve-se consultar a referência [70].

5.3 Experimentos e Resultados

Foram conduzidos experimentos e seus resultados analisados a fim de se avaliar o desempenho deste método de IC usando Markov Blanket. A metodologia dos experimentos usada foi:

1. Gerar uma lista de independências usando o método aqui proposto e como teste de IC utilizando tanto a probabilidade conjunta como o baseado em pontuação.
2. Avaliar a taxa de classificação das bases de dados usando as listas geradas no passo 1 ao se executar um algoritmo de aprendizado. Os algoritmos de aprendizado utilizados foram o PC e MarkovPC.

O objetivo desses testes é avaliar como o método se comporta de acordo com o método de IC utilizado e se as independências encontradas não prejudicam o desempenho de classificação dos algoritmos de aprendizado. Vale mais uma vez notar que o método de IC aqui proposto é voltado para tarefas de classificação.

As bases de dados escolhidas foram algumas já utilizadas pelo MarkovPC, mais especificamente: Synthetic1, Synthetic2, Synthetic3, Patient, Solar Flare 1, Solar Flare 2 e Car. Algumas bases não foram escolhidas devido a restrições na ferramenta de software utilizada. A Tabela 5.2 mostra os resultados obtidos. Na Tabela 5.2, cada linha para cada rede expressa o resultado obtido por

um diferente método de IC. Os nomes dos métodos de IC foram abreviados, e seu mapeamento é como se segue:

- Score: teste de independência por pontuação, conforme item 5.2.
- ScoreMB: teste Score mas otimizado usando o conceito de MB.
- Prob: teste de independência por probabilidade conjunta, conforme item 5.1.
- ProbMB: teste Prob mas otimizado usando o conceito de MB.

De acordo com os resultados obtidos nos domínios selecionados, conforme a Tabela 5.2, houve um significativo ganho de desempenho computacional utilizando a estratégia baseada em MB. Na base Car, o esforço dos testes de IC baseados em MB foi o mesmo que os testes de IC tradicionais, mas em todas as outras bases o teste de IC proposto neste trabalho obteve um desempenho fortemente superior, em alguns casos chegando a ganhos superiores a 90% no esforço. O esforço foi medido considerando-se o número de testes de IC executados, e quanto menor, menos testes e consequentemente menor esforço para se executar o processo de obtenção de independências.

Analisando-se qual foi o impacto dessa abordagem (teste de IC usando conceito de MB) sobre os resultados de classificação, pode-se observar que a taxa de classificação não melhorou considerando-se métodos baseados em pontuação. Um dos motivos para tal resultado é que esses métodos de pontuação, como o descrito no item 5.2, baseam-se fortemente no uso de uma Rede Bayesiana inicial, e a partir dela os relacionamentos vão sendo alterados e então calculados os *scores*. Na abordagem usada para estes testes a rede inicial não foi gerada, e para cada teste de IC uma rede era considerada para verificar se as variáveis eram independentes, o que acabou reduzindo o desempenho deste método. Considerando-se os testes de probabilidade conjunta, a média das taxas de classificação foi semelhante, até ganhando em alguns casos e perdendo em outros.

Uma outra importante observação nestes resultados é sobre o esforço dos algoritmos. A abordagem de testes de IC otimizados usando MB não conseguiu reduzir o esforço de indução das redes na maioria dos casos quando considera-se o algoritmo PC. Uma vez que vários testes de IC foram eliminados devido à otimização, algumas

independências acabaram não sendo encontradas. Essas independências que ficaram ausentes teriam eliminado mais arcos durante a execução do PC e assim diminuindo o seu esforço. Entretanto, o algoritmo MarkovPC não foi influenciado, na maioria dos casos, negativamente no seu esforço nos métodos de IC usando MB pois o PC é mais sensível às independências pois sempre considera todas as variáveis na indução (enquanto o MarkovPC elimina variáveis durante a indução).

5.4 Conclusões

De maneira geral pode-se concluir que os métodos de IC usando o conceito de MB realmente tendem a diminuir fortemente o esforço computacional desses testes, entretanto essa otimização de desempenho pode prejudicar o resultado de classificação dos algoritmos devido a diminuição nas independências. Entretanto, alguns casos onde o processo de obtenção de independência pode ser computacionalmente inviável, devido a um grande número de atributos na base ou necessidade de testes com cardinalidade maiores, por exemplo, essa otimização pode tornar viável esse processo. Ou seja, mesmo que em alguns casos possa reduzir o desempenho de classificação, pode viabilizar o método de aprendizado por IC.

Tabela 5.2. Resultados do método de IC usando uma estratégia baseada em MB

REDE	Esforço		Taxa Classificação		Esforço IC	Nro. Nós	|MB|	
	PC	MarkovPC	PC	MarkovPC				
Synthetic1	1036	822	89,160	89,160	24415	32	1	Score
	8373	1085	76,300	79,920	499			ScoreMB
	908	737	50,660	50,660	25224			Prob
	1936	789	87,620	89,020	820			ProbMB
Synthetic 2	710	642	93,200	93,200	20667	32	16	Score
	1120	546	93,200	93,200	526			ScoreMB
	652	577	93,200	50,760	15474			Prob
	859	537	93,200	93,200	496			ProbMB
Synthetic 3	726	634	88,000	84,840	20480	32	31	Score
	1273	548	85,880	85,180	496			ScoreMB
	624	610	50,540	50,540	13533			Prob
	834	535	83,420	83,420	496			ProbMB
Patient	43	63	71,111	71,111	62	9	N/A	Score
	51	53	71,111	71,111	36			ScoreMB
	546	76	61,111	63,333	1506			Prob
	546	76	61,111	61,111	36			ProbMB
Solar-Flare 1	211	215	73,684	74,302	842	13	N/A	Score
	758	311	71,207	71,826	340			ScoreMB
	1905	325	60,681	66,563	8917			Prob
	1928	165	58,823	59,133	578			ProbMB
Solar-Flare 2	231	247	76,360	76,078	1075	13	N/A	Score
	866	244	73,170	71,669	193			ScoreMB
	1550	155	69,793	72,514	7822			Prob
	1550	155	70,450	70,262	2138			ProbMB
Car	50	68	81,134	81,134	115	7	N/A	Score
	76	34	88,831	93,800	21			ScoreMB
	99	34	70,023	70,023	171			Prob
	99	34	70,023	70,023	171			ProbMB

6 CONCLUSÕES

Este trabalho propôs e discutiu um novo algoritmo de aprendizado de Classificador Bayesiano baseado em Independência Condicional, batizado de MarkovPC, para induzir Classificadores Bayesianos. Em vez de se usar o conceito de Markov Blanket para selecionar atributos após a indução da Rede Bayesiana, como feito em outros trabalhos discutidos na literatura, o MarkovPC usa o conceito de Markov Blanket para impor algumas restrições e otimizar o algoritmo PC tradicional durante a indução do classificador. É importante ressaltar que o MarkovPC é designado especificadamente para tarefas de classificação.

Experimentos em diversos domínios foram realizados e analisados, revelando que o MarkovPC tende a ser mais preciso (em termos de taxas de classificação) do que o PC tradicional e do que o NaiveBayes. Além disso, o MarkovPC produz classificadores mais simples que demandam menos testes de comparação durante o processo de aprendizagem do que o PC. O Markov Blanket da Classe aproximado, gerado pelo Markov PC, é consistente com o MBC do PC. Desta forma, o MarkovPC pode ser considerado consistente e promissor.

A mesma idéia baseada no conceito de Markov Blanket pode ser usada durante a geração das listas de independências nos dados usando-se testes de independência condicional. Os testes de independência são testes que exigem grande esforço computacional, uma vez que podem ser exponenciais em relação ao número de atributos. Foram realizados experimentos em diversos conjuntos de dados e pôde-se concluir que houve contribuição nesta área também: os testes de independência utilizando o conceito de MB tiveram um esforço computacional significativamente menor e geraram listas de independência consistentes em diversos domínios.

Pode-se concluir, então, que para tarefas de classificação o uso do conceito de Markov Blanket otimiza, tanto em esforço computacional como em desempenho de classificação, o processo de indução de um Classificador Bayesiano baseado em independência condicional bem como a utilização desse classificador.

6.1 Trabalhos Futuros

Outros métodos de independência condicional, como alguns descritos em [51,67,69, 76,77,51], por exemplo, podem ser implementados e o impacto de seus usos, tanto no MarkovPC quanto no teste de independência baseado em MB, serem avaliados. Ainda nos testes de independência usando MB, seria possível avaliar outros métodos de identificação de Markov Blanket, como, por exemplo, a partir de uma rede markoviana conforme apresentado em [71].

A idéia do uso de uma rede markoviana poderia ser usada também durante a indução do classificador, considerando-se que, até o momento que a rede seja direcionada, ela é uma rede markoviana. Desta forma, poderia ser utilizado o conceito de Markov Blanket markoviano e assim as variáveis que não fossem vizinhas à classe seriam eliminadas durante a indução da rede.

Além disso, outros algoritmos de aprendizado baseados em independência condicional, entre eles o clássico IC[80], o SGS[23] e o Grow Shrink (GS)[78], podem ser implementados seguindo a mesma idéia já definida no MarkovPC. Por fim, com estudos empíricos mais robustos poderia ser possível identificar com mais precisão qual o grau de redução a nova metodologia traz e o impacto na qualidade das redes construídas. Desta forma poderia ser generalizado um processo de aprendizado de independência condicional baseado no conceito de Markov Blanket.

6.2 Publicações

Até o momento, foram publicados os seguintes artigos, referentes de alguma forma aos métodos desenvolvidos neste trabalho:

- GALVÃO, S. D. C. O., HRUSCHKA JR., E. R. A Markov Blanket based strategy to optimize the induction of Bayesian Classifiers when using Conditional Independence Learning Algorithms. In: 9th International Conference on Data Warehousing and Knowledge Discovery (DAWAK), Germany, 2007 (artigo selecionado entre os 5 melhores da conferência e convidado a ser extendido para um periódico).
- HRUSCHKA JR., E.R., SANTOS, E. B., GALVÃO, S. D. C. O. An Optimized Evolutionary Conditional Independence Bayesian Classifier

Induction Process. In: special issue of the International Journal on Neural and Mass-Parallel Computing and Information Systems, 2007.

- HRUSCHKA JR., E.R., SANTOS, E. B., GALVÃO, S. D. C. O. Variable Ordering in the Conditional Independence Bayesian Classifier Induction Process: An Evolutionary Approach. In: 7th International Conference on Hybrid Intelligent Systems, Germany, 2007.
- VIVENCIO, D. P., HRUSCHKA JR., E. R., NICOLETTI, M. C., SANTOS, E. B. and GALVÃO, S. D. C. O., Feature-weighted k-Nearest Neighbor Classifier, IEEE Symposium on Foundations of Computational Intelligence (FOCI'07), USA, 2007.

7 REFERÊNCIAS

[1] MENA, J. Data mining your website. Digital Press, 1999.

[2] FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery: an overview. In: ADVANCES IN KNOWLEDGE DISCOVERY AND DATA MINING, 1 - 34. AAAI Press, Menlo Park, CA, 1996.

[3] HECKERMAN, D. Bayesian networks for data mining. Data Mining; Knowledge Discovery Journal, 1, 79-119, 1997.

[4] PEARL, J., Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, 1988.

[5] HECKERMAN, D., A tutorial on learning bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, Advanced Technology Division, Microsoft Corporation, 1995.

[6] HRUSCHKA Jr., E. R., Propagação de Evidências em Redes Bayesianas: Diagnóstico sobre Doenças Pulmonares. Dissertação de mestrado, Universidade de Brasília, Departamento de Ciência da Computação, Brasília, 1997.

[7] COZMAN, F. G., Generalizing Variable Elimination in Bayesian Networks, Workshop on Probabilistic Reasoning in Artificial Intelligence, 27-32, Atibaia, Brazil, 2000.

[8] CASTILLO, E., GUTIERREZ, J.; HADI, A., Expert Systems and Probabilistic Network Models. New York: Springer-Verlag, 1996.

[9] CHOW, C. K.; LIU, C. N., Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory, 14(3), 462-467, 1968.

[10] REBANE, G. ; PEARL, J., The recovery of causal ploy-trees from statistical data. In Proc. of Workshop on Uncertainty in Artificial Intelligence, pages 222--228, Seattle, 1987.

[11] COOPER, G.; HERSKOVITZ, E., A Bayesian Method for the Induction of Probabilistic Networks from Data. Machine Learning Journal, volume 9, number 4, 309-347, 1992.

[12] HECKERMAN, D., GEIGER, D.; CHICKERING, D., Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning Journal, volume 20, number 3, 197-243, 1995.

[13] LAM, W.; BACCHUS, F., Learning Bayesian Belief Networks, Na Approach based on the MDL principle. Computational Intelligence, 10, 269-293, 1994.

[14] ACID, S.; de CAMPOS, L. M., Searching for Bayesian Network Structures in the Space of Restricted Acyclic Partially Directed Graphs. Journal of Artificial Intelligence Research, 18, 445-490, 2003.

[15] DE CAMPOS, L. M., FERNADEZ-LUNA, J. M.; PUERTA, J. M., Local Search Methods for Learning Bayesian Networks Using a Modified Neighborhood in the Space of DAGs. In Lecture Notes in Artificial Intelligence, Volume 2527: Proceedings of the Eight Ibero-American Conference on AI, Eds. F. J. Garijo; J. C. Riquelme and M. Toro, 182-192, 2002.

[16] HRUSCHKA JR., E. R.; EBECKEN, N. F. F., Variable Ordering for Bayesian Networks Learning from Data In: International Conference on Computational Intelligence for Modelling, Control and Automation - CIMCA'2003, Vienna, 2003.

[17] HRUSCHKA JR., E. R., HRUSCHKA, E. R.; EBECKEN, N. F. F., Feature Selection by Bayesian Networks. In: The Seventeenth Canadian Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence. Berlin: Springer-Verlag, 2004. v.3060. p.370 – 379, 2004.

[18] CHICKERING, D. M., Optimal Structure Identification with Greedy Search. Journal of Machine Learning Research, 3, 507-554, 2002.

[19] DE CAMPOS, L. M., Independency relationships and learning algorithms for singly connected networks. Journal of Experimental and Theoretical Artificial Intelligence, 10:511-549, 1998.

[20] DE CAMPOS, L. M.; HUETE, J. F., A new Approach for learning belief networks using independence criteria. International Journal of Approximate Reasoning Vol. 24, 11-37, 2000.

[21] CHENG, J., BELL, D. A.; LIU, W., An algorithm for Bayesian belief network construction from data. Proc. Al & STAT'97, Ft. Lauderdale, FL, 83-90, 1997.

[22] VERMA, T.; PEARL, J., Equivalance and synthesis of causal models. Proceedings of the Sixth Conference in Artificial Intelligence, Mountain View, CA, 220-227, 1991.

[23] SPIRTES, P., GLYMOUR, C.; SCHEINES, R., Causation, Predication, and Search. Springer-Verlag, New York, 1993.

[24] SPIRTES, P.; GLYMOUR, C., An algorithm for fast recovery of sparse causal graphs, Social Science Computer Review, 9, 62-72, 1991.

[25] DE CAMPOS, L. M. and HUETE, J. F., A new approach for learning belief networks using independence criteria. International Journal of Approximate Reasoning, 24(1), 11-37, 2000.

[26] COWELL, R. G., Conditions Under Which Conditional Independence and Scoring Methods Lead to Identical Selection of Bayesian Network Models. The Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence, Eds. Jack S. Breese and Daphne Koller and Jack S. Breese and Daphne Koller, 91-97, 2001.

[27] ACID, S., DE CAMPOS, L. M., BENEDICT: An algorithm for learning probabilistic belief networks, in the Proceedings of the IPMU-96 Conference, 979-974, 1996.

[28] DASH, D.; DRUZDZEL, M., A Hybrid Anytime Algorithm for the Construction of Causal Models From Sparse Data. Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence," Morgan Kaufmann Publishers, Inc., San Francisco, 142-149, 1999.

[29] SINGH, M.; VALTORTA, M., Construction of Bayesian Network Structures From Data: A Brief Survey and an Efficient Algorithm International Journal of Approximate Reasoning, 12 (2), 111-131, 1995.

[30] ACID, S.; DE CAMPOS, L. M., A hybrid methodology for learning belief networks: BENEDICT. International Journal of Approximate Reasoning, 27, 235-262, 2001.

[31] LIU, H.; MOTODA, H., Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers, 1998.

[32] REUNANEN, J., Overfitting in Making Comparisons Between Variable Selection Methods, Journal of Machine Learning Research 3, 1371-1382, 2003.

[33] LANGLEY, P.; SAGE, S., Induction of selective bayesian classifiers. Proceedings of the tenth conference on uncertainty in artificial intelligence. Morgan Kauffman, Seattle, 1994.

[34] SINGH, M; PROVAN, G. M., A comparison of induction algorithms for selective and non-selective bayesian classifiers. Proceedings of the 12th International conference on machine learning, 497-505, 1995.

[35] BASAK, J., SUDARSHAN, A., TRIVEDI, D.; SANTHANAM, M.S., Weather Data Mining Using Independent Component Analysis, Journal of Machine Learning Research, n.5, 239-253, 2004.

[36] GUYON, I.; ELISSEEFF, A., An Introduction to Variable and Feature Selection, Journal of Machine Learning Research 3, 1157-1182, 2003.

[37] HRUSCHKA JR., E. R., HRUSCHKA, E.R.; EBECKEN, N.F.F., Applying Bayesian Networks for Meteorological Data Mining. Proceedings of the The Twenty-fifth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence. Cambridge, 122-133, 2005.

[38] HRUSCHKA JR., E. R., HRUSCHKA, E. R.; EBECKEN, N. F. F. A feature selection Bayesian approach for a clustering genetic algorithm. In: Data Mining 2003, ed. Southampton, 181-192, 2003.

[39] KOLLER, D.; SAHAMI, M., Toward optimal feature selection, Proceedings of the 13th International Conference on Machine Learning, 284-292, July, 1996.

[40] HRUSCHKA JR, E. R., HRUSCHKA, E. R., EBECKEN, N. F. F., A Feature Selection Bayesian Approach for Extracting Classification Rules with a Clustering Genetic Algorithm. Applied Artificial Intelligence. London: , v.17, n.5-6, 489 - 506, 2003.

[41] NEAPOLITAN, R. E., Learning Bayesian Networks. Prentice Hall, 2003.

[42] PEARL, J., Causality: Models, Reasoning, and Inference. Cambridge University Press, 2000.

[43] CHENG, J., BELL, D., LIU, W. Learning bayesian networks from data: an efficient approach based on Information Theory. Technical Report in Expert Systems and probabilistic network models, New York, Springer, 1997.

[44] LIU, H.; YU, L., Feature selection for data mining. http://www.public.asu.edu/~huanliu/feature_selection.html, 2002.

[45] KOHAVI, R.; JOHN, G. H., Wrappers for feature subset selection. Artificial Intelligence, 97 (1-2): 273-324, 1997.

[46] LEE, H. D., Seleção de atributos importantes para a extração de conhecimento de bases de dados. Tese de doutorado, Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, São Carlos, 2005.

[47] VIVENCIO, D. P., HRUSCHKA JR., E. R., NICOLETTI, M. C., SANTOS, E. B.; GALVÃO, S. D. C. O., Feature-weighted k-Nearest Neighbor Classifier, IEEE Symposium on Foundations of Computational Intelligence (FOCI'07), 2007.

[48] FUYAN, L., An Attribute Selection Approach and Its Application. Neural Networks and Brain, ICNN&B '05, International Conference on, volume 2, 636-640, 2005.

[49] LAURITZEN, S. L., DAWID, A. P., LARSEN, B. N.; LEIMER, H. S., Independence properties of directed Markov fields. Networks, 20:491-505, 1990.

[50] ACID, S.; DE CAMPOS, L. M., An algorithm for finding minimum d-separation sets in belief networks. In Proceedings of the twelfth Conference of Uncertainty in Artificial Intelligence, 1996.

[51] SILVA, W. T., Algoritmo d-separador em rede Bayesiana e separador em grafo não orientado. Relatório de Pesquisa, CIC/UnB 2002, ftp://ftp.cic.unb/pub/cic/wagner/relatorios/d-separador.zip, 2002.

[52] AHUJA, R. K., MAGNANTI, T. L.; ORLI, J. B., Network Flows: Theory, Algorithms and Applications. Englewood Cliffs, New Jersey: Prentice Hall, 1993.

[53] SILVA, W. T; LADEIRA, M., Mineração de Dados em Redes Bayesianas. Em PORTO, I. J.; CUSTÓDIO, R. Jornada de Atualização em Informática, capítulo 6. Florianópolis: Editora SBC, 2002.

[54] MEGANCK, S., MAES, S., LERAY, P.; MANDERICK, B., Learning semi-markovian models using experiments. The Third European Workshop on Probabilistic Graphical Models, 12-15, PGM'06, 2006.

[55] DRUZDZEL, M. J., SMILE: Structural modeling, inference, and learning engine and GeNIe: A development environment for graphical decision-theoretic models. In: Proc. of the Sixteenth National Conference on Artificial Intelligence, Orlando, FL, pp. 902-903, 1999.

[56] BEINLICH, I., SUERMONDT, H. J., CHAVEZ, R. M.; COOPER, G. F., The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In: Proc. of the Second European Conf. on Artificial Intelligence in Medicine, London, UK, 247-256, 1989.

[57] DUDA, R. O. ; HART, P. E., Pattern Classification and Scene Analysis. New York, John Wiley & Sons, 1973.

[58] LANGLEY, P., IBA, W.; THOMPSON, K., An analysis of Bayesian classifiers. In Proceedings, Tenth National Conference on Artificial Intelligence (pp. 223–228). Menlo Park, CA: AAAI Press, 1992.

[59] MICHIE, D., SPIEGELHALTER, D. J.; TAYLOR, C. C., Machine learning, neural and statistical classification, (edited collection). New York: Ellis Horwood, 1994.

[60] SPIRTES, P.; MEEK, C., Learning Bayesian networks with discrete variables from data. KDD95, pp. 294-299, 1995.

[61] Bellman, R., Adaptive Control Processes: A Guided Tour. Princeton University Press, 1961.

[62] BLAKE, C. L.; MERZ, C. J, UCI Repository of Machine Learning Databases, Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[63] BOHANEC, M.; RAJKOVIC, V., Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990.

[64] SHAPIRO, A. D., Structured Induction in Expert Systems. Addison-Wesley, 1987.

[65] HONG, Z.Q.; YANG, J.Y., Optimal Discriminant Plane for a Small Number of Samples and Design Method of Classifier on the Plane. Pattern Recognition, Vol. 24, No. 4, pp. 317-324, 1991.

[66] BUDIHARDJO, A., GRZYMALA-BUSSE, J.; WOOLERY, L., Program LERS_LB 2.5 as a tool for knowledge acquisition in nursing. Proceedings of the 4th Int. Conference on Industrial & Engineering Applications of AI & Expert Systems, pp. 735-740, 1991.

[67] HAYS, W., Statistics. Wadsworth Publishing, 5th edition, 1994.

[68] FRIEDMAN, N., GEIGER, D.; GOLDSZMIDT, M., Bayesian network classifiers. Machine Learning, 19(4):131-163, 1997.

[69] AGRESTI, A., Categorical Data Analysis. Wiley, 2nd edition, 2002.

[70] BOUCKAERT, R. R., Bayesian network classifiers in Weka. Working Paper 14/2004, Department of Computer Science, University of Waikato, New Zealand, 2004.

[71] BROMBERG, F., MARGARITIS, D.; HONAVAR, V.,. Efficient Markov Network Structure Discovery from Independence Tests. Proceedings of The Sixth SIAM International Conference on Data Mining (SDM), 20-22, 2006

[72] BINS, J., DRAPER, B., Feature selection from huge feature sets. International Conference on Computer Vision, volume 2, pages 159-165, Vancouver, Canada, 2001.

[73] DAS, S., Filters, wrappers and a boosting based hybrid for feature selection. 8th Int. Conf. on Machine Learning, pages 74-81, Williams College, 2001.

[74] DASH, M., LIU, H., Hybrid search of feature subsets. Pacific Rim International Conference on Artificial Intelligence, pages 238-249, 1998.

[75] BARANAUSKAS, J. A., Extração Automática de Conhecimento por Múltiplos Indutores. Tese de Doutorado, ICMC-USP, 2001.

[76] YAO, Q.; TRITCHLER, D., An exact analysis of conditional independence in several 2 X 2 contingency tables. Biometrics, 49(1):233-236, 1993.

[77] LIANGJUN, S.; HALBERT, W., Testing Conditional Independence Via Empirical Likelihood. Department of Economics, UCSD. Paper 2003-14 disponível em http://repositories.cdlib.org/ucsdecon/2003-14, 2003.

[78] MARGARITIS, D.; THRUN, S., Bayesian network induction via local neighborhoods. Advances in Neural Information Processing Systems 12 (NIPS), pp 505-511, MIT Press, 1999.

[79] WITTEN, I. H.; EIBE, F., Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[80] VERMA, T.; PEARL, J., An algorithm for deciding if a set of observed independencies has a causal explanation. Proc. of the Eighth Conference on Uncertainty in Artificial Intelligence, 323-330, 1992.

[81] PEARL, J., Causality: Models, Reasoning, and Inference. Cambridge University Press, 2000.