Universidade do Porto
Faculdade de Economia

Integração de termos frequentes na indução de árvores de decisão

Francisco Miguel Saramago Rodrigues

Dissertação para a obtenção do grau de Mestre em
Análise de Dados e Sistemas de Apoio à Decisão

Orientação: Prof. Doutor João Gama

Outubro de 2003

Nota biográfica

Francisco Miguel Saramago Rodrigues é natural de Santa Maria da Feira onde residiu até aos 18 anos. Licenciou-se em Economia pela Faculdade de Economia do Porto em 1995. Começou a sua carreira profissional da unidade fabril da Philips em Ovar no departamento de Marketing. Em 1997 ingressou na Sonae Distribuição, tendo desenvolvido várias funções ligadas área operacional do negócio e área do controlo de gestão.

Agradecimentos

Gostaria de agradecer aos meus pais, irmãs e cunhado, e à minha avó, por todo o carinho e incentivos que me deram ao longo desta caminhada.

Quero também agradecer ao meu orientador, Professor Doutor João Gama, por todos os ensinamentos, colaboração e disponibilidade que sempre demonstrou.

Aos meus amigos, Fernando Oliveira e Miguel Portela, pelos incentivos e apoio que me deram na revisão deste documento.

Um agradecimento especial para o ao meu chefe, Sr. João Melo, por todo apoio e compreensão dados, para que fosse possível terminar esta etapa da minha vida académica.

Resumo

Nesta tese investigamos formas de integração de duas técnicas de extracção de conhecimento, uma técnica descritiva, a descoberta de regras de associação, e outra técnica de previsão, a aprendizagem de árvores de decisão. O principal objectivo consiste em integrar estas duas técnicas com base numa metodologia de generalização em cascata e, obter modelos menos complexos e com maior poder de previsão do que os modelos de árvores de decisão produzidos pelos algoritmos convencionais.

O método que propomos consiste na execução sequencial de um algoritmo de aprendizagem de regras de associação seguido de um algoritmo de árvores de decisão. Dado um problema concreto de classificação, o algoritmo de aprendizagem de regras de associação encontra todos os conjuntos de termos frequentes existentes nos dados, que tomam a forma de $atributo = valor$. Os conjuntos de termos frequentes mais interessantes são depois seleccionados recorrendo a uma heurística de ordenação. Após esta operação, estes conjuntos são projectados sobre o conjunto de dados sob a forma de atributos binários. Cada conjunto de termos frequente dá origem a um novo atributo. Para cada instância do conjunto de treino e do conjunto de teste, o novo atributo tomará o valor *1* se e só se, a conjunção de pares $atributo = valor$ se verificar para a instância em concreto. O algoritmo de aprendizagem de árvores de decisão gera um modelo a partir do conjunto de treino estendido pelos novos atributos. Se alguns dos novos atributos forem seleccionados para um teste num nó de decisão, então a linguagem de representação da árvore de decisão será estendida, incorporando também a linguagem de representação das regras de associação.

A avaliação empírica deste método permite concluir que os modelos gerados têm um maior nível de exactidão do que os gerados pelos métodos convencionais e são menos complexos, no respeita ao número de nós existentes na árvore de decisão. Foram experimentadas várias heurísticas para ordenação dos conjuntos de termos frequentes encontrados. Em todas, foram obtidos bons resultados, sendo os mais promissores aqueles conseguidos com o rácio de informação.

Abstract

In this thesis we investigate ways to integrate two different data mining techniques, one is descriptive, the association rules discovery, and the other is predictive, the decision tree learning. The main objectives are integrating the two techniques using a cascade generalization methodology and obtaining less complex models with better prediction capability than those produced by conventional decision tree algorithms.

We propose a method that consists in the sequential application of discovery association rules algorithm and learning decision tree algorithm. Given a certain classification problem, the algorithm of discovery association rules finds all the item sets available on data, which are represented by *attribute=value.* The most interesting item sets are selected using an ordering heuristic. Afterwards they are projected on the data set as binary attributes. Each instance of the training and testing set will assume the value one if and only if, the pair conjunction *attribute=value* occurs for that instance. A model is generated by the algorithm of decision tree learning. If the new attributes are selected for a decision node test, the decision tree representation language will also be extended, thus incorporating the association rules representation language.

The empirical evaluation of this methodology proves that the generated models are more exact than those generated by conventional methods of decision trees. Furthermore, they are less complex. Several ordering heuristics were used in order to select item sets. Although good results were obtained in all heuristics, the most promising were those obtained through the information ratio.

Índice

Índice de figuras

Índice de tabelas

1 Introdução

A descoberta de conhecimento a partir de bases de dados (*Kownledge Discovery Databases*) é um ramo multidisciplinar que resultou de contribuições da estatística, da aprendizagem automática e de bases de dados. Esta área tem despertado um crescente interesse devido ao aumento exponencial de dados armazenados e à necessidade económica e científica de extrair informação. O processo de automação da aquisição de conhecimento é um dos factores que tornam a aprendizagem automática uma área de investigação muito activa.

As diferentes fases que conduzem à extracção de conhecimento contido nas bases de dados foram identificadas por Fayyad *et al*. (1996). Este processo, ilustrado na Figura 1.1, inicia-se pela selecção dos dados a analisar, seguido pela sua limpeza, transformação e redução. O processo de data mining, definido como um procedimento automático de extracção de padrões úteis, não triviais e inteligíveis, constitui a fase seguinte. Segue-se a avaliação dos seus resultados e a definição de um conhecimento possível da base de dados.

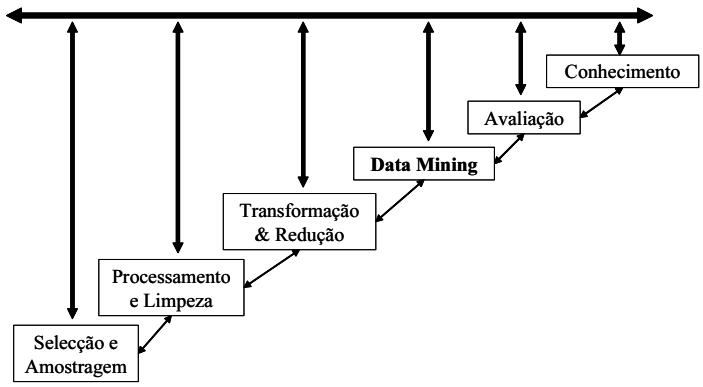


Figura 1.1: Processo de descoberta de conhecimento

Espera-se que através de um modelo de inferência e de um conjunto dados, se possa, com acesso a recursos computacionais, obter uma representação de conhecimento contido nos dados. Este conhecimento é útil para efeitos de diagnóstico,

modelação, previsão ou na descoberta de relações existentes nas bases de dados. O seu reconhecimento manual seria extremamente difícil dada a sua complexidade.

1.1 Conceitos fundamentais

A aprendizagem indutiva consiste em construir teorias gerais a partir de um conjunto limitado de factos. Num problema típico, os factos são apresentados sob a forma de um conjunto de exemplos, também designados de instâncias, indivíduos ou casos, e o objectivo é o de produzir a descrição de um conceito, teoria ou hipótese, que generalize os factos. A descrição do conceito é o resultado da aprendizagem e deve ser inteligível, na medida, em pode ser apreendida, discutida e aplicado aos actuais exemplos.

A informação fornecida a um sistema de aprendizagem toma a forma de um conjunto de instâncias. Cada instância representa um exemplo do conceito a ser apreendido e é caracterizada pelos valores de um conjunto de atributos pré--determinados. Cada atributo toma os valores num domínio. Em problemas reais, um exemplo pode conter atributos de natureza diferente. Neste estudo, a classificação dos atributos quanto ao tipo de valores que assumem é feita da seguinte forma:

- Atributos nominais, podem assumir um conjunto limitado de valores diferentes;
- Atributos binários, tomam um de dois valores possíveis;
- Atributos contínuos, contêm valores numéricos, sendo possível estabelecer relações de ordem.

No contexto desta aprendizagem, é usual distinguir entre as técnicas de previsão e descritivas. Enquanto nas primeiras o objectivo é prever uma variável, ou atributo, através de um modelo, nas segundas, o objectivo é descrever as relações existentes entre as variáveis do conjunto de dados.

Nas técnicas de previsão, é fornecido um conjunto de exemplos onde o valor da variável a prever é conhecido. A partir dos exemplos do conjunto de treino é construído

um modelo que poderá ser utilizado para prever o valor da variável objectivo, em exemplos para os quais o valor desta variável é desconhecido. No contexto desta técnica podemos encontrar dois tipos de aprendizagem: a de classificação, se a variável a prever contiver um conjunto limitado de categorias; e a de regressão se a variável a prever for contínua. A aprendizagem de classificação é muitas vezes designada de supervisionada (Witten e Frank, 1999).

Nas técnicas descritivas o objectivo é descobrir estruturas interessantes existentes nos dados. Na aprendizagem de regras de associação são procuradas quaisquer relações existentes nos dados e não apenas aquelas que prevêem um valor particular da classe. E a aprendizagem de agrupamentos (*clustering*) é usada para agrupar termos que naturalmente estão associados (Witten e Frank, 1999).

Um algoritmo de aprendizagem de conceitos consiste num conjunto de procedimentos que visam a construção de um modelo. Segundo Fayyad *et al.* (1996), aquele é caracterizado por três componentes:

(i) Representação, que é a linguagem formal de descrição dos conceitos;

(ii) Procura, que é a forma como o algoritmo encontra a descrição do conceito objectivo;

(iii) Componente de avaliação, mede a qualidade das várias descrições de conceitos, e é usada simultaneamente para guiar a procura e decidir quando é que esta deve parar.

O modelo resultante da aplicação do algoritmo de aprendizagem permite obter o conceito objectivo para cada exemplo que lhe seja submetido. Na Figura 1.2 estão descritas as várias componentes de um sistema de aprendizagem, tal como foram descritas.

Os modelos induzidos podem ser aplicados para prever a classe. Para cada exemplo que lhe seja submetido, o modelo consegue inferir o valor da classe. Esta fase é descrita pela Figura 1.2 como a componente de performance do sistema de aprendizagem.

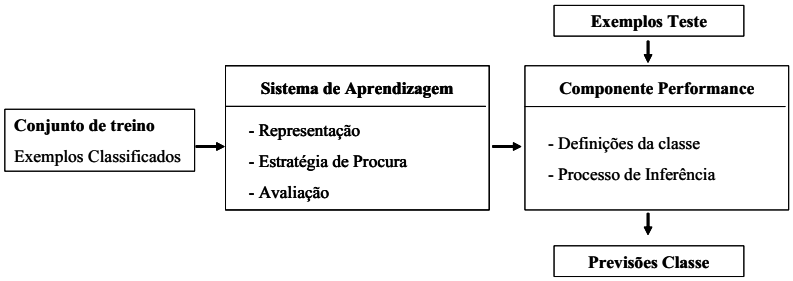


Figura 1.2: Componentes de um sistema de aprendizagem de conceitos

Para induzir uma teoria a partir de um conjunto de treino, um sistema de aprendizagem assume determinados pressupostos, que são designados de viés (Mitchell, 1997). O viés pode ser dividido em absoluto ou relativo. O primeiro está ligado à descrição de linguagem das teorias usadas, o viés da linguagem, e restringe os domínios das teorias que podem ser, expressas e apreendidas, pelo sistema de aprendizagem. O segundo é o resultado da ordenação das hipóteses.

Um algoritmo de aprendizagem que não recorra a quaisquer pressupostos tem uma reduzida probabilidade de gerar uma teoria útil, pois existe um elevado número de teorias consistentes com o conjunto de treino, e ao mesmo tempo não existe qualquer indicação sobre qual delas a melhor (Mitchell, 1997). Assim, os sistemas de aprendizagem recorrem ao viés para reduzir o amplo espaço de hipóteses possíveis a um conjunto de hipóteses preferenciais.

1.2 Motivação

Nesta tese estudamos formas de integrar os dois tipos de técnicas tendo como principal objectivo a construção de modelos com maior poder de previsão. Vamos utilizar dois tipos de técnicas de aprendizagem, uma descritiva, a descoberta de regras de associação, e outra de previsão, a aprendizagem de regras de associação.

Por um lado, os algoritmos de descoberta de regras de associação são compostos por duas fases, a extracção de conjuntos de termos frequentes e geração de regras de associação. Na primeira fase é executada uma procura completa do espaço e são encontrados os conjuntos de termos frequentes. Estes conjuntos tomam a forma de

conjunções $atr_i = valor$ e representam a co-ocorrência simultânea de conjunções de valores de diferentes atributos. Na segunda fase, são geradas as regras de associação.

Por outro lado, os algoritmos de aprendizagem de árvores de decisão fazem uma procura heurística do espaço de hipóteses, em cada nó de decisão é seleccionado um teste no valor de um atributo assumindo que todos os atributos são independentes. Uma vez instalado um teste, os exemplos de treino são divididos e o processo é repetido para cada subgrupo de exemplos. Cada percurso desde a raiz da árvore até uma das folhas define uma regra de decisão. As regras de decisão tomam a forma, *se* $atr_i = a \wedge atr_j = b$ *então* $classe = c$, e cada uma das condições representa um teste num nó de decisão.

O recurso à combinação das técnicas surge para colmatar as restrições individuais dos algoritmos de aprendizagem. Alguns dos algoritmos de aprendizagem de árvores de decisão testam um único atributo em cada nó decisão. Se consideramos a representação numa árvore de decisão do conceito $(A \wedge B) \vee (C \wedge D)$, assumindo que todos os atributos são binários, verificamos que essa árvore de decisão contém testes repetidos em dois dos seus ramos. Este problema é designado de fragmentação de conceitos e resulta do teste realizado no nó de decisão não reflectir a real distribuição dos registos. A descoberta de regras de associação não é uma técnica que possa ser directamente aplicada a todo o tipo de problemas, é mais apropriada para análise e descrição de estruturas nos dados. Assim, cada uma destas técnicas de aprendizagem possui determinadas características que as tornam mais adequadas para determinadas tarefas em detrimento de outras.

Pretendemos integrar estas duas técnicas de tal forma que, a extracção de conhecimento através das regras de associação possa beneficiar a classificação do conjunto de dados pela árvore de decisão. O sucesso da integração destas técnicas depende da forma como conseguirmos responder a conjunto de questões:

(i) Como integrar o conhecimento dos conjuntos de termos frequentes encontrados pelo algoritmo de aprendizagem de regras de associação nas árvores de decisão?

(ii) Quais os conjuntos de termos frequentes a serem seleccionados?

(iii) Quantos novos atributos deverão ser integrados no conjunto de dados?

O objectivo último é desenvolver uma metodologia que permita gerar modelos que obtenham melhor performance do que aqueles gerados pelos algoritmos convencionais de aprendizagem de árvores de decisão, no que respeita à exactidão das previsões e à complexidade das teorias geradas.

1.3 Principais contribuições

As principais contribuições desta tese são:

i. Uma nova metodologia para combinar regras de associação e árvores de decisão, melhorando os níveis de previsão dos modelos e a sua complexidade;

ii. Uma metodologia para obter modelos sob a forma de árvores de decisão que utilizam conjunções de condições em nós de decisão. Estes modelos representam árvores multivariadas;

iii. Explorar estratégias para a avaliação dos conjuntos de termos frequentes.

1.4 Organização da tese

No segundo capítulo apresentamos uma descrição das áreas relacionadas, as árvores de decisão, as regras de associação, heurísticas para ordenação de regras, os modelos múltiplos e a avaliação de modelos. No terceiro capítulo descrevemos a metodologia proposta para comprovar as hipóteses subjacentes a esta tese: neste contexto, apresentamos o método proposto e descrevemos as etapas necessárias à sua construção. Os resultados experimentais para vários conjuntos de dados são apresentados e discutidos no quarto capítulo. Por último, no quinto capítulo são apresentadas as conclusões do trabalho, bem como indicações para futuros trabalhos.

2 Técnicas de extracção de conhecimento

A integração de modelos envolve a selecção de diferentes técnicas de aprendizagem para tornar a sua combinação possível. Neste capítulo descrevemos as técnicas a que recorremos. Na primeira secção, são abordados os aspectos fundamentais da aprendizagem de árvores de decisão, e na segunda, descrevemos os algoritmos de aprendizagem de regras de associação. As heurísticas para seleccionar os conjuntos de termos frequentes em pós-processamento são descritas na terceira secção. Os modelos múltiplos são abordados na quarta secção. Por último, descreveremos algumas das técnicas mais usadas na avaliação de modelos.

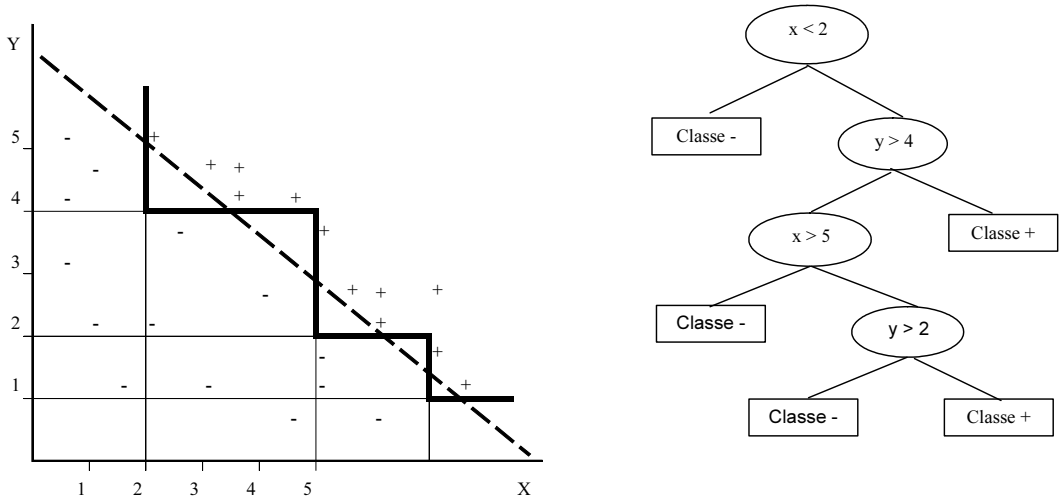
2.1 Árvores de decisão

As árvores de decisão são um dos mais populares métodos de aprendizagem automática, pois adaptam-se a diferentes domínios de dados e são facilmente interpretáveis, quer através da sua representação gráfica, quer pela sua transformação em regras de decisão. São construídas recorrendo a estratégias do tipo "dividir para conquistar", ou seja, um problema complexo é dividido em problemas mais simples. Este processo é repetido até ser encontrada uma solução para o problema.

Uma árvore de decisão pode ser definida por uma folha ou por um nó de decisão. Em cada nó de decisão é realizado um teste e para cada um dos possíveis valores do teste sai um ramo para formar uma nova árvore de decisão. A sequência de nós de decisão desde a raiz da árvore até às folhas forma um caminho e todos os caminhos são mutuamente exclusivos e exaustivos. As regiões definidas pelos caminhos cobrem todo o espaço de instâncias e não existe qualquer sobreposição entre eles. A classificação de uma instância é iniciada na raiz da árvore, seguindo depois o ramo indicado pelo resultado do teste, e assim sucessivamente até que um nó folha seja encontrado. O nome da classe contido no nó folha é o resultado final da classificação.

Uma árvore de decisão pode ser caracterizada relativamente à forma como são testados os atributos num nó de decisão. As que admitem um teste a um único atributo, em cada nó de decisão são designadas de *univariadas* por contraposição às árvores *multivariadas*, cujo teste pode ser representado por uma combinação de vários atributos (Brodley e Utoff, 1992). Esta limitação das árvores *univariadas* condiciona a sua capacidade para representar sucintamente os conceitos e manifesta-se pela existência de testes repetidos ao longo da árvore (Pagallo e Haussler, 1990).

Considerando o espaço bidimensional de instâncias apresentado na Figura 2.1(a) e a correspondente árvore de decisão (b), verificamos que a árvore de decisão *univariada* aproxima-se do hiper-plano formado por $x + y \leq 8$ através de uma série de cortes ortogonais.



a) Espaço de Procura

b) Árvore de Decisão

Figura 2.1: Espaço de instâncias e correspondente árvore de decisão

Na figura 2.1 (a), a linha a tracejado representa a superfície de separação entre as classes (+ e -) e a linha a cheio representa a aproximação feita pela árvore de decisão a essa superfície. O primeiro teste é realizado para o atributo *X* e permite isolar um conjunto de instâncias negativas, todos os valores em que *X* é menor do que *2*. A partir deste teste, é necessário percorrer todo o espaço cujos valores de *X* são superiores a *2*. No nó decisão seguinte, é testado o atributo *Y* com a condição do seu valor ser maior ou

igual a *4*, conseguindo-se assim, isolar instâncias positivas. Este processo é repetido até todo espaço de procura ser todo percorrido.

Para construir árvores de decisão compactas e com bom poder de previsão de exemplos não observados, é crucial a selecção dos testes a realizar em cada nó de decisão. Um bom teste deverá dividir o conjunto de dados em subconjuntos os mais puros possível e que contenham apenas um dos valores do atributo objectivo.

Um problema fundamental na construção de árvores de decisão é o de saber quando é que se deve parar a sua construção. Esta questão é colocada, pois alguns conjuntos de dados têm problemas de ruído, quer por erros de introdução dos dados ou por quaisquer interferências. Pode também acontecer que os melhores atributos para representar as classes não estejam disponíveis ou que o conjunto de dados contenha uma dimensão insuficiente para representar o problema.

Este tipo de problemas condiciona a performance dos algoritmos de aprendizagem e provoca o seu ajustamento perfeito ao conjunto de treino, originando o fenómeno designado por sobre-ajustamento (Breiman *et al.*, 1984). Quando detectado este problema pode ser um indicador do fraco poder de previsão da árvore de decisão como classificador. Para evitar este tipo de problemas existem dois tipos de estratégias que consistem em parar o crescimento da árvore mais cedo ou em podar a árvore após o seu crescimento.

A primeira estratégia consiste em tentar parar o crescimento da árvore antes que esta atinja o ponto perfeito de ajustamento. Isto pode ser conseguido pelo recurso à fixação de um limite, que em paralelo com a função de avaliação dos atributos pare o crescimento da árvore. Outra forma poderá ser recorrer a um teste de significância como critério de paragem.

O segundo tipo de estratégias consiste em deixar crescer a árvore de decisão sem impor limites ao seu crescimento e depois recorrer a técnicas de poda para substituir as sub-árvores geradas por folhas. A poda das árvores de decisão é efectuada no sentido das folhas para a raiz da árvore, calculando a estimativa da taxa de erro para cada sub--árvore. Se a substituição da árvore por uma folha resultar numa menor estimativa da

taxa de erro então a árvore é podada. Ao reduzirmos a estimativa da taxa de erro das várias sub-árvores estamos a reduzir também a estimativa da taxa de erro da árvore.

Sistemas de aprendizagem de árvores de decisão

O processo de aprendizagem de árvores de decisão é desenvolvido em duas etapas, na primeira é construída a árvore e na segunda é executada a sua poda. Na fase da construção, é gerada uma árvore de uma forma ambiciosa sem que haja a preocupação com o sobre-ajustamento dos dados. Este problema será resolvido na fase de poda. Para parar o crescimento da árvore são usados os seguintes critérios: *(i)* num determinado nó de decisão todos os exemplos do conjunto de treino pertencem a uma classe; *(ii)* menos que duas potenciais sub-árvores para cada teste possível, têm mais exemplos do que um número pré-definido à partida; *(iii)* não pode ser encontrado qualquer teste com uma função de avaliação.

CART

Breiman *et al.* (1984) propuseram o sistema CART que recorre à função de *Gini*, descrita pela equação 2.1, como medida para avaliar o nível de impureza dos dados e seleccionar os atributos.

$$Gini = \sum_j p(j \mid t)(1 - p(j \mid t)) = 1 - \sum_j p^2(j \mid t) \qquad (2.1)$$

Nesta função, a expressão $p(j \mid t)$ representa a frequência relativa da classe *j* no nó *t*. Num determinado nó devemos afectar todos os objectos da classe *j* com valor *1* e os restantes com o valor de *0*. A variância da amostra destes valores é dada por $p(j \mid t)(1 - p(j \mid t))$. O nível máximo de impureza dos dados é dado por $(1 - 1/n_c)$, em que n_c representa o número de valores possível da variável classe e é atingido quando os exemplos estão igualmente distribuídos pelas classes.

Este sistema adopta como estratégia de poda o *Cost Complexity Pruning* (custo da complexidade de poda). Esta medida de poda baseia-se em dois parâmetros, *R(T)* que

é erro por má classificação e $|T|$ que é o tamanho da árvore, medido pelo número de nós nela contidos. O custo da complexidade de poda é dado por, $R_\alpha(T) = R(T) + \alpha |T|$, em que α é o parâmetro que pondera a importância relativa do tamanho da árvore com a taxa de erro. Para cada valor de α, o objectivo é encontrar uma sub-árvore de tal forma que $T_\alpha \leq T$ e que $R_\alpha(T)$ seja minimizado. Se α for pequeno, o custo por existir um grande número de nós folha é pequeno e a árvore T_α será grande. Á medida que α aumenta, a árvore terá um menor número de nós folha. O processo de poda produz um conjunto finito de árvores (*T1*, *T2*, *T3*, …), sendo escolhida aquela que minimize $R_\alpha(T)$.

C4.5

O sistema C4.5 (Quinlan, 1993) em cada nó de decisão é escolhido o atributo que maximiza a função de avaliação, que pode ser o ganho de informação ou o rácio de informação[1]. A informação ganha por dividir o conjunto de treino *T* recorrendo ao teste $Teste_A$ é definida por:

$$Ganho(Teste_A) = inf(T) - inf_{Teste_A}(T) \tag{2.2}$$

$$Inf_{Teste_A}(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \inf(T_i) \tag{2.3}$$

$$Inf(T) = -\sum_{j=1}^{k} \frac{freq(C_j, T)}{|T|} \log_2 \left(\frac{freq(C_j, T)}{|T|}\right) bits \tag{2.4}$$

Na equação 2.2, a expressão *Inf(T)* é o valor médio de informação necessária para identificar a classe num exemplo *T* e a outra componente, $inf_{Teste_A}(T)$, é o valor médio da informação necessária para identificar a classe recorrendo ao atributo *A*. Nesta última, o conjunto de treino (*T*) é dividido em n subconjuntos (T_i) de acordo com os valores possíveis para o atributo. O valor de $freq(C_j,T)$ é o número de instâncias de *T* que pertencem à classe C_j.

[1] O sistema C4.5 usa por defeito o rácio de informação

O ganho de informação representa a redução média de informação esperada por recorrer a determinado atributo para separar os valores da classe. Apesar desta função proporcionar bons resultados, favorece os atributos com muitos valores possíveis. No limite tomará um valor máximo, se o atributo escolhido contiver um valor diferente para cada instância do conjunto de treino. Para colmatar este viés, foi implementado outro critério que normaliza os valores dos atributos testados e que é denominado de rácio de informação.

$$RacioGanho(Teste_A) = \frac{Ganho(Teste_A)}{SepInfo(Teste_A)} \tag{2.5}$$

$$\text{SepInfo}(\text{Teste}_\text{A}) = -\sum_{\text{i=1}}^{\text{n}} \frac{|\text{Ti}|}{|\text{T}|} \log_2 (\frac{|\text{Ti}|}{|\text{T}|}) \tag{2.6}$$

A expressão $SepInfo(Teste_A)$ é a informação potencial gerada pela divisão do conjunto de treino (*T*) em *n* subconjuntos.

Ainda na fase de crescimento da árvore de decisão, o C4.5 executa um procedimento de pré-poda. Depois de construir cada sub-árvore é verificado se esta tem uma taxa de erro de resubstituição maior do que a raiz da árvore de decisão se esta fosse eliminada. Neste caso, a sub-árvore é substituída por uma folha.

Para podar uma árvore de decisão, é necessário uma função para estimar a taxa de erro da árvore, sub-árvore ou folha. Uma hipótese será testar o erro num conjunto de dados independente, que proporciona estimativas de erro não enviesadas. As estratégias, *Cost Complexity Pruning* (Breiman *et al.*, 1984) e *Reduced Error Pruning* (Quinlan, 1987) recorrem a este método. Parte dos dados têm ser reservados para a poda da árvore de decisão e esta tem de ser construída com base num conjunto de dados menor, o que pode representar um problema.

Outra técnica consiste em avaliar uma árvore de decisão baseando-se apenas no conjunto de treino para calcular a taxa de erro. O erro de resubstituição (no conjunto de treino) não é uma função de avaliação apropriada, pois à medida que a poda avança, o erro de resubstituição também aumenta. Uma solução encontrada foi estimar o erro

pessimista que ajusta o erro de resubstituição aumentando o erro observado em cada folha em 0.5 (Quinlan, 1987).

A abordagem do C4.5 pertence ao conjunto de técnicas que recorrem ao erro pessimista. Nas versões mais recentes do C4.5, são adoptadas estratégias ainda mais pessimistas de erro. Supondo que uma folha cobre *N* instâncias com *E* erros e que o erro de resubstituição é *E/N*, então a estimativa da taxa de erro é dada pelo limite superior do intervalo de confiança, pressupondo que a taxa de erro segue uma distribuição binomial.

O processo da poda da árvore de decisão é realizado examinando cada nó decisão a partir das folhas até à raiz da árvore. Se a sub-árvore tiver uma taxa de erro estimada superior a da folha obtida pela sua eliminação, então essa árvore é substituída pela folha.

Os valores desconhecidos presentes no conjunto de dados são tratados de uma forma probabilística de acordo com a frequência relativa dos valores conhecidos. Um exemplo com um valor de teste desconhecido é dividido em fragmentos cujos pesos são proporcionais à frequência relativa de cada atributo, significando isto, que um único exemplo poderá seguir múltiplos caminhos na árvore.

A produção de regras de decisão também foi contemplada pelo sistema C4.5. O caminho da raiz até às folhas da árvore representa uma conjunção de condições que poderão ser representadas por uma regra. O lado esquerdo da regra conterá o conjunto de condições estabelecidas pelo caminho e o lado direito da regra especificará a classe na folha. O conjunto de regras obtido para cada classe constituirá um classificador de regras que terá o mesmo nível de exactidão das árvores de decisão e será mais facilmente interpretado.

Problemas associados às árvores de decisão

Os principais problemas associados às árvores de decisão são: a sua instabilidade, ou seja, uma pequena variação no conjunto de treino poderá provocar uma grande variação no modelo apreendido; e o da replicação de conceitos, dependendo do conjunto de dados, poderemos encontrar sub-árvores iguais dentro da mesma árvore. Para ilustrar

este último problema, apresentamos na Figura 2.2 uma árvore de decisão o conceito $(A \wedge B) \vee (C \wedge D)$, considerando que todos os atributos são binários.

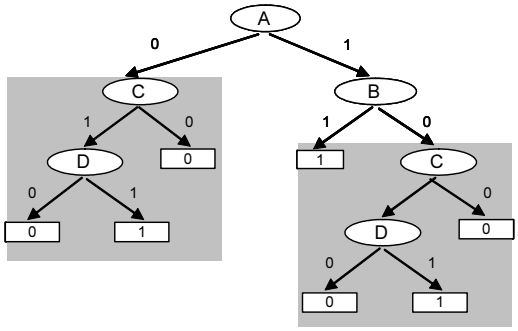


Figura 2.2: Árvore de decisão de $(A \wedge B) \vee (C \wedge D)$ com replicação de conceitos

O problema da replicação de conceitos nas árvores de decisão resulta da segmentação dos dados que ocorre após a realização de um teste. Na Figura 2.2, foi testado o atributo *A* na raiz da árvore, sendo dividido conjunto de dados em exemplos que contêm um valor para *A* igual a um, daqueles em que o valor de *A* assume zero. Nos nós de decisão seguintes, os subconjuntos de dados contêm todos os atributos à excepção do *A*, pelo que é possível que venham a ser seleccionados os mesmos atributos em dois ramos da árvore. Foi esta situação que aconteceu nesta árvore de decisão.

Árvores de decisão multivariadas

As árvores multivariadas ou oblíquas vieram colmatar uma limitação de representação das árvores univariadas, ao permitir testar em cada nó de decisão uma combinação de atributos.

A árvore de decisão representada na Figura 2.1 poderia ser simplificada se fosse adoptada uma árvore de decisão multivariada. Como podemos observar pela Figura 2.3 (b), o mesmo problema é agora representado por uma árvore de decisão que contém um único nó de decisão, cuja condição contém uma combinação linear de atributos. Com este teste é possível representar a fronteira do hiper-plano, separando as instâncias positivas das negativas.

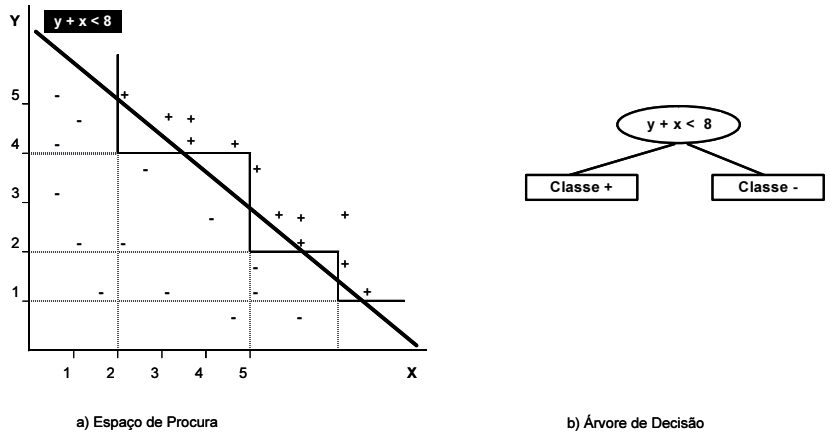


Figura 2.3: Árvore de decisão multivariada

Existe um paralelismo entre o método que propomos nesta tese e as árvores de decisão multivariadas. No método que propomos, são gerados novos atributos que resultam de conjunções de valores dos atributos originais. Pode ser ilustrado recorrendo à representação do conceito $(A \wedge B) \vee (C \wedge D)$ numa árvore de decisão. Se assumirmos a possibilidade de num nó de decisão testar conjunções de valores de atributos, então a árvore de decisão representada pela Figura 2.2 seria transformada numa árvore sem fragmentação de conceitos, descrita pela Figura 2.4 (b). O atributo *A* é substituído pela conjunção dos valores dos atributos *A* e *B*.

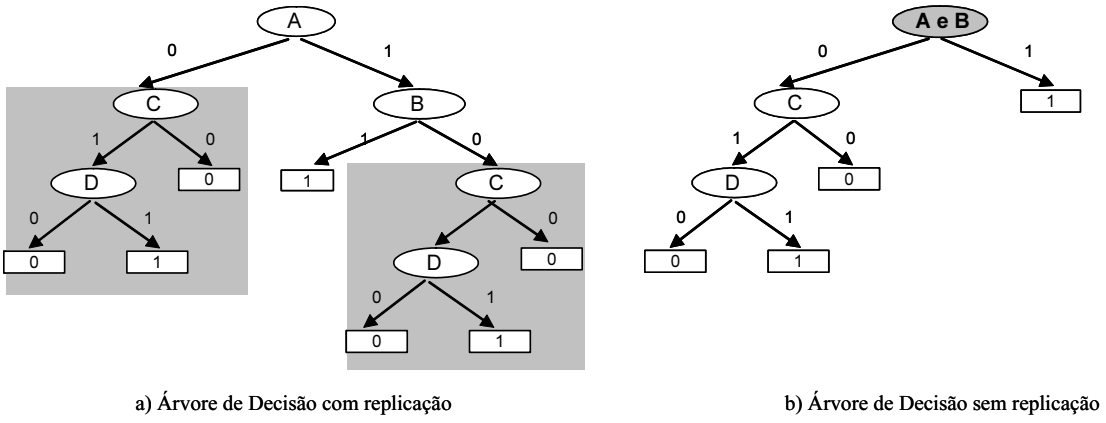


Figura 2.4: Árvore de decisão de $(A \wedge B) \vee (C \wedge D)$ sem replicação de conceitos

2.2 Aprendizagem de regras de associação

O contínuo armazenamento de dados gerou a necessidade de extracção de informação útil e motivou o aparecimento de técnicas que permitem a descoberta de padrões nos dados. A descoberta de regras de associação foi uma das técnicas que resultou desta linha de investigação.

Esta técnica foi inicialmente formulada por Agrawal *et al.* (1993) com vista à descoberta de padrões existentes em bases de dados transaccionais. Nestas bases de dados, cada linha de dados corresponde a uma transacção e cada uma destas é composta por um conjunto de artigos, a que denominam por cesto de compras. Cada artigo é designado por termo (item) e o conjunto de artigos comprados designa-se por conjunto de termos (itemset). Na Tabela 2.1 é apresentado um exemplo de uma base de dados transaccional.

Tabela 2.1: Base de dados transaccional

Nº Tran.	**Conjunto de termos (*itens*)**
1	bolachas(i1), café (i3), açúcar (i4)
2	café (i3), açúcar (i4)
3	bolachas (i1), amendoins (i2)
4	bolachas (i1), café (i3), açúcar (i4)
5	bolachas (i1), amendoins (i2), café (i3)

Um dos objectivos desta técnica é o de conhecer quais são os artigos comprados em conjunto. Algumas das conclusões que se podem retirar são do tipo: "20% das pessoas que compram café também compram bolachas". Esta informação, quando conhecida, poderá resultar num conjunto de acções que vão desde, a promoção conjunta de artigos ou a alterações da sua localização no supermercado.

No estudo das regras de associação estamos interessados em encontrar conjuntos de termos cujo suporte, calculado como o quociente entre o número de vezes que o conjunto de termos aparece pelo número total de linhas da base de dados, seja superior a um determinado limite mínimo, definido como suporte mínimo. Quando isto acontece, designamos o conjunto de termos por conjunto de termos frequente (*large ltemset*).

O suporte representa a significância estatística de um conjunto de termos e é definido pela equação 2.8. Supondo um conjunto de termos frequente $\mathrm{I} = \{\mathrm{i}_1, \mathrm{i}_2, i_3, i_4\}$ e um conjunto de dados com dimensão $|D|$, o suporte é dado pelo seu quociente.

$$Suporte = \frac{freq(\{i_1, i_2, , i_3, i_4\})}{|D|} \tag{2.8}$$

Para além de pretendermos saber, quais são os termos que são adquiridos conjuntamente, também é-nos útil saber quais são as combinações de termos que poderão ser descobertas e qual a seu nível de interesse. As combinações de termos são dadas pela construção de regras sob a forma $(I - a) \Rightarrow a$, em que $a \subseteq I$. O seu grau de interesse é representado pela confiança das regras, dada pela equação 2.9, e que consiste na probabilidade de ocorrer um conjunto de termos dado que ocorreu outro conjunto.

$$Conf[a \mid (I-a)] = \Pr ob[a \mid (I-a)] == \frac{\Pr ob(I)}{\Pr ob(I-a)} = \frac{\sup(I)}{\sup(I-a)}. \tag{2.9}$$

2.2.1 Algoritmo *Apriori*

O grande desafio associado à descoberta de regras de associação é o de encontrar o número desejável de regras no menor tempo de computação. Um algoritmo típico de descoberta de regras de associação é construído dividindo este problema em dois sub--problemas, o primeiro consiste em gerar todos os conjuntos de termos frequentes e o segundo, consiste em gerar para cada conjunto de termos frequente, todas as regras que contenham uma confiança superior ao valor mínimo.

O algoritmo *Apriori* (Agrawal e Strikant, 1994) é um dos algoritmos mais conhecidos e utilizados. Este algoritmo efectua múltiplas passagens pelo conjunto de dados. Gera os candidatos a ser contados numa passagem baseando-se apenas nos conjuntos de termos frequentes encontrados na passagem anterior, sem recorrer à base de dados. A intuição por detrás deste procedimento é a de que qualquer subconjunto de um conjunto de termos frequente também terá de ser frequente. Assim, os candidatos a

conjuntos frequentes com *k* termos, podem ser gerados combinando os conjuntos frequentes com *k-1* termos. Este procedimento resulta na geração de um número muito menor de conjuntos candidatos e permite que este algoritmo faça uma procura completa do espaço sem ser necessário testar todos os possíveis conjuntos de termos.

Considerando o conjunto *L= {1,2,3,4}* como um conjunto de termos, então o espaço de procura definido para este conjunto terá no máximo *16* conjuntos de termos frequentes como pode ser observado na Figura 2.5.

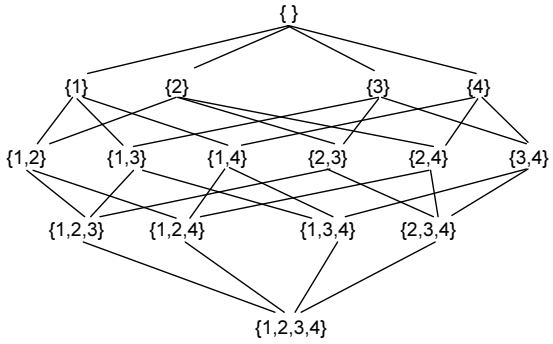


Figura 2.5: Complexidade do espaço de procura do algoritmo *Apriori*

O algoritmo de aprendizagem inicia a procura pelos termos individuais, aqueles que tiverem um nível de suporte superior ao mínimo serão escolhidos para o nível seguinte. O processo continua até que não se possam gerar mais conjuntos de termos frequentes de nível superior.

Ao crescimento linear do número de termos está associado um crescimento exponencial do número de conjuntos de termos frequentes gerados. No limite, poderemos ter 2^K combinações, em que *k* corresponde ao número de termos. Assim, um dos problemas associados a esta técnica, é o da complexidade do espaço de procura e a capacidade computacional necessária para processar um grande volume de dados.

Como podemos observar pela Figura 2.6, o primeiro passo do algoritmo, consiste em transformar o conjunto de dados num formato transaccional. Para isso é necessário construir uma tabela (b) que contenha em coluna todos os termos e que para cada linha sejam assinalados com o código *{1}* os termos presentes nessa transacção e com código *{0}* os restantes termos.

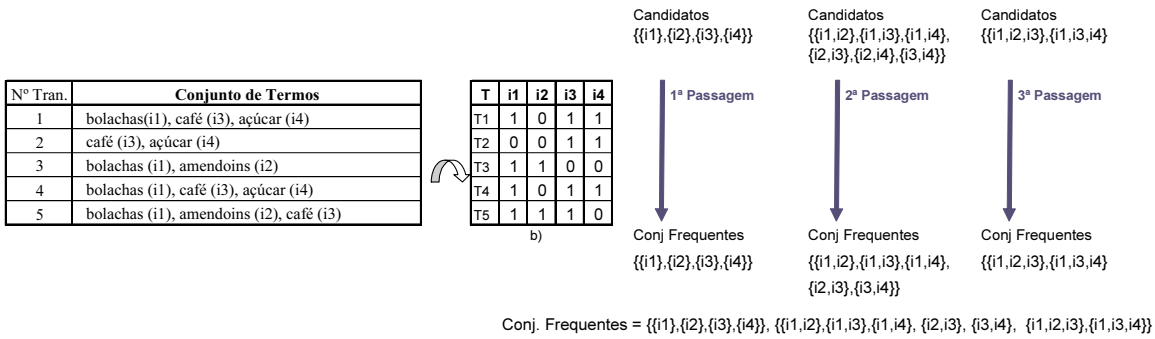


Figura 2.6: Primeira fase do algoritmo *Apriori*

A partir da tabela b) e da fixação de um valor para o suporte mínimo, que neste caso concreto foi de *20%*, iniciaremos as passagens sobre o conjunto de dados. Foram necessárias três passagens sobre o conjunto de dados para encontrar todos os conjuntos de termos frequentes. Na Figura 2.7 é descrito o algoritmo para gerar os conjuntos de termos frequentes.

Algoritmo APRIORI

```
1)  L1 = {conj freq tam.1}
2)  for ( K = 2 , Lk-1 <> 0; k ++)
3)        Ck = apriori-gen (Lk-1)
4)        For todas as transacções t em D
5)              Ct= sub-conjunto(Ck,t)
6)              para todos os candidatos  c em Ct
7)                    c.contagem++
8)        end
9)        Lk = {c  E Ck | c.contagem >= minsup}
10) end for
11) Resposta = Reunião de todos os Lk
```

Apriori-gen (L_{k-1})

```
    //Passo Join
1)  inserir para Ck
2)  select p.item1, p.item2, p.itemk-1,q.itemk-1
3)  From Lk-1 as p, Lk-1 as q
4)  where p.item1= q.item1 …p.itemk-2= q.itemk-2, p.itemk-1<q.itemk-1

    // Passo Prune
5)  for all itemsets c E Ck  do
6)        for all (k-1)-subsets s de c do
7)              if (s não pertence  Lk-1) then
8)                    delete c from Ck
9)        end
10) end
```

Figura 2.7: Algoritmo *Apriori* - geração dos conjuntos frequentes

A segunda fase do algoritmo *Apriori* consiste em gerar regras, a partir dos conjuntos de termos frequentes encontrados e que obedeçam ao critério de confiança mínima. Retomando o exemplo da Figura 2.6 e para o conjunto de termos frequente $\{i_1, i_3, i_4\}$ vamos extrair todas as regras, considerando o valor de *80%* para o parâmetro de confiança (Tabela 2.2).

Tabela 2.2: Segunda fase do algoritmo *Apriori*

Consequente com 1 termo		Consequente com 2 termos	
$\{i_1, i_3\} \Rightarrow \{i_4\}$	confiança = 0,4/0,6= 66%	$\{i_1\} \Rightarrow \{i_3, i_4\}$	confiança = 0,4/ 0,8 = 0,5
$\{i_1, i_4\} \Rightarrow \{i_3\}$	confiança = 0,4/ 0,4 = 100%	$\{i_3\} \Rightarrow \{i_1, i_4\}$	confiança = 0,4/ 0,8 = 0,5
$\{i_3, i_4\} \Rightarrow \{i_1\}$	confiança = 0,4/ 0,6 = 66%	$\{i_4\} \Rightarrow \{i_1, i_3\}$	confiança = 0,4/0,6= 0,66

O processo consiste em testar todas as combinações de termos que possam formar regras. Deverá existir, pelo menos, um termo como antecedente da regra e outro como consequente. Para o conjunto de termos em análise foram testadas seis possíveis regras mas apenas uma obedeceu ao critério de confiança mínima. A regra seleccionada foi $\{i_1, i_4\} \Rightarrow \{i_3\}$, que significa "*SE* comprar bolachas *ENTÃO* também comprará açúcar e café". Esta regra aplica-se em *40%* dos casos com uma confiança de *100%*. A construção de regras de associação é implementada de acordo com o algoritmo descrito na Figura 2.8.

```
1) for all k item-sets L_k, K>= 2 do begin
2)       H_1={conseq das regras derivados de l_k com 1 item no conseq}
3)       call ap-genrules(l_k, H_1)
4) end

   Procedimento ap-gen-rules (l_k: con freq tam. k, H_m: conj de itens de tam m}
         if (k>m+1) then begin
               H_m+1 = apriori-gen(H_m)
               for all h_m+1 E H_m+1 do begin
                     conf = support (l_k)/suporte(l_k-h_m+1)
                     if (conf >= mincon then
                           output  the rule (l_k-h_m+1) => h_m+1 com confiança = conf e suporte = support (l_k)
                     else
                           delete h_m+1 from H_m+1
               end
               call ap-gen-rules (l_k,H_m+1)
         end
```

Figura 2.8: Algoritmo *Apriori* - construção de regras de associação

Um dos problemas associados ao algoritmo *Apriori* é o da complexidade computacional exigida para o processamento dos dados. Com vista a colmatar este problema foram desenvolvidos outros algoritmos que minimizam as operações de pesquisa no conjunto de dados.

O algoritmo *Apriori-Tid*, definido por Agrawal e Strikant (1994) é um desses algoritmos, incidindo em particular, no problema do acesso constante ao conjunto de dados. Como o acesso (operação *I/O*) ao conjunto de dados é mais lento do que o acesso à memória central, este algoritmo acede uma vez ao conjunto de dados para guardar em memória os termos frequentes de tamanho um. A partir destes termos frequentes são gerados candidatos de tamanho superior e contados a partir dos dados existentes em memória.

Comparando o *Apriori* e o *Apriori-Tid*, verificou-se que o segundo tem desempenho superior quando o conjunto de candidatos cabe em memória e inferior caso contrário. Neste contexto, Agrawal e Strikant, propuseram uma solução algorítmica híbrida, *Apriori Hybrid* (1994). Neste caso recorre-se ao *Apriori* quando os candidatos não cabem em memória, passando então para o algoritmo *Apriori Hybrid*. Com base nestes algoritmos, *Apriori* e *Apriori-Tid*, vários investigadores propuseram implementações alternativas para a descoberta de regras de associação.

Savasese *et al.* (1995), propõem um algoritmo paralelizável que consiste em dividir o conjunto de dados em várias partições que cabem em memória e identificar os conjuntos candidatos em cada partição. Sabe-se que um conjunto de termos frequente aparece pelo menos numa partição. No final da primeira passagem existem um super conjunto de candidatos e na segunda passagem, estes conjuntos de candidatos são contados em toda a base de dados. Esta proposta revela-se muito eficiente.

Toivowen (1996) propôs um método que consiste em obter o conjunto de candidatos a partir de uma amostra da base de dados

Uma outra alternativa é o método *DIC*, *Dynamic Itemset Counting,* sugerido por Brin *et al.* (1997) e cujo objectivo fundamental foi a generalização do algoritmo *Apriori*. Pretendeu-se reduzir o número de passagens pela base de dados, melhorando-se assim, o desempenho da descoberta das associações.

2.2.2 Aprendizagem de regras de classificação

A descoberta de regras de associação não é orientada para um determinado conceito, o que se pretende é descobrir padrões nos dados, satisfazendo os critérios de suporte e confiança definidos à partida. As técnicas de classificação, pelo contrário, têm como objectivo encontrar um conjunto restrito de regras que consigam descrever um conceito objectivo.

Nesta secção vamos abordar os métodos associados ao desenvolvimento de classificadores recorrendo à descoberta de regras de associação. O método *CARS* (*Classification Association rules*) desenvolvido por Liu *et al.* (1998) e o *Apriori-C* (*Apriori for Classification*) criado por Jovanoski e Lavrac (2001).

O algoritmo *CARS* integra as duas técnicas de *data mining*, a descoberta de regras de associação e a construção de um modelo para problemas de classificação. A integração destas técnicas é realizada a partir da pré-selecção de um conjunto de regras de associação cujo consequente contenha um dos valores possíveis para a variável classe. Na primeira fase, o sistema gera um conjunto completo de regras que satisfazem as condições de suporte e confiança mínimos. Na segunda fase, é construído um classificador a partir das regras encontradas. A classificação de novos exemplos será feita pela procura da primeira regra que satisfizer o exemplo. Caso não exista uma regra que cubra o exemplo então este será classificado com a classe maioritária.

O método *Apriori-C* recorre à mesma lógica de construção do método *CARS*. No entanto, as contribuições deste algoritmo estão na redução substancial do tempo de execução do algoritmo, na redução da complexidade do espaço de procura através da selecção de atributos e na melhor interpretabilidade dos resultados através do pós-processamento de regras.

2.3 Heurísticas para a selecção de regras

As regras de associação representam padrões dos dados e permitem-nos perceber as relações existentes entre os atributos de um conjunto de dados. A natureza intrínseca

de um algoritmo de descoberta de regras de associação conduz a que o número de regras gerado seja muito grande, problema de quantidade, e que nem todos as regras encontradas sejam interessantes, problema de qualidade (Tan *et al.*, 2000).

O problema da quantidade de regras geradas já foi abordado por vários autores. Agrawal *et al*., na formulação do algoritmo *Apriori*, criaram um mecanismo que permite limitar o número de regras geradas, tanto por via suporte como por via da fixação de um nível mínimo de confiança. Liu *et al.* recorreram ao teste de χ^2 (qui-quadrado) para podar as regras não interessantes. A qualidade das regras definidas pelo seu nível de interesse, pode ser aferida através de métricas que permitam ordená-las segundo determinado critério.

Nas próximas secções, vamos abordar um procedimento estatístico para resumir os dados relevantes entre dois atributos e referenciar algumas das heurísticas existentes para avaliar o interesse de um determinado padrão de dados. Esta abordagem é feita agrupando as medidas de interesse por área de origem, isto é, medidas aplicadas na área de *data mining*; da estatística e da teoria da informação.

2.3.1 Tabelas de contingência

A descoberta de dependências entre variáveis é uma área da estatística que se encontra bem estudada. Os atributos ou variáveis considerados nos conjuntos de dados em estudo são nominais, contendo várias categorias que são exaustivas e mutuamente exclusivas.

Quando pretendemos obter uma classificação cruzada entre duas variáveis, recorremos a uma tabela que em linha contém as categorias de uma variável e em coluna as categorias da outra. Cada célula desta tabela é preenchida pela contagem do número de vezes que as duas categorias (uma de cada variável) aparecem conjuntamente. Estas tabelas são denominadas de Tabelas de Contingência. Na Tabela 2.3 é apresentado um exemplo de uma destas tabelas entre duas variáveis, a variável *A* com *r* categorias e a variável *B* com *s* categorias. A frequência observada de exemplos, da categoria *r* da variável *1* e da categoria *s* da variável *2*, é definida por $\boldsymbol{n}_{rs}$. O número

total de observações de uma categoria da variável *A* é definida por $n_{r.}$, enquanto que para a variável *B* é $n._{..s}$. O número total de observações é dado pelo somatório dos totais, em linha ou em coluna, e é definido por *N*.

Tabela 2.3: Tabela de contingência

Variável A \ Variável B	1	2	…	s	Total
1	n_{11}	n_{12}	…	n_{1s}	$n_{1.} = \sum_{j=1}^{s} n_{1j}$
2	n_{21}	n_{22}	…	n_{2s}	$n_{2.} = \sum_{j=1}^{s} n_{2j}$
…	…	…	…	…	…
r	n_{r1}	n_{r2}	…	n_{rs}	$n_{r.}$
Total	$n_{.1} = \sum_{i=1}^{r} n_{i1}$	$n_{2.} = \sum_{j=1}^{s} n_{2j}$	…	$n_{.s}$	$N = \sum_{i=1}^{r} n_{i.} = \sum_{j=1}^{s} n_{.j}$

Para avaliarmos o grau de interesse de cada conjunto de termos frequente teremos de recorrer a uma tabela contingência para resumir a informação entre, a projecção dos conjuntos de termos frequentes sobre o conjunto de treino e o atributo classe.

2.3.2 Heurísticas aplicadas na área de *data mining*

Nos últimos anos, têm sido propostas várias métricas para extrair padrões interessantes a partir de grandes bases de dados. Agora, vamos apresentar as características gerais de uma medida de interesse e definir algumas medidas concretas para extracção de padrões interessantes.

Uma medida objectiva deverá ser baseada nos dados e ser independente do domínio de dados considerado. Foram definidos por Piatetsky-Shapiro (1991) três princípios a que qualquer medida (*M*) deve obedecer:

1. Se *A* e *B* são duas variáveis estatisticamente independentes, *P(A,B)=P(A).P(B)* e *M=0*. Isto significa que quando duas variáveis são estatisticamente independentes, a medida de interesse tomará o de valor zero;

2. A medida *M* cresce, se *P(A,B)* crescer e todos os restantes parâmetros permanecerem constantes;
3. A medida *M* decresce, se *P(A)* ou *P(B)* decrescerem, mantendo-se constantes todos os restantes parâmetros.

O primeiro princípio estabelece que os padrões que ocorrerem aleatoriamente não têm interesse. O segundo princípio estabelece que o interesse de uma medida deverá ser, tanto maior quanto maior for a sua significância estatística. E por ultimo, o terceiro, é usado para comparar os valores de interesse de padrões com mesma significância estatística.

Fronteira de suporte e confiança

Foi demonstrado por Bayardo e Agrawal (1999) através do estudo da monotonia das funções que avaliam as regras de associação que o conjunto de regras óptimas se encontra na fronteira de suporte e confiança. O conceito de fronteira de suporte--confiança implica a ordenação parcial das regras em termos do seu suporte e confiança. Neste conceito, os autores exploram as propriedades de monotonia das funções. Uma função diz-se monótona em *x*, se $x_1 < x_2$ implica que $f(x_1) < f(x_2)$. Concluíram que o suporte goza desta propriedade e que a confiança é anti-monótona.

Considerando uma ordem parcial das regras como $\leq_{SC}$, ordem parcial de suporte e confiança, então para as regras R_1 e R_2 estabelecem que a relação $R_1 <_{SC} R_2$ ocorre, se e só se:

- $\sup orte(R_1) < \sup orte(R_2) \wedge confiança(R_1) < confiança(R_2)$;
- $\sup orte(R_1) < \sup orte(R_2) \wedge confiança(R_1) \leq confiança(R_2)$.

Todas as regras que respeitam as anteriores condições, formam a fronteira superior de suporte e confiança. Adicionalmente, $R_1 =_{SC} R_2$, se e só se, $\sup(R_1) = \sup(R_2)$ e $conf(R_1) = conf(R_2)$.

Considerando outra ordem parcial similar, $\leq_{s\neg c}$, tal que $R_1 <_{s\neg c} R_2$ se e só se:

- $\sup orte(R_1) < \sup orte(R_2) \wedge confiança(R_1) > confiança(R_2)$
- $\sup orte(R_1) < \sup orte(R_2) \wedge confiança(R_1) \geq confiança(R_2)$.

Estas condições são equivalentes às anteriores e o conjunto de regras que obedecem as estas condições formam a fronteira inferior de suporte e confiança.

Segundo os autores, o conjunto de regras óptimas é formado pelas regras que se encontram na fronteira superior ($\leq_{SC}$) e inferior ($\leq_{s\sim c}$). Esta fronteira é denominada de fronteira de suporte- confiança (SC) e está representada na Figura 2.9.

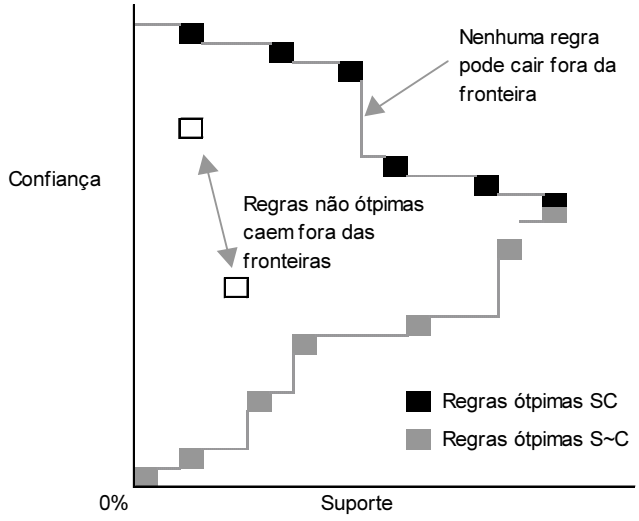


Figura 2.9: Fronteira suporte-confiança (SC)

Ficou também demonstrado que as várias medidas de interesse, como por exemplo, coeficiente de *Laplace*, o *ganho de informação*, o *coeficiente de interesse*, caem na fronteira *SC*.

Coeficiente de interesse

O coeficiente de interesse reflecte a noção estatística de independência entre duas variáveis aleatórias. Foi abordado por muitos autores como uma medida para avaliar os

níveis de associação. Esta métrica é definida pelo quociente entre a probabilidade conjunta de duas variáveis relativamente à sua probabilidade pressupondo a hipótese de independência. O valor *1* corresponde à independência estatística entre as variáveis *A* e *B*.

$$I(A,B) = \frac{P(A,B)}{P(A).P(B)} \tag{2.10}$$

Coeficiente de *Laplace*

Por último, o coeficiente de *Laplace*, apresentado como uma heurística de avaliação no sistema CN2 desenvolvido por Clark e Bowell (1991), por substituição da heurística baseada na entropia. Os autores concluíram que este coeficiente tende a privilegiar as regras mais gerais e com maior poder de previsão. Para a regra $A \Rightarrow C$ e para os *k* valores possíveis da variável classe, a função será dada por

$$Laplace\,(A => C) = \frac{\sup(A => C) + 1}{\sup(A) + k} = \frac{n_c + 1}{n_{tot} + k}. \tag{2.11}$$

O suporte da regra *A=>C* é dado por *sup(A=>C)* que corresponde à frequência relativa do conjunto de termos frequente *{A,C}*, o suporte do termo *A* é dado por *sup(A)*, *k* é o número de classes do domínio de dados, n_i é o número de exemplos da classe *c* cobertos pela regra e n_{tot} é o número total de exemplos cobertos pela regra.

2.3.3 Heurísticas da área da estatística

Brin *et al.* (1997) propuseram medir a significância das regras de associação recorrendo ao teste de χ^2. O objectivo consiste em encontrar uma fronteira entre os conjuntos de termos frequentes correlacionados e não correlacionados. Um conjunto de termos frequentes é correlacionado, se os termos que o compõem também o forem entre si. Os autores provaram que se um conjunto de termos frequentes for correlacionado, então todos os conjuntos de termos frequentes de tamanho superior também o serão. Concluíram que o problema da descoberta de regras associação fica menos complexo,

quando se recorre ao suporte e ao teste de χ^2 como estratégias de poda. Os conjuntos de termos frequentes interessantes estão contidos numa fronteira formada pelo suporte mínimo e pelo nível crítico da estatística de χ^2.

Outros autores, Liu *et al.* (1999) recorrem ao teste de χ^2 para podar as regras não significantes e seleccionam um subconjunto de regras que formam um resumo das associações encontradas nos dados. Este subconjunto de regras designa-se de regras direccionais, por definirem um resumo das associações descoberta nos dados. O processo de selecção do conjunto de regras direccionais consiste em determinar o tipo de correlação existente em cada regra, sabendo que apenas aquelas regras que contêm uma correlação positiva é que são consideradas. Após esta fase, cada regra é analisada para decidir se a regra segue uma direcção já estabelecida por outras regras ou se estabelece uma nova direcção.

Teste de χ^2

A independência de duas variáveis pode ser avaliada pelo teste χ^2**.** Sendo a probabilidade de uma observação pertencer à categoria *i* da variável *A* e à categoria *j* da variável *B* dada por p_{ij}, em que p_{ij} poderá ser calculada dividindo a frequência da categoria pelo número total de observações. A hipótese das duas variáveis serem independentes é dada pela hipótese nula (H_0), contra uma hipótese alternativa (H_1) que sustenta que as variáveis não são independentes. A função conjunta de duas variáveis independentes é igual ao produto das respectivas funções marginais de probabilidade. As estimativas das probabilidades são calculadas a partir dos valores observados e resumidos numa tabela de contingência. As estimativas das probabilidades marginais de p_{ij} são respectivamente, $\hat{p}_{i.} = n_{i.} / N$ e $\hat{p}_{.j} = n_{.j.} / N$. Se admitirmos que a hipótese nula é verdadeira, então as frequências esperadas, e_{ij}, podem ser calculadas recorrendo à relação $e_{ij} = N.\hat{p}_{i.}.\hat{p}_{.j}$.

A estatística do teste de independência de χ^2 é dado pela equação 2.12. São comparados os desvios entre as frequências observadas e as esperadas.

$$ET = \sum_{i=1}^{r} \sum_{j=1}^{s} \frac{(n_{ij} - e_{ij})^2}{e_{ij}} \tag{2.12}$$

No caso da hipótese nula ser verdadeira, a estatística segue aproximadamente uma distribuição de χ^2 com *(r-1).(s-1)* graus de liberdade. Os valores baixos da estatística de teste suportam a hipótese nula, pelo que os valores críticos do teste devem ser fixados na cauda direita da distribuição $\chi^2_{(r-1)(s-1)}$.

Coeficiente de Correlação

Tan e Kumar (2000) investigaram a proximidade existente entre várias medidas interesse e a definição estatística de correlação, tendo concluído que muitas das medidas de interesse (*Interesse*, *Laplace*, *Entropia, Ganho de informação*) são muito similares ao conceito estatístico de correlação em regiões de baixos suportes.

O coeficiente de correlação (ρ_{AB}) entre duas variáveis, *A* e *B*, mede o grau de linearidade entre duas variáveis aleatórias e teoricamente, é definido pelo quociente entre a covariância (*cov(A, B)*) e o desvio-padrão das duas variáveis (σ_A e σ_B).

$$\rho_{AB} = \frac{\text{cov}(A,B)}{\sigma_A \sigma_B} \tag{2.13}$$

O cálculo do desvio-padrão para a variável A assumindo que esta é uma variável dicotómica, é dado por $\sigma_A = \sqrt{p(1-p)}$ em que $p \equiv P(A) = f_{1.} / N$. Com o auxílio de uma tabela de contingência é possível definir o coeficiente correlação entre as variáveis *A* e *B* pela equação 2.14.

$$\rho_{AB} = \frac{f_{11} f_{00} - f_{10} f_{01}}{\sqrt{f_{1.} f_{0.} f_{.1} f_{.0}}} \tag{2.14}$$

Nesta equação, f_{ij} representa os valores das frequências observadas de uma tabela de contingência com duas dimensões, por exemplo, o valor de f_{11} descreve o valor das

frequências quando as duas variáveis tomam o valor *1* e f_{10} representa o valor da frequência quando a variável A toma o valor *1* e a variável *B* toma o valor *0*.

Índice de associação de *Kruskal-Goodman*

Goodman e Kruskal, num artigo publicado em 1954 propuseram uma medida que designaram de índice de associação. Basearam-se na observação de que, se duas variáveis forem muito dependentes, então o erro de previsão de uma das variáveis poderá ser menor, se a outra variável já for conhecida. O índice de associação para a variável *A* é definido pela equação 2.15.

$$\lambda_A = \frac{P(E_A) - P(E_A \mid B)}{P(E_A)} \tag{2.15}$$

O valor de λ_A representa a redução relativa do erro em prever *A*, E_A, dado o conhecimento da outra variável (*B*). Se não for disponibilizada mais informação, então a melhor estimativa do valor de *A* será aquele que tiver maior probabilidade, isto é, $\hat{A} = arg(max_k\, P(A_k))$. O erro por se recorrer a esta estimativa é dado por $P(\mathrm{E}_A) = 1 - P(\hat{A}) = 1 - max_k(P(A_k))$. Supondo que $B = B_1$, então a melhor estimativa para A será dada pelo valor que maximiza a probabilidade condicional $\hat{A} = arg(max_k\, P(A_k \mid B_1))$ e o erro associado a este estimador é dado por $P(\mathrm{E}_A \mid B_1) = 1 - max_k(P(A_k \mid B_1))$. A média do erro de previsão de *A* dado *B*, é dado pela ponderação de todos os valores de *B*.

$$P(\mathrm{E}_A \mid B) = P(\mathrm{E}_A \mid B)P(B_1) + \ldots + P(\mathrm{E}_A \mid B_m)P(B_m) = \ldots = 1 - \sum_j \max_k P(A_k, B_j).$$

Em que $\sum_j P(B_j) = 1$ e $P(A_k \mid B_j)P(B_j) = P(A_k, B_j)$. O cálculo de λ_A pode ser reescrito da seguinte forma:

$$\lambda_A = \frac{\sum_j \max_k P(A_k, B_j) - \max_k P(A_k)}{1 - \max_k P(A_k)} =$$

$$\lambda_A = \frac{\sum_j \max_k f_{ik} - \max_k f_{.k}}{N - \max_k f_{.k}}. \qquad (2.16)$$

2.3.4 Heurísticas da área da teoria da informação

A teoria da informação foi desenvolvida por *Shannon* na década de 40 com o objectivo de determinar o montante de informação que pode ser transmitido através de um canal de comunicação imperfeito. Este investigador recorreu ao conceito de entropia, que foi herdado da termodinâmica, para provar o teorema de codificação sem ruído (*noiseless source coding theorem)*. A entropia mede o montante de informação contido numa variável aleatória discreta e representa o comprimento médio de uma mensagem, necessário para transmitir o seu resultado. Para além do conceito de entropia, foram também desenvolvidos os conceitos de informação condicional e de informação mútua.

Os conceitos relacionados com entropia foram depois recuperados e usados em vários domínios como medidas de incerteza de uma variável aleatória. Nos trabalhos desenvolvidos por Quinlan, ID3 (1987) e C4.5, estes conceitos são usados como critérios de avaliação de atributos.

Sendo X uma variável aleatória, $H(X)$ dá-nos o valor da entropia para essa variável.

$$H(X) = E\left[\log \frac{1}{P(X)}\right] = -\sum_{i=1}^{n} P(X_i) \log P(X_i) \qquad (2.17)$$

A entropia conjunta de duas variáveis aleatórias, X e Y, é o montante médio de informação necessário em média para especificar ambos os valores.

$$H(X,Y) = E\left[\log \frac{1}{P(X,Y)}\right] = -\sum_{i,j=1}^{n,m} P(X_i, Y_j) \log P(X_i, Y_j) \qquad (2.18)$$

A entropia condicional de uma variável aleatória *Y* dado o conhecimento de *X*, expressa o montante de informação adicional que é em média necessário fornecer para comunicar *Y*, dado que a outra parte já conhece *X*.

$$\begin{aligned} H(X \mid Y) &= E_{XY}\left[\log \frac{1}{P(X \mid Y)}\right] = \\ &= \sum_{i,j=1}^{n,m} (X_i, Y_j) \log P(X_i \mid Y_j) = \\ &= -\sum_{j=1}^{m} P(Y_j) \sum_{i=1}^{n} P(X_i \mid Y_j) \log P(X_i \mid Y_j) = \\ &= \sum_{j=1}^{m} P(Y_j) H(X \mid Y_i) \end{aligned} \tag{2.19}$$

A informação mútua entre as variáveis, *X* e *Y,* é dada por *MI(X,Y)*. Corresponde à redução da incerteza de uma variável aleatória devido ao conhecimento existente sobre a outra variável, ou dito de outra forma, representa o montante de informação que uma variável aleatória contém da outra variável. O valor do coeficiente *MI(X,Y)* é igual a zero quando, *X* e *Y,* são independentes, isto é, quando *H(X| Y)=H(X)*. Este conceito representa o mesmo que o ganho de informação definido por Quinlan.

$$MI(X;Y) = H(X) - H(X \mid Y) = \sum_{i,j=1}^{n,m} p(X_i, Y_j) \log \frac{P(X_i, Y_j)}{P(X_i) P(Y_j)} \tag{2.20}$$

2.4 Modelos múltiplos

Os modelos múltiplos surgiram da necessidade de superar as restrições dos modelos simples. Estes métodos recorrem à indução construtiva para incorporar o conhecimento obtido nas diferentes etapas da sua construção.

2.4.1 Indução construtiva

Os algoritmos de aprendizagem baseados numa inferência indutiva convencional representam teorias (hipóteses) recorrendo aos atributos fornecidos no conjunto de treino. O sistema C4.5 divide o espaço de instâncias em regiões com o mesmo valor de

classe através de cortes ortogonais face aos eixos. Cada corte é baseado apenas no valor de um único atributo.

Em determinados domínios e dado um problema concreto, os peritos humanos apenas conseguem fornecer atributos que não descrevem com clareza o conceito objectivo, contêm a informação, mas não existe uma forma simples e directa de a representar. Se os atributos disponibilizados no conjunto de dados forem apropriados para representar as teorias objectivo dadas as linguagens de representação, então os algoritmos de aprendizagem deverão ter uma boa performance, em termos de exactidão das previsões e de complexidade das teorias geradas. Se os atributos disponíveis não forem apropriados para descrever as teorias objectivo, então deparamo-nos com problemas para os quais não existem soluções no domínio dos algoritmos de aprendizagem convencionais.

Um método para ultrapassar a limitação dos atributos originais é o da indução construtiva (Michalski, 1978). Os algoritmos baseados nesta técnica constroem novos atributos a partir dos atributos originais, devendo estes últimos serem mais apropriados para apreender o conceito objectivo. As diferenças entre a variedade de sistemas de construção indutiva existentes, residem na forma como constroem os novos atributos. Podem diferir nos operadores construtivos usados, que podem ser conjuntivos ou disjuntivos. Também podem diferir nos métodos de procura de operandos.

Com base na estratégia de construção usada ou na forma como os novos atributos são construídos, ou seja, se usaram teorias já apreendidas ou descrições contidas nos dados, os sistemas de indução foram divididos em quatro categorias: *(i)* os baseados em hipóteses; *(ii)* nos dados, *(iii)* no conhecimento; e *(iv)* em estratégias múltiplas.

Neste estudo, propomos um método para construir novos atributos binários para a aprendizagem de árvores de decisão, usando para isso outro sistema de aprendizagem.

2.4.2 Tipos de modelos múltiplos

O recurso a modelos múltiplos de aprendizagem resulta de nenhum modelo, por si só, conseguir cobrir todos os espaços de hipóteses possíveis e também, por não existirem modelos que obtenham os melhores resultados em todos os problemas.

O objectivo geral da utilização de modelos múltiplos de aprendizagem é o de podermos obter estimadores que consigam melhorar a capacidade de previsão e a estabilidade dos modelos simples (Domingos, 1998). Outro aspecto muito importante, é o da simplicidade e da transparência dos modelos criados, pois não basta ter uma boa capacidade de previsão, é também necessário que os resultados possam ser interpretados de uma forma intuitiva.

Na literatura existem diferentes tipos de modelos múltiplos, uns que utilizam apenas um algoritmo de aprendizagem e manipulam o conjunto de dados e outros que combinam vários algoritmos de aprendizagem.

No grupo dos modelos que combinam as previsões provenientes de vários classificadores, temos os sistemas de voto, que agregam as previsões dos diferentes classificadores e fornecem a classificação final. Destacam-se dois tipos de sistemas de voto: os uniformes, em que as previsões dos diferentes classificadores contribuem da mesma forma para a decisão final, que é dada pela classe mais votada; e os sistemas de votação ponderada, para os quais, cada algoritmo tem associado uma ponderação que reflecte a sua importância nas tarefas de previsão.

Outro grupo de modelos, que podemos classificar de homogéneos, são aqueles que recorrem apenas a um algoritmo de aprendizagem e que manipulam o conjunto de dados. No sistema de *Bagging* (Breiman, 1996), a aprendizagem do modelo consiste em obter n réplicas, com reposição, do conjunto de treino. As amostras têm o mesmo número de exemplos do conjunto de treino (normalmente 25 amostras) e para cada amostra é gerado um classificador. Para cada exemplo de teste é determinada a classe prevista por cada classificador e depois as previsões são agregadas por votação uniforme.

Ainda no grupo dos modelos homogéneos, temos o sistema de *Boosting* (Schapire, 1994) em a aprendizagem do modelo é realizada através de um algoritmo iterativo. A cada exemplo é associado um determinado peso, que inicialmente será igual para todos os exemplos. Iterativamente o algoritmo gera um classificador conforme a distribuição actual dos exemplos. Os exemplos incorrectamente classificados verão o seu peso incrementado para o passo seguinte. Por fim, os classificadores serão agregados por votação ponderada.

Podemos ainda encontrar um grupo de modelos que combinam vários algoritmos de aprendizagem. Um dos métodos que pertencem a este grupo é o Método de Generalização em Pilha (*Stacking Generalization*) que foi criado por Wolpert (1992) e é baseado numa arquitectura distribuída por camadas. Outro método que pertence à família dos métodos de generalização em pilha é o método de generalização em cascata (*Cascade Generalization*) desenvolvido por Gama (1999).

2.4.3 Generalização em cascata

A ideia subjacente ao método de Generalização em Cascata consiste em utilizar algoritmos de aprendizagem em sequência. Em cada iteração ocorre um processo com dois passos: no primeiro passo, recorre-se a um classificador para construir um modelo; e no segundo passo, o espaço definido pelos atributos originais é estendido pela inserção de novos atributos usando o modelo criado. A Figura 2.10 descreve esta sequência.

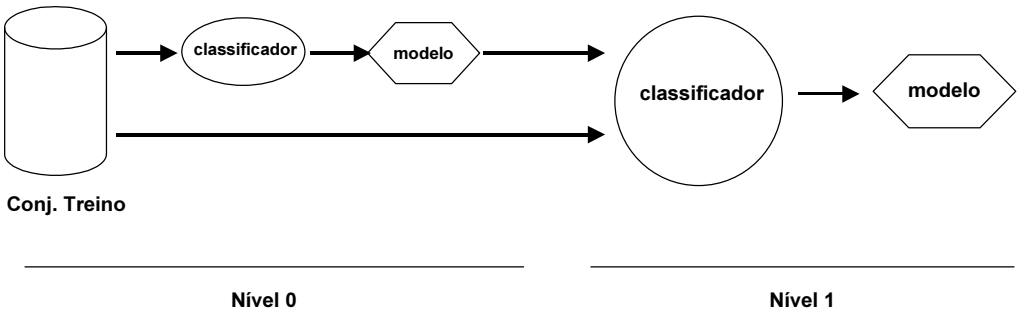


Figura 2.10: Arquitectura do modelo de generalização em cascata

Se na sequência, um classificador recorrer aos novos atributos para construir o seu modelo, então a sua linguagem de representação foi estendida. As restrições existentes na linguagem de representação dos classificadores de alto nível são menores devido à inclusão de termos das linguagens de representação dos classificadores de base.

O método que propomos nesta tese baseia-se numa arquitectura de generalização em cascata. Numa primeira camada recorremos à aprendizagem de regras de associação para construir novos atributos que serão inseridos no conjunto de dados e que numa segunda camada estes novos atributos serão usados para construir um modelo baseado em árvores decisão.

2.4.4 Métodos alternativos para construção de árvores de decisão

Terabe *et al.* (2002) publicaram um artigo no qual propõem um método que consiste em alargar o conjunto de dados através da construção de novos atributos. Pretendem melhorar a performance das árvores de decisão fornecendo para classificação fornecendo um conjunto de dados com mais atributos do que os originais. Os novos atributos são construídos directamente a partir da informação extraída das regras de associação.

Neste método os autores seleccionam as regras de associação que contêm como consequente um dos valores possíveis para a variável classe. A partir destas regras são construídos os atributos candidatos. São seleccionados os atributos que contenham um valor de ganho de informação superior a zero. Após esta fase, os novos atributos são inseridos no conjunto de dados sob a forma de atributos binários e o conjunto de dados alargado é classificado por uma árvore de decisão.

Os resultados experimentais deste método usando o *C4.5* como algoritmo de aprendizagem de árvores de decisão, foram melhores do que os resultados obtidos no conjunto de dados original.

Berzal (2002) na sua tese de doutoramento apresentou outro método, o ART (*Association Rules Tree*) para construção de árvores de decisão. Este método não

recorre a uma medida de impureza como a entropia, mas sim, à informação contidas nas regras de associação para decidir como ramificar a árvore de decisão.

O método *ART* constrói um modelo de classificação com base nas regras de associação e utiliza o subconjunto do conjunto de regras obtido para ramificar a árvore de decisão. As regras de associação são extraídas tentando encontrar as mais simples. Assim sendo, este algoritmo começa por tentar seleccionar regras que apenas contenham um antecedente e se não existir nenhuma que satisfaça os parâmetros definidos, então são procuradas as regras com dois antecedentes. A partir das regras extraídas, são seleccionadas aquelas com maior valor de confiança. Com esta informação é já possível construir o primeiro nível da árvore. Os exemplos do conjunto de treino classificados no primeiro nível da árvore são agora excluídos deste conjunto. Após esta fase, o *ART* procura as regras mais simples que permitam classificar os exemplos actuais. As regras são seleccionadas e é construído um segundo nível da árvore. O processo só termina quando não existirem mais exemplos para classificar.

Este método de construção de árvores de decisão é caracterizado por usar simultaneamente vários atributos para ramificar a árvore de decisão, o que permite construir modelos mais compactos.

2.5 Avaliação de modelos

Um dos problemas com que normalmente nos deparamos é o de escolher dentro de um conjunto alargado de algoritmos disponíveis, aquele que fornecerá a melhor solução para nosso problema. Os algoritmos de aprendizagem podem ser avaliados segundo várias dimensões, nomeadamente, a exactidão das suas estimativas, o tempo de aprendizagem necessário para gerar os modelos, a complexidade dos modelos, a sua simplicidade e a transparência dos resultados.

2.5.1 Estimativa da taxa de erro

A avaliação da exactidão das previsões dos modelos assenta na estimação da sua taxa de erro. Qualquer estratégia para estimar a taxa de erro deve ter em conta as causas da sua variabilidade, que poderão estar ligadas à aleatoriedade interna do algoritmo, à

variabilidade do conjunto de teste, provocada pela existência ou não de informação suficiente para calcular a estimativa da taxa de erro, e à variabilidade do conjunto de treino, nomeadamente porque os modelos produzidos pelos algoritmos de aprendizagem poderão ser instáveis a pequenas variações do conjunto de treino. A própria manipulação de alguns parâmetros dos algoritmos de aprendizagem poderá conduzir a diferentes modelos.

A taxa de erro dos algoritmos de aprendizagem é calculada pelo quociente entre o número de exemplos mal classificados e o número total de exemplos. Podemos distinguir dois tipos de erro. O primeiro, o erro de amostragem, que resulta da estimação da taxa de erro a partir do conjunto de treino, e o segundo, o erro de generalização, que é a estimativa da taxa de erro obtida a partir de um conjunto independente de dados.

Existem várias técnicas para obter a estimativa da taxa de erro: *(i)* o *boostrap*, definido como um processo de amostragem com reposição; *(ii)* o *hold-out* em que se recorre a um conjunto independente de dados que não é utilizado para a construção do modelo; *(iii)* a validação cruzada que é um processo de amostragem com reposição em que se divide o conjunto de dados em *n* partições e iterativamente é gerado um modelo com *n-1* partições dos dados, sendo este testado na partição que não foi utilizada na construção do modelo; *(iv)* e no processo de validação cruzada estratificada cujas partições mantêm a proporção das classes observadas no conjunto de treino.

2.5.2 Outras dimensões de avaliação

A complexidade dos modelos criados é outra dimensão com interesse, pois um dos objectivos na construção de modelos é torná-los o mais simples possível. As métricas para medir a complexidade dos algoritmos variam, pois cada algoritmo representa o conhecimento de uma forma diferente. Por exemplo, a complexidade dos modelos criados pelos algoritmos de aprendizagem de árvores de decisão é medida pelo número de nós, de decisão e terminais, contidos no modelo.

Outra dimensão relevante é o tempo de aprendizagem necessário para o algoritmo construir o modelo. A aplicação dos algoritmos a problemas reais depende da sua rapidez na construção de modelos.

2.5.3 Testes de significância

Para comparar as performances entre dois modelos e perceber se as estimativas das taxas de erro são diferentes, devemos recorrer a testes estatísticos de significância. No modelo que propomos neste estudo, temos de perceber se as estimativas das taxas de erro produzidas pelo modelo, são ou não, estatisticamente diferentes, das obtidas pelo modelo obtido no conjunto de dados original.

O teste *t*, de significância, averigua as hipóteses de as médias das diferenças (μ_Δ) dos erros obtidos pelos dois tipos de modelos, são estatisticamente diferentes de zero. Considerando duas amostras, *A* e *B*, com dimensão de *N* elementos, são calculadas as diferenças entre os elementos das duas amostras ($\Delta = x_A - x_B$), a média ($\overline{\Delta}$) e o desvio-padrão das diferenças (S_Δ). A hipótese nula (H_0) subjacente a este teste assume que a média das diferenças é igual a zero, contra uma hipótese alternativa (H_1) em que esta é diferente de zero.

$$H_0 : \mu_\Delta = 0$$
$$H_1 : \mu_\Delta > 0$$

Atendendo a que as diferenças seguem uma distribuição normal e que a amostra constituída por tais diferenças é pequena, então a estatística do teste pode ser definida por

$$ET = \frac{\overline{\Delta}}{S_\Delta / \sqrt{N}}. \qquad (2.21)$$

Quando H_0 é verdadeira, esta estatística segue uma distribuição *t-student* com *n-1* graus de liberdade, t_{n-1}. Para concluir pela rejeição ou não da hipótese H_0, é necessário definir um nível de significância crítico, normalmente é fixado um nível de *5%*.

3 Metodologia

O presente estudo tem como objectivo principal melhorar a exactidão das previsões geradas por modelos treinados por algoritmos de aprendizagem de árvores de decisão. Neste capítulo descrevemos a metodologia que foi seguida para avaliar a possibilidade de obter ganhos de performance pela combinação de duas técnicas de aprendizagem: árvores de decisão e regras de associação. Após a descrição da motivação são descritas, a arquitectura do modelo, o ambiente de modelação das experiências e a avaliação dos modelos. Finalmente, será apresentado um exemplo ilustrativo da aplicação deste método.

3.1 Motivação

Propomos uma metodologia que combine duas técnicas de aprendizagem, uma descritiva, a descoberta de regras de associação e outra de previsão, a aprendizagem árvores de decisão.

Por um lado, a descoberta de regras de associação não são uma técnica que possa ser directamente aplicada a todo o tipo de problemas, é mais apropriada para análise e descrição de estruturas nos dados. Os algoritmos são compostos por duas fases, a extracção de conjuntos de termos frequentes e geração de regras de associação. Na primeira fase é executada uma procura completa do espaço e são encontrados os conjuntos de termos frequentes. Estes conjuntos tomam a forma de conjunções $atr_i = valor$ e representam a co-ocorrência simultânea de conjunções de valores de diferentes atributos. Na segunda fase, são geradas as regras de associação.

Por outro lado, os algoritmos de aprendizagem de árvores de decisão fazem uma procura heurística do espaço de hipóteses, em cada nó de decisão é seleccionado um teste no valor de um atributo assumindo que todos os atributos são independentes. Uma vez instalado um teste, os exemplos de treino são divididos e o processo é repetido para cada subgrupo de exemplos. Se o teste realizado no nó de decisão não reflectir a real

distribuição dos registos pode originar um problema de replicação de conceitos. Cada percurso desde a raiz da árvore até uma das folhas define uma regra de decisão. As regras de decisão tomam a forma, $se\ atr_i = a \wedge atr_j = b\ então\ classe = c$.

Cada uma destas técnicas de aprendizagem possui determinadas características que as tornam mais adequadas para determinadas tarefas em detrimento de outras. Segundo Schaffer (1994), não existe um único algoritmo igualmente bom em todos os problemas, isto é, todos os algoritmos podem ser particularmente bons num tipo de problema, mas ineficazes numa outra classe de problemas. Esta observação inviabiliza a possibilidade de se modificar um algoritmo de forma a torná-lo rigorosamente melhor em todos os possíveis problemas: o desempenho de generalização é transferido de uma situação de aprendizagem para outra.

A par com as limitações associadas às técnicas de aprendizagem, os atributos fornecidos nos conjuntos de dados também podem ser limitadores da aprendizagem das teorias. Se aqueles não forem os mais apropriados para representar as teorias objectivo, dadas as linguagens de representação, então os modelos construídos serão pouco precisos em termos de previsão. É também comum neste contexto a existência de valores desconhecidos ou omissos, que perturbam ainda mais a aprendizagem.

Com vista a colmatar as limitações na aprendizagem, quer devido às técnicas de aprendizagem ou pelos conjuntos de dados usados, é vantajoso recorrer à indução construtiva (Michalski, 1978). Com base nos atributos originais são construídos novos atributos, devendo estes ser mais apropriados para captar o conceito objectivo dos problemas.

O nosso objectivo final será obter modelos que melhorem a performance dos algoritmos de aprendizagem que geram árvores de decisão, na exactidão das previsões efectuadas e na redução da complexidade das teorias geradas. Recorremos ao método de generalização em cascata, no qual, dois ou mais algoritmos são usados em sequência para gerar um modelo. Nesta tese usamos os algoritmos de descoberta de regras de associação e de aprendizagem de árvores de decisão. O algoritmo de descoberta de regras de associação vai extrair os conjuntos de termos frequentes que serão a base para a construção dos novos atributos. Os novos atributos se forem seleccionados para a

construção da árvore de decisão, então a linguagem de representação das árvores de decisão é estendida pela linguagem de representação das regras de associação. Depois do conjunto de dados ser estendido pelos novos atributos é construído um modelo com base no algoritmo de aprendizagem de árvores de decisão.

Terabe *et al.* (2002) e Berzal (2002) propuseram métodos que recorrem à aprendizagem de regras de associação como suporte à construção de árvores de decisão.

3.2 Arquitectura do modelo

O método que propomos está representado pela arquitectura do modelo descrita na Figura 3.1. Numa primeira fase, são extraídos os conjuntos de termos frequentes através do recurso ao algoritmo *Apriori*, de aprendizagem de regras de associação. Os conjuntos de termos frequentes são filtrados para eliminar aqueles não interessantes para análise e depois são seleccionados os mais relevantes. O conjunto de dados é estendido pelos novos atributos gerados a partir dos conjuntos de termos frequentes seleccionados. De seguida, é gerada uma árvore de decisão usando todos os atributos disponíveis, os originais e os construídos na fase anterior.

A implementação do método é executada obedecendo aos seguintes passos:

(i) Pré-processamento dos dados;
(ii) Extracção dos conjuntos de termos frequentes;
(iii) Filtro de poda dos conjuntos sem interesse para análise;
(iv) Heurística de ordenação dos conjuntos;
(v) Filtro de selecção dos conjuntos;
(vi) Inserção de novos atributos no conjunto de dados;
(vii) Classificação do conjunto de dados recorrendo a uma árvore de decisão.

Passaremos de seguida à sua descrição mais detalhada.

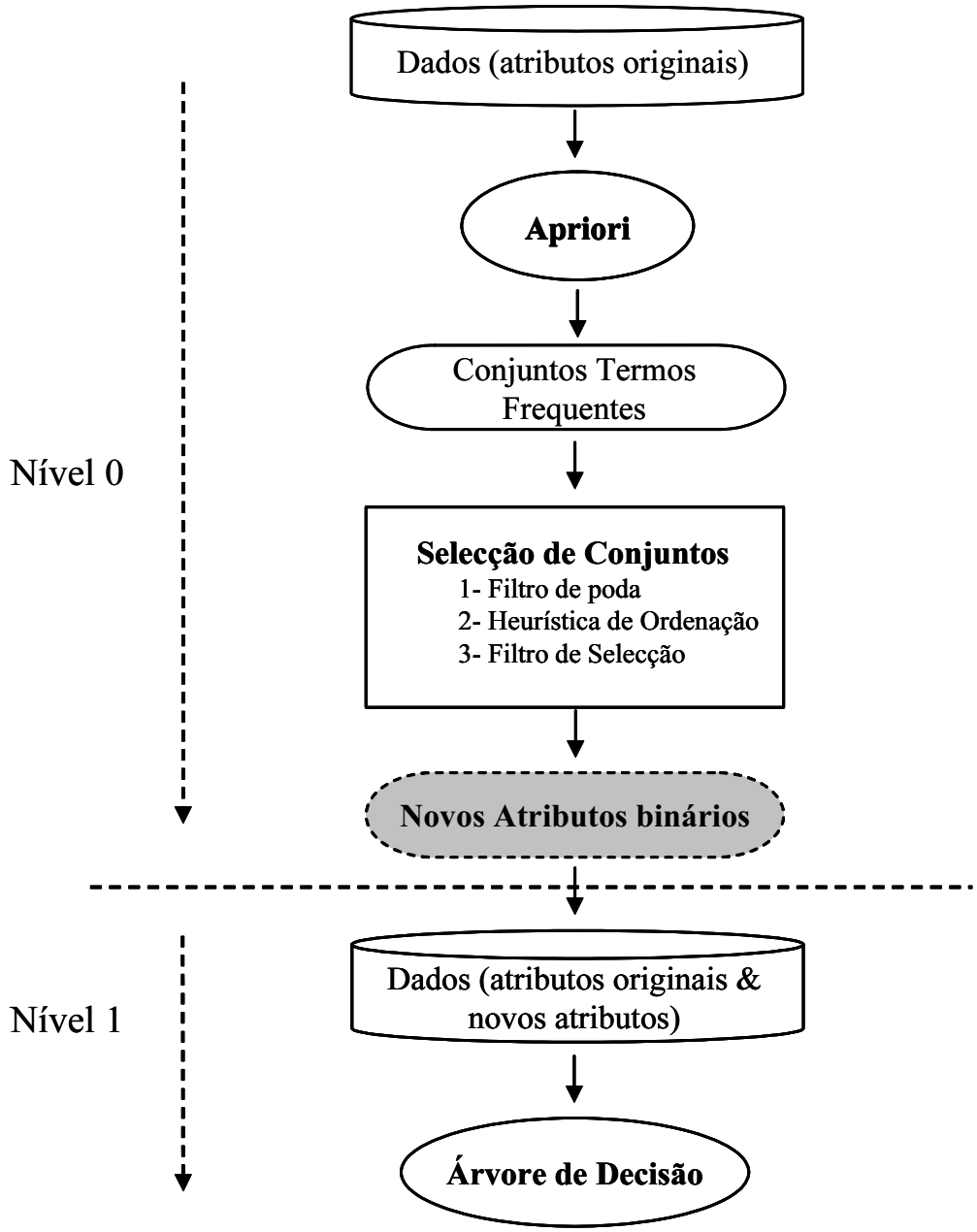


Figura 3.1: Arquitectura do método proposto

Pré-processamento dos dados

Esta fase compreende um conjunto de procedimentos de verificação dos conjuntos de dados. É necessário verificar o tipo de atributos, pois se estes forem contínuos terão de ser discretizados. Outro procedimento consiste em averiguar se existem valores omissos ou desconhecidos, e nestes casos, procedemos à eliminação física das instâncias com esses valores. Nesta tese para simplificar a análise recorremos a conjuntos de dados com atributos nominais e sem valores omissos ou desconhecidos.

Extracção dos conjuntos de termos frequentes

O algoritmo de descoberta de regras de associação é composto por duas fases distintas: na primeira fase são extraídos, os conjuntos de termos frequentes que contenham um suporte superior a um mínimo pré-fixado; e na segunda fase são construídas regras que contenham o parâmetro de confiança superior a um valor mínimo pré-definido. Para este estudo em concreto, é usado o *Apriori*, como algoritmo de aprendizagem, sendo apenas necessário proceder à extracção dos conjuntos de termos frequentes.

Um conjunto de treino é descrito como um conjunto de instâncias, que podem ser caracterizadas por um conjunto de atributos. Uma instância *i* pode ser caracterizada por $\{(x_{i,j}, y_z) | \forall i \in K, j \in L_i, z \in L_c\}$ em que x_{ij} é o valor do atributo i para o valor j que é um dos valores possíveis do atributo, L_i é o número dos valores possíveis do atributo *i* e *K* é o número de atributos excluindo o atributo classe. A classe é representada por y_z e L_c é o número dos seus valores possíveis.

Cada par atributo (atr_i) igual a valor ($x_{i,j}$), $atr_i = x_{ij}$, é transformado num termo na forma $termo_i = (atr_i, x_{i,j})$ ou $termo_c = (classe, y_z)$. Um conjunto de termos frequentes corresponde a uma conjunção de pares, atributo igual a valor, formalizado por,

$$\{(< atr_{1j}, x_{i,j} >, < atr_2, x_{2,j} >, ..., < atr_k x_{k,j} >, < classe, y_z >) : j \in L_i \wedge z \in L_c\} \qquad (3.1)$$

O nosso interesse consiste em extrair todos os conjuntos de termos frequentes que tenham uma frequência mínima superior a um dado suporte mínimo (*s%*).

Filtros de poda dos conjuntos de termos frequentes

O algoritmo *Apriori* utilizado na aprendizagem de regras de associação, extrai todas as combinações de termos que contenham suporte superior a um determinado valor mínimo pré-fixado. A procura exaustiva de conjuntos de termos frequentes em regiões de suportes baixos conduz frequentemente a um problema de excesso de conjuntos de termos frequentes. O número excessivo de conjuntos de termos frequentes encontrados pode ser reduzido pelo recurso a estratégias de filtro que eliminem parte dos conjuntos. As estratégias de eliminação são as seguintes:

(i) A primeira estratégia consiste em excluir os conjuntos de termos frequentes que não contenham um dos valores possíveis do atributo classe. Como pretendemos saber como é que os restantes atributos estão relacionados com um determinado atributo objectivo, então um atributo que esteja fortemente associado a um valor da classe deverá ser útil para prever o seu valor;

(ii) A segunda estratégia consiste numa combinação da primeira estratégia com outros processos de filtro. Após termos aplicado a primeira estratégia de filtro calculamos o teste de χ^2 para cada conjunto de termos frequentes. Este teste é realizado com base numa tabela de contingência entre a projecção de cada conjunto de termos frequentes no conjunto de treino e o atributo classe. O objectivo é excluir os conjuntos de termos em que não seja possível rejeitar a hipótese de independência para com o atributo classe, para um nível de confiança de *95%*. Outros autores, tais como Liu *et al.* (1999), exploraram esta técnica para eliminar os conjuntos de termos frequentes não significantes, definindo uma fronteira entre os conjuntos dependentes e independentes face ao atributo objectivo.

Como elemento de controlo a este processo mantivemos, paralelamente, todos os conjuntos de termos frequentes gerados, sem considerar qualquer estratégia de filtro.

Heurísticas de ordenação de conjuntos

A qualidade dos conjuntos de termos frequentes foi avaliada pelo recurso a um conjunto de heurísticas que permitem medir a dependência entre os atributos ou a forma como estes estão associados. Como heurísticas que medem o grau de dependência entre os atributos seleccionamos o coeficiente de interesse e o coeficiente de correlação. Relativamente aquelas que medem o grau de associação entre os atributos, temos um conjunto mais alargado que integra o coeficiente de entropia, ganho de informação, rácio de informação, coeficiente de *Gini*, índice de associação de *Kruskal-Goodman* e o coeficiente de *Laplace*.

O primeiro conjunto de heurísticas que nos propusemos recorrer foram aquelas usadas nos algoritmos de aprendizagem de árvores de decisão como medidas de avaliação de atributos: consideramos as medidas baseadas na entropia e o coeficiente de *Gini*. Também recorremos ao coeficiente de *Laplace*, por este já ter sido usado com sucesso no sistema *CN2*. Por fim, incluímos o índice de associação de *Kruskal--Goodman*, pois este mede a redução relativa do erro na previsão de um atributo, por se conhecer outro atributo e permite-nos aferir da importância de cada um dos novos atributos para as tarefas de classificação. O nosso objectivo na selecção de um conjunto alargado de heurísticas foi o de poder seleccionar aquelas que melhor se adequam a este tipo de avaliação.

Todas as medidas são calculadas com base numa tabela de contingência construída a partir da projecção do conjunto de termos frequentes sobre o conjunto de treino. Nesta projecção foi ignorado o valor do atributo classe para que os resultados não fossem viciados.

Após serem calculadas as heurísticas para cada conjunto de termos frequentes, estes são ordenados por ordem decrescente de interesse e são seleccionados o número de conjuntos a integrar no modelo.

Filtros de selecção de conjuntos de termos frequentes

Após a ordenação dos conjuntos de termos frequentes é necessário definir filtros para seleccionar o número de conjuntos. Optamos por três estratégias de filtro:

(i) Seleccionar os conjuntos de termos frequentes cujo valor da heurística seja superior ao valor máximo desta nos atributos originais. Consideramos quanto maior for o seu valor, mais interessantes serão os atributos. A nossa motivação foi seleccionar atributos mais interessantes que os originais;

(ii) A segunda estratégia consiste em seleccionar um número de conjuntos de termos frequentes igual ao número de atributos originais. Consideramos que no máximo serão necessários, tantos novos atributos quantos os atributos originais necessários para representar o espaço de procura onde se encontra o conceito objectivo. Como cada novo atributo resulta de uma combinação de valores dos atributos originais e considerando que esses atributos originais serão aqueles que originalmente permitiam definir o conceito objectivo, então admitimos numa atitude pessimista, que no limite serão necessários o mesmo número de atributos que os originais;

(iii) A última estratégia resulta da combinação das estratégias anteriores. Quando numa experiência não são seleccionados conjuntos de termos frequentes recorrendo à primeira estratégia, então, mesmo correndo o risco de estarmos a introduzir atributos não significantes, optamos por inserir tantos novos atributos quantos os atributos originais existentes (aplicar a segunda estratégia definida).

Construção de novos atributos

Depois de seleccionarmos os conjuntos de termos frequentes, iniciaremos o processo de construção de novos atributos. Cada conjunto de termos frequente dará origem a um novo atributo. Para cada instância do conjunto de treino e do conjunto de

teste, o novo atributo tomará o valor 1 se e só se, a conjunção de pares *atributo = valor*, se verificar para a instância em concreto. Este processo é executado, omitindo do conjunto de dados e do conjunto de termos frequente, a coluna referente ao valor da classe.

Classificação

O novo conjunto de dados, estendido pelos novos atributos, será classificado por um algoritmo de aprendizagem de árvores de decisão e os modelos gerados são avaliados em cada experiência realizada. Uma experiência resulta da aplicação a um conjunto de dados, de um algoritmo de aprendizagem e de uma heurística de ordenação dos conjuntos de termos frequentes. Para cada uma destas experiências, serão geradas duas árvores de decisão, uma para o conjunto de dados original e a outra para o conjunto de dados estendido pelos novos atributos. A avaliação dos modelos é feita pelos níveis de erro obtidos e pela complexidade dos modelos gerados.

3.3 Avaliação dos modelos gerados

Os modelos gerados através dos processos descritos na secção anterior podem ser avaliados segundo várias dimensões. O mais usual é avaliar a capacidade de previsão de exemplos não observados, a complexidade das teorias e o tempo de aprendizagem dos modelos. O nosso objectivo é avaliar os modelos segundo as duas primeiras dimensões.

As árvores de decisão são geradas a partir de um conjunto de treino que contém cerca de *67%* das observações e testadas noutro conjunto com as restantes *33%* das observações. Recorremos a uma técnica de validação cruzada com *10* iterações e ao processo de amostragem com reposição. Para cada iteração, são calculadas as taxas de erro (ERR^0) do conjunto original de dados e as taxas de erro (ERR^1) no conjunto de dados estendido pelos novos atributos. Após o processamento das dez iterações, são calculadas as taxas médias de erro para cada conjunto de dados e o ganho médio (*GA*) em pontos percentuais. O nível de exactidão de um modelo (*ACC*) foi usado em

algumas situações como complemento à informação fornecida pelas taxas de erro. As equações seguintes descrevem as formulações utilizadas.

$$ERR^p = \frac{\sum_{i=1}^{10} ERR_i^p}{10} : p = \{0,1\} \quad (3.2)$$

$$GA = \frac{\sum_{j}^{10} (ERR_j^0 - ERR_j^1)}{10} \quad (3.3)$$

$$ACC = \frac{\sum_{j}^{10} (1 - ERR_j)}{10} \quad (3.4)$$

De seguida recorremos a testes de significância para comparar as performances dos dois modelos e aferir se as estimativas das taxas de erro são estatisticamente diferentes.

A complexidade das teorias é aferida pelo tamanho da descrição da teoria apreendida, e no caso das árvores de decisão é definida pelo número total de nós, de decisão e folhas, gerados na construção da árvore ($\sum_i nós_i$).

3.4 Exemplo ilustrativo

Como experiência exemplificativa deste método, recorremos ao conjunto de dados *TicTacToe* (Blake *et al*., 1999) que contém *9* atributos nominais, *2* classes e *958* exemplos. Foi usada a técnica de validação cruzada, com *10* iterações, e amostragem não estratificada. Os critérios de avaliação usados foram, a taxa de erro de generalização e a complexidade das árvores geradas.

Algumas das instâncias do conjunto de dados que recorremos para este exemplo estão representadas na Tabela 3.1. Cada coluna do conjunto de dados representa um

atributo e na primeira linha estão descritos os nomes dos atributos. Estão representadas quatro instâncias do conjunto de dados, duas positivas e duas negativas.

Tabela 3.1: Exemplo do conjunto de dados *TicTacToe* de nível *0*

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
b	o	b	b	o	x	x	o	x	negative.
o	x	o	b	o	x	x	x	o	negative.
x	b	b	x	o	o	x	o	x	positive.
x	x	x	b	b	o	o	x	o	positive.

O modelo composto, isto é, árvore de decisão após Regras de Associação (*Rpart*∇*Apriori* **),** funciona em sequência, de acordo com a arquitectura apresentada na Figura 3.1. No nível *0*, são descobertos os conjuntos frequentes mais interessantes e no nível 1, recorremos ao algoritmo de aprendizagem de árvores de decisão (*rpart)* para classificar o conjunto de dados estendido pelos novos atributos.

O algoritmo de aprendizagem *Apriori* foi executado assumindo um suporte mínimo de *5%*. Foram gerados em média *1412* conjuntos de termos frequentes.

Neste exemplo, consideramos os níveis de filtro de poda de conjuntos de termos frequentes definidos na secção anterior. A aplicação da primeira estratégia que consistiu em eliminar todos os conjuntos de termos frequentes que não contivessem qualquer termo igual a um dos valores possíveis do atributo classe, resultou numa redução em 66% dos conjuntos de termos. O recurso à segunda estratégia de filtro, que consistia em calcular o teste de χ^2 para os conjuntos de termos, conduziu a uma redução de *83%* dos conjuntos gerados, passando de um universo de *1412* conjuntos para um total de *245* de termos frequentes. O resumo dos resultados destas estratégias está descrito na tabela 3.2.

Tabela 3.2: Resultados do filtro dos conjuntos frequentes

Filtro de conjuntos de termos frequentes	**Nº conjuntos**	**% redução**
Conjuntos de termos frequentes **totais**	1412	0
Conjuntos frequentes com um **termo = valor da classe**	479	66
Conjuntos frequentes com um **termo = valor da classe e p-value< 5%**	245	83

O interesse dos conjuntos de termos frequentes foi avaliado recorrendo ao coeficiente de entropia. A partir da projecção do conjunto de termos frequentes sobre o conjunto de treino e do atributo classe, foi construída uma tabela de contingência. A Tabela 3.3 resume a distribuição das observações do conjunto de treino, tendo em conta estes dois atributos.

Tabela 3.3: Exemplo de uma tabela de contingência

		Classe		
		negative	positive	$f_{i.}$
Projecção do Conjunto	0	254	442	**696**
	1	045	121	**166**
	$f_{.j}$	**299**	**563**	**862**

A entropia representa o valor médio de informação necessário para identificar a classe com base na projecção do conjunto frequente. O seu valor para os atributos contidos na Tabela 3.3 é de *0.927* bits. É calculado da seguinte forma:

$$Entropia_{projecção}(classe) = (696/862) * (-(254/696) * log_2(254/696) - \\ -(442/696) * log_2(442/696)) + (166/862) * (-(45/166) * log_2(45/166) - \\ -(121/166) * log_2(121/166)) = 0.927 bits$$

Depois de calcular o valor do coeficiente de entropia associado a cada conjunto de termos frequentes, procedemos à selecção dos conjuntos de termos frequentes a reter na análise. Para isso, recorremos ao primeiro filtro de selecção e vamos seleccionar todos os conjuntos de termos frequentes que contenham um valor de entropia inferior ao valor mínimo obtido nos atributos originais. Seguindo esta estratégia, na iteração 9 foi seleccionado o seguinte conjunto de termos frequentes,

$$X -> y = \{(<V1,"o">,<V5,"o">,<V9,"o">,<V10,"negative">)\}.$$

O passo seguinte é projectar este conjunto de termos frequentes sobre os conjuntos de treino e de teste. No nosso exemplo, o conjunto de dados, tomou a forma descrita na Tabela 3.5.

Tabela 3.4: Conjunto de dados *TicTacToe* de nível 1

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	NovoAtrib
b	o	b	b	o	x	x	o	x	negative.	0
o	x	o	b	o	x	x	x	o	negative.	1
x	b	b	x	o	o	x	o	x	positive.	0
x	x	x	b	b	o	o	x	o	positive.	0

O resumo desta experiência é apresentado na Tabela 3.5. A coluna 1 desta tabela fornece-nos a informação da iteração corrente e a coluna 2 indica-nos o suporte médio dos conjuntos frequentes gerados. As colunas 3 e 4 indicam-nos, respectivamente, o número total de conjuntos gerados e o seu número após aplicação do filtro de poda. A coluna 5 descreve o número de novos atributos seleccionados. Para avaliação da exactidão das previsões do modelo, temos informação sobre o erro obtido no conjunto de dados original (coluna 6), no conjunto de dados estendido (coluna 7) e sobre o ganho em pontos percentuais (coluna 8). No que respeita à complexidade dos modelos gerados, é descrita a informação sobre a complexidade das árvores de decisão geradas, no conjunto de dados original (coluna 9) e no conjunto de dados estendido (coluna 10).

Tabela 3.5: Resultados do exemplo do modelo em cascata:

		Conjuntos	Conjuntos	Numero	Exactidão das Previsões			Complexidade da Teoria	
Iteração	Suporte	Gerados	Finais	atributos	Erro.orig	Erro.est	ganho	Árv. Incial	Árv. Final
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
1		1415		0	0,083	0,083	0,000	39,0	39,0
2	0,056	1403	252	1	0,052	0,042	**0,010**	41,0	37,0
3		1436		0	0,094	0,094	0,000	45,0	45,0
4	0,055	1411	245	1	0,094	0,052	**0,042**	43,0	37,0
5		1409		0	0,073	0,073	0,000	45,0	45,0
6		1394		0	0,104	0,104	0,000	41,0	41,0
7		1399		0	0,063	0,063	0,000	45,0	45,0
8	0,051	1403	236	1	0,125	0,021	**0,104**	45,0	37,0
9	0,052	1427	239	1	0,084	0,042	**0,042**	45,0	37,0
10	0,056	1421	254	1	0,116	0,042	**0,074**	45,0	37,0
Média	**0,054**	**1412**	**245**	**1**	**0,089**	**0,062**	**0,027**	**43,4**	**40,0**
D. Padrão					**0,023**	**0,027**	**0,037**		

O erro médio do modelo composto foi de *6.2%* contra um erro médio no conjunto de dados de nível 0 de *8.9%*. Assim, observou-se um ganho de 2.7 pontos percentuais, o que representa uma redução de *30%* no erro cometido pelo modelo.

Na iteração 9 foi seleccionado um conjunto frequente para integrar os dados de nível 1. O erro produzido pelo modelo foi de *4.2%* face a um erro no conjunto de dados

de nível *0* de *8.4%*, o que corresponde um ganho de *4.2* pontos percentuais a uma redução de *50%* no erro produzido pela árvore de decisão.

A árvore de decisão gerada para o conjunto de dados de nível *0* na iteração *9* é apresentada na Figura 3.3. Podemos também observar algumas sub-árvores iguais, assinaladas por rectângulos a tracejado, que representam fragmentações de conceitos.

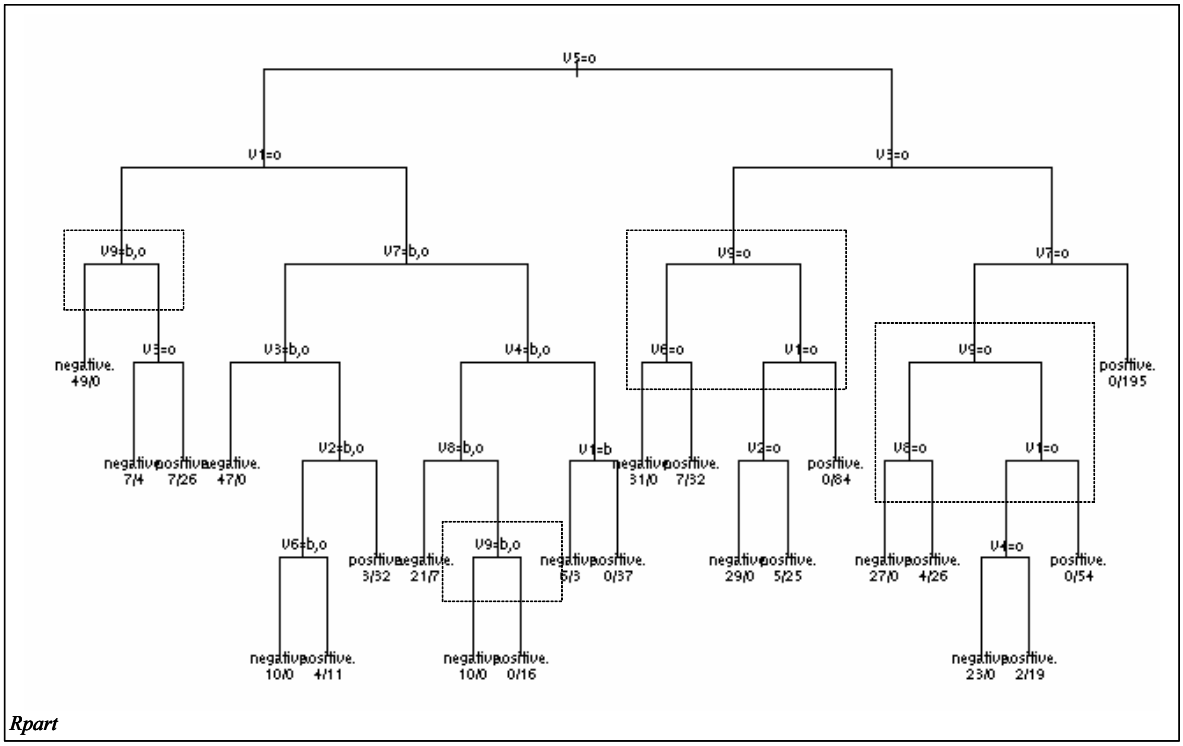


Figura 3.2: Árvore de decisão construída a partir dos dados de nível *0* (iteração *9*)

Finalmente, a Figura 3.4 é representada a árvore obtida para o conjunto de dados de nível *1*. Observa-se que o algoritmo de aprendizagem confere ao novo atributo um maior poder de discriminação, pois coloca-o na raiz da árvore. Podemos também observar que esta árvore é menos complexa do que a obtida na Figura 3.3. Obtemos uma complexidade de *37* nós contra os 45 da anterior.

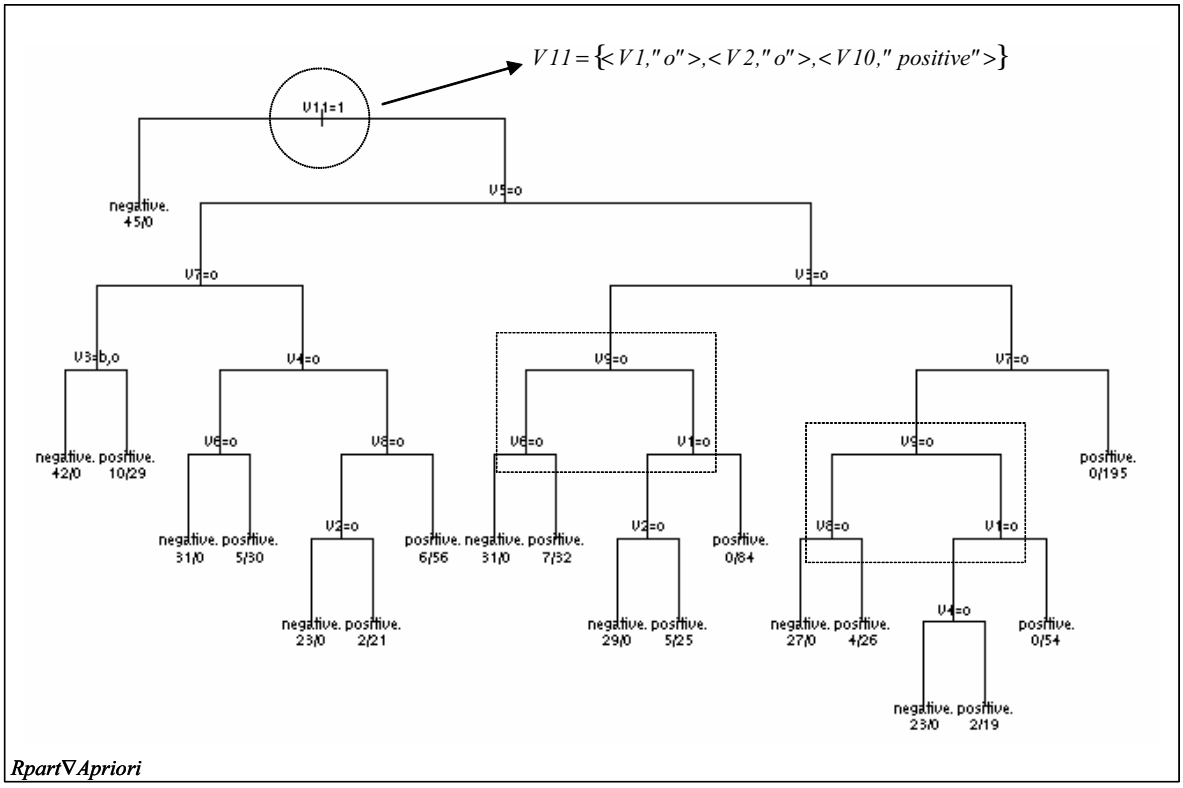


Figura 3.3: Árvore de decisão gerada pela metodologia proposta (iteração *9*)

4 Resultados experimentais

Neste capítulo realizamos testes experimentais ao método que propusemos no terceiro capítulo. Avaliamos um conjunto de heurísticas de ordenação e filtros de selecção dos conjuntos de termos frequentes. Estes testes assentam na comparação da performance deste método com a aprendizagem convencional de árvores de decisão, através da análise dos níveis de exactidão das previsões e da complexidade dos modelos gerados.

4.1 Ambiente de modelação das experiências

As experiências empíricas foram realizadas recorrendo a duas implementações de algoritmos de aprendizagem, um algoritmo de descoberta de regras de associação e outro de aprendizagem de árvores de decisão. Para a extracção dos conjuntos de termos frequentes recorremos à implementação do algoritmo *Apriori* disponibilizada pelo *WEKA* (Witten e Frank, 1999), software de data *mining* desenvolvido pela Universidade de *Waikato*, Nova Zelândia. Para a aprendizagem das árvores de decisão recorremos à implementação *rpart* disponibilizada como uma biblioteca do "R" (*www.r-project.org*), dialecto da conhecida linguagem de programação "S".

4.2 Conjuntos de dados

O método proposto foi testado em cinco conjuntos de dados, dois de domínios reais (*TicTacToe* e *Promoters*) e três de domínios artificiais (*Monk1*, *Monk2* e *Monk3*). Todos os conjuntos de dados foram retirados do repositório de dados da Universidade de Califórnia Irvine e são descritos por Blake *et al.* (1999). Os critérios de selecção dos conjuntos de dados foram os seguintes: primeiro, seleccionar conjuntos de dados que reflectissem problemas de classificação e segundo, que esses conjuntos de dados apenas fossem compostos por atributos nominais.

Na tabela 4.1 estão descritas as características dos conjuntos de dados, nomeadamente, o nome dos conjuntos de dados (coluna 1), o número de observações ou instâncias (coluna 2), o número de atributos (coluna 3) e o número de classes (coluna 4). Uma caracterização mais pormenorizada destes dados é apresentada no Apêndice A.

Tabela 4.1: Conjuntos de dados considerados no estudo empírico

Conjunto de Dados	Casos	Atributos	Classes
Tic Tac Toe	958	9	2
Monk1	432	6	2
Monk2	432	6	2
Monk3	432	6	2
Promoters	106	57	2

4.3 Filtro de conjuntos frequentes

Os conjuntos de termos frequentes foram extraídos a partir da execução do algoritmo *Apriori* e filtrados de acordo com as estratégias definidas no terceiro capítulo. Nos resultados apresentados na Tabela 4.2, apenas apresentamos os resultados obtidos pela segunda estratégia de filtro poda. Esta estratégia consiste: em primeiro lugar, na eliminação de todos os conjuntos de termos frequentes que não contenham um dos termos igual a um dos valores do atributo classe; e em segundo, na aplicação do teste de qui-quadrado entre a projecção de cada conjunto de termos frequente sobre o conjunto de treino e o atributo classe, para avaliar da independência estatística dos conjuntos de termos face à classe.

O processamento do algoritmo *Apriori* foi implementado definindo como parâmetros de entrada, o valor de *5%* para o suporte mínimo e de *50.000* para o número de regras a serem geradas[2], para todos os conjuntos de dados à excepção do *Promoters*. Para esse último, apenas definimos o suporte mínimo e não o número de regras.

Os resultados obtidos pelas estratégias de filtro estão contidos na Tabela 4.2. Nesta tabela são descritos os nomes dos conjuntos de dados (coluna 1), o suporte médio

[2] A fixação de um número elevado de regras deve-se às particularidades desta implementação do Apriori, pois contém um parâmetro de entrada que permite controlar o número de regras a gerar. Este parâmetro contém o valor por defeito de 10 e se não for alterado apenas serão extraídos conjuntos de termos frequentes necessários para gerar esse número de regras

dos conjuntos (coluna 2), o número de conjuntos após a aplicação das técnicas de filtro (coluna 3) e o percentual de redução de conjuntos após técnicas de filtro (coluna 4).

Tabela 4.2: Filtro de conjunto de termos frequentes

Conjuntos de Dados (1)	Suporte (2)	Conj. Gerados (3)	Conj. Finais (4)	% Redução (5)
Tic Tac Toe	0,08	1412	251	82,2%
Promoters	0,25	84	25	70,3%
Monk1	0,12	467	56	88,0%
Monk2	0,07	464	27	94,1%
Monk3	0,13	471	100	78,8%

As estratégias de filtro de poda conduziram a reduções entre os *70%* e *94%*, dos conjuntos de termos frequentes a analisar. Para garantir que neste processo não seriam eliminados conjuntos de termos frequentes interessantes, mantivemos experiências paralelas o universo de todos conjuntos de termos frequentes.

4.4 Resultados com as diferentes heurísticas de ordenação

Após a fase inicial de extracção dos conjuntos de termos frequentes e da redução do seu número, vamos seleccionar os mais interessantes. Para cada heurística de ordenação referenciada no terceiro capítulo, vamos seleccionar um determinado número de conjuntos de termos frequentes, estender os conjuntos de dados pela introdução dos novos atributos e verificar o comportamento das árvores de decisão geradas. Todos os resultados das experiências realizadas serão resumidos em duas tabelas, uma com a descrição das experiências e a outra com a descrição dos testes de significância entre os efectuados. Estas tabelas vão ser apresentadas nos próximos parágrafos.

Na primeira tabela são apresentados os resultados das experiências realizadas. São descritos, o número médio de atributos construídos (coluna 2); os erros médios obtidos no conjunto de dados original e estendido e o ganho em pontos percentuais (coluna 3). Estes indicadores são descritos pela sua média e desvio-padrão. Também são descritos, o tipo de ganhos obtido (coluna 9, 10 e 11), e finalmente, um indicador da complexidade das árvores de decisão geradas (coluna 11 e 12).

Na segunda tabela são resumidos os testes de significância realizados. Estes testes comparam, as diferenças entre os erros médios obtidos pelos modelos gerados a partir dos conjuntos de dados original, com aqueles obtidos nos conjuntos de dados estendidos por novos atributos. A segunda coluna da tabela contém o valor da estatística *p-value* para cada conjunto de dados. Esta estatística corresponde à probabilidade da estatística de teste tomar um valor igual ou mais extremo do que, de facto, foi observado. Se esta estatística for superior a 0.05, não poderemos rejeitar a hipótese dos erros obtidos pelos dois modelos serem iguais.

Nas próximas secções, e para cada heurística de ordenação, são apresentados os resultados das experiências realizadas com o terceiro filtro de selecção de conjuntos de termos frequentes, por ter sido aquele que permitiu obter melhores resultados. As restantes experiências são descritas nos apêndices respectivos.

4.4.1 Coeficiente de entropia

Nestas experiências, como é evidenciado pela Tabela 4.3, foram obtidos ganhos de performance em quatro dos cinco conjuntos de dados. No total das *50* iterações realizadas, *10* iterações por conjunto de dados, apenas foram registadas perdas em 6 iterações. No conjunto de dados *Monk1*, o conceito objectivo foi completamente identificado e no conjunto *Monk3* não foi possível obter ganhos de performance nem redução da complexidade dos modelos gerados. Os modelos gerados segundo o método proposto são menos complexos nos conjuntos de dados *TicTacToe* e *Monk1*. Para o conjunto de dados *Monk2*, os ganhos de performance foram obtidos à custa do aumento da complexidade das árvores de decisão

Os testes de significância (Tabela 4.4) realizados às diferenças entre os erros obtidos pelos dois modelos indicam-nos que os dois tipos de erro são estatisticamente diferentes em todos os conjuntos de dados à excepção do conjunto *Monk3*. Neste conjunto de dados o modelo gerado pela árvore de decisão é o mesmo para os dois conjuntos de dados, original e estendido.

Tabela 4.3: Resultados recorrendo ao coeficiente de entropia com o terceiro filtro de ordenação

Conjuntos de Dados (1)	Atrib. (2)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (3)	d.p. (4)	Méd (5)	d.p. (6)	Méd (7)	d.p. (8)	G> 0 (9)	G= 0 (10)	G< 0 (11)	Arv 0 (12)	Arv 1 (13)
Tic Tac Toe	5,0	0,09	0,02	0,06	0,03	0,033	0,05	8	0	2	43	39
Promoters	3,0	0,31	0,11	0,10	0,11	0,205	0,12	9	1	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	26,6	0,28	0,08	0,19	0,06	0,088	0,07	5	1	4	31	41
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

Tabela 4.4: Testes de significância para o coeficiente de entropia

Conjunto de Dados	p-value
Tic Tac Toe	0,049
Promoters	0,001
Monk1	0,000
Monk2	0,004
Monk3	NA

Como exemplo dos modelos gerados, apresentamos as árvores de decisão geradas para o conjunto de dados *Monk1* na iteração *9* (Figuras 4.1 e 4.2). Dos 6 atributos inseridos no conjunto de dados, foram seleccionados *2* novos atributos para construção da árvore de decisão. Os novos atributos que foram integrados na árvore de decisão foram construídos com base nos seguintes conjuntos de termos frequentes:

$$V13 = \{< V1,"a">,< V2,"a">,< V7,"um">\}$$
$$V12 = \{< V1,"c">,< V2,"c">,< V7,"um">\}$$

O modelo de árvore de decisão gerado a partir conjunto de dados estendido possui um nível máximo de exactidão. Isto acontece pois os conjuntos de termos frequentes seleccionados conseguem captar com o conceito objectivo (ver Apêndice A) subjacente a este conjunto de dados.

O algoritmo de aprendizagem de árvores de decisão ao seleccionar estes atributos para construir a árvore de decisão está a estender a linguagem de representação usada, está a usar a linguagem de representação das regras de associação.

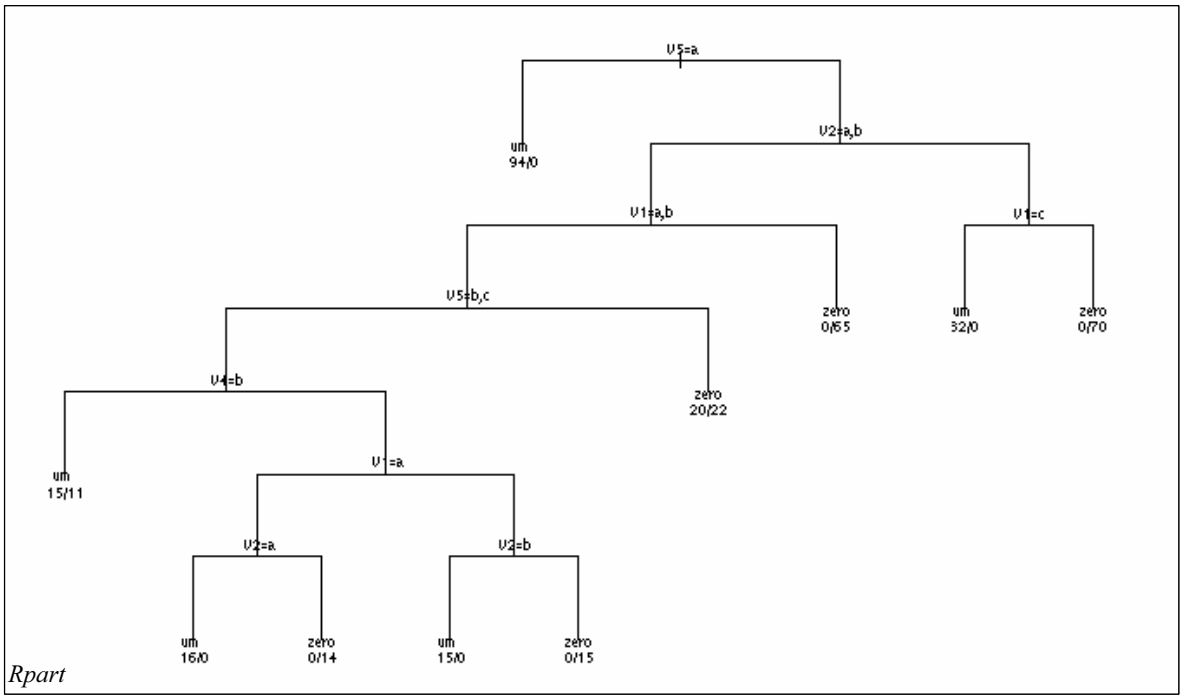


Figura 4.1: Árvore decisão no conjunto de dados (original) *Monk1* (iteração 9)

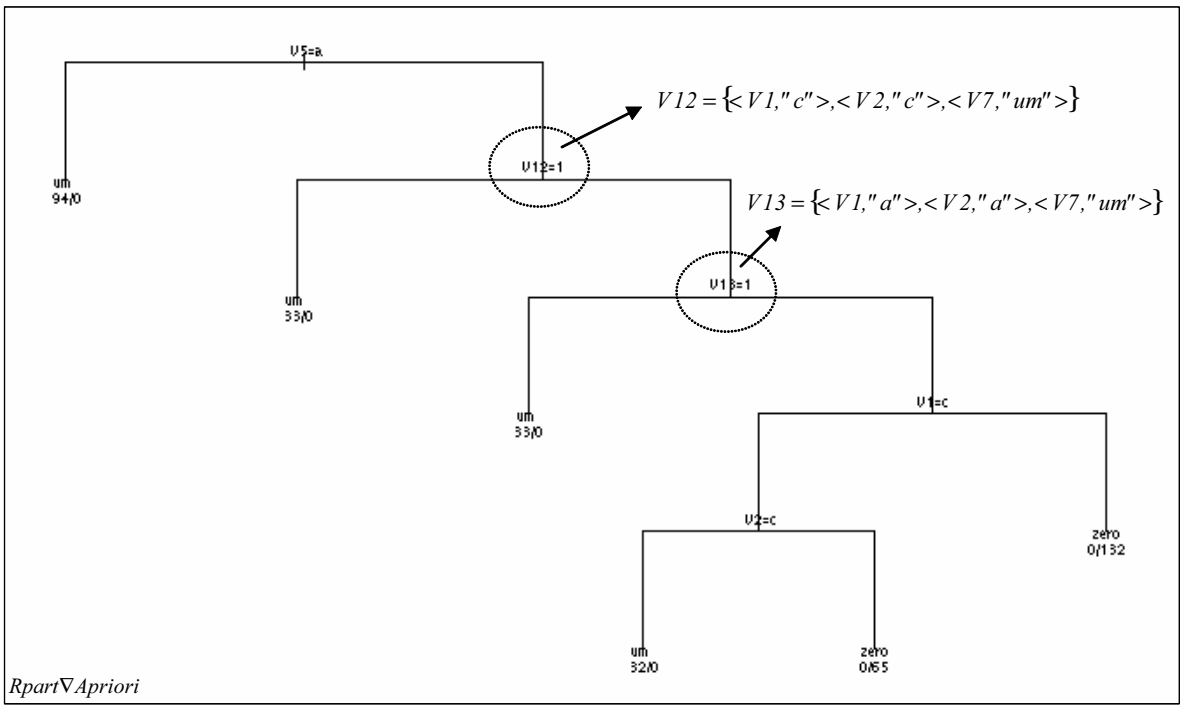


Figura 4.2: Árvore decisão obtida no conjunto de dados (estendido) *Monk1* com o coeficiente de entropia e filtro de selecção 3 (iteração 9)

4.4.2 Ganho de informação

A aplicação da heurística ganho de informação permitiu obter ganhos de performance em quatro dos cinco conjuntos de dados. A excepção fica para o conjunto *Monk3* como ser observado pelos resultados da Tabela 4.5. Não foram registadas perdas em nenhuma das experiências realizadas. Das *50* experiências houve *14* com ganhos nulos, *10* das quais no conjunto de dados *Monk3*. Neste último conjuntos dados, apesar de não termos conseguido obter ganhos de performance, reduzimos a complexidade das árvores de decisão geradas.

Nestas experiências, todos os conjuntos frequentes disponíveis para selecção foram integrados como novos atributos, não houve qualquer exclusão pela aplicação do filtro. No conjunto de dados *TicTacToe* foram construídos *251* atributos, a partir dos *251* conjuntos de termos frequentes disponíveis e no *Monsk3* foram construídos *100* novos atributos.

Os testes de significância apresentados na Tabela 4.6 indicam-nos que os erros obtidos pelos dois modelos são estatisticamente diferentes, para um nível de confiança de *95%*.

Tabela 4.5: Resultados recorrendo ao ganho de informação com o terceiro filtro de selecção

Conjuntos de Dados (1)	Atrib. (2)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (3)	d.p. (4)	Méd (5)	d.p. (6)	Méd (7)	d.p. (8)	G> 0 (9)	G= 0 (10)	G< 0 (11)	Arv 0 (12)	Arv 1 (13)
Tic Tac Toe	251,4	0,09	0,02	0,02	0,02	0,068	0,04	10	0	0	43	27
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	56,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	27,4	0,28	0,08	0,19	0,06	0,086	0,07	8	2	0	31	41
Monk3	100,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	7

Tabela 4.6: Testes de significância para o ganho de informação

Conjunto de Dados	p-value
Tic Tac Toe	0,000
Promoters	0,002
Monk1	0,000
Monk2	0,004
Monk3	NA

As árvores de decisão geradas para o conjunto de dados *Monk1* na terceira iteração estão representadas pelas Figuras 4.4 e 4.5. Na árvore de decisão representada pela Figura 4.5 foi usado na sua construção um novo atributo (*V66*).

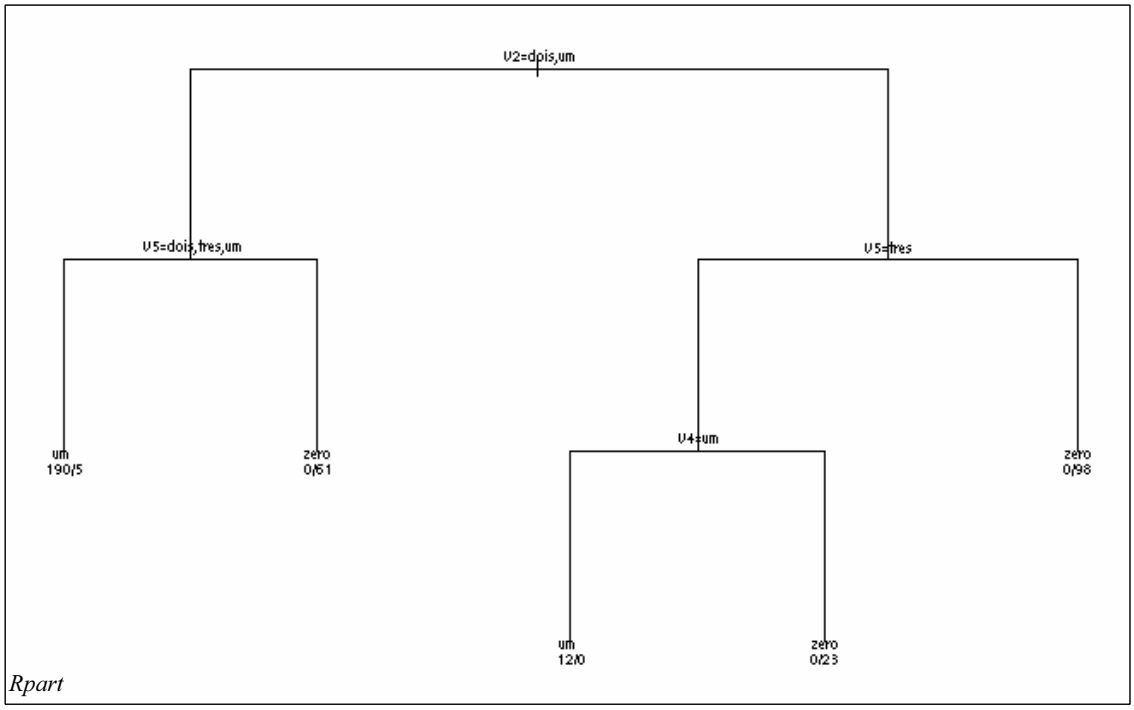


Figura 4.3: Árvore de decisão obtida no conjunto de dados (original) *Monk3* (iteração 3)

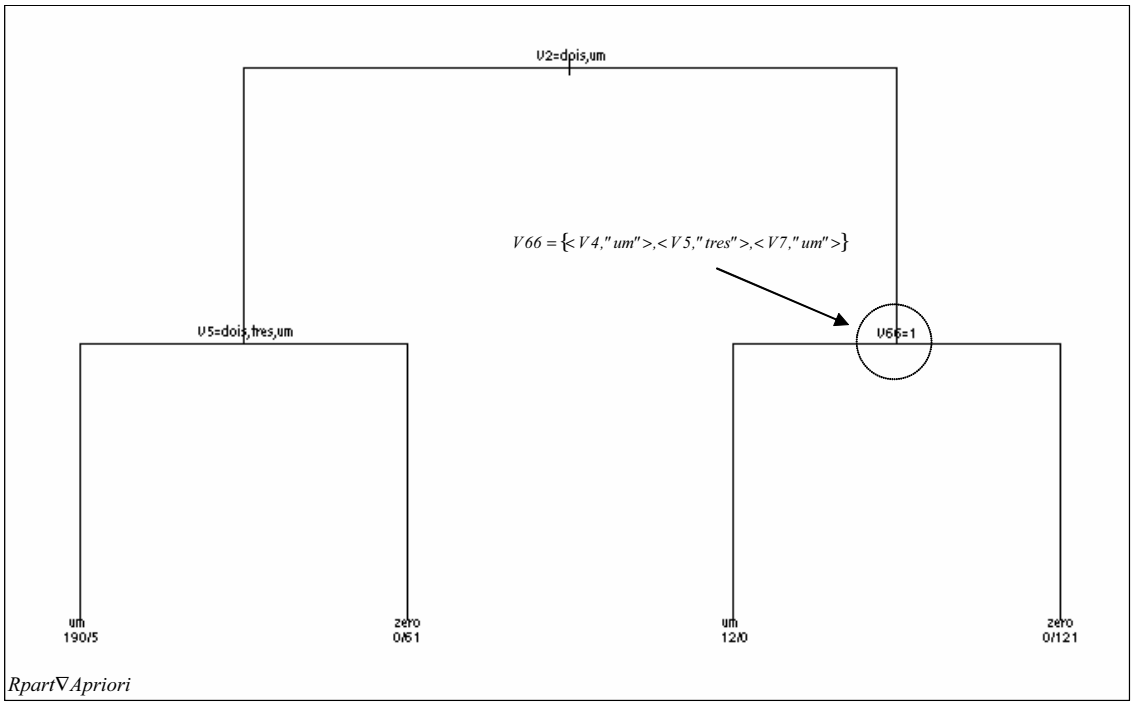


Figura 4.4: Árvore de decisão obtida no conjunto de dados (estendido) *Monk3* com o ganho de informação e filtro de selecção 3 (iteração 3)

4.4.3 Rácio de informação

O recurso ao rácio informação como heurística de ordenação resultou em ganhos de performance em todos os conjuntos de dados à excepção do conjunto de dados Monk3, não sendo registadas perdas de performances em nenhuma das experiências realizadas.

Nos conjuntos de dados, *TicTacToe* e *Monk1*, o método que propomos gera modelos que conseguem obter níveis de exactidão máximos e árvores menos complexas, como pode ser observado pela Tabela 4.7. No conjunto *Promoters* obtivemos ganhos significativos ao nível da performance à custa do aumento da complexidade dos modelos gerados. No conjunto *Monk3* não foram observados, nem ganhos de performance nem redução substancial da complexidade das árvores de decisão geradas, apesar de numa das iterações, a complexidade da árvore de decisão ter sido reduzida.

A comparação estatística das diferenças entre os dois tipos de erro (Tabela 4.8), permite-nos concluir que os modelos produzem resultados significativamente diferentes para um nível de confiança de *95%*. Para o conjunto *Monk3* estes testes não estão disponíveis porque os erros produzidos pelos dois modelos são iguais.

Tabela 4.7: Resultados recorrendo à função rácio de informação com o terceiro filtro de ordenação

Conjuntos de Dados (1)	Atrib. (2)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (3)	d.p. (4)	Méd (5)	d.p. (6)	Méd (7)	d.p. (8)	G> 0 (9)	G= 0 (10)	G< 0 (11)	Arv 0 (12)	Arv 1 (13)
Tic Tac Toe	26,3	0,09	0,02	0,01	0,01	0,082	0,03	10	0	0	43	37
Promoters	18,1	0,31	0,11	0,10	0,11	0,215	0,13	9	1	0	6	7
Monk1	31,6	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	27,4	0,28	0,08	0,19	0,06	0,086	0,07	8	2	0	31	41
Monk3	30,8	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

Tabela 4.8: Testes de significância para o rácio de informação

Conjunto de Dados	p-value
Tic Tac Toe	0,000
Promoters	0,002
Monk1	0,000
Monk2	0,004
Monk3	NA

Como exemplo da aplicação desta heurística recorremos à árvore de decisão gerada para o conjunto de dados *TicTacToe* representadas na Figura 4.6. Na iteração *9*, foram inseridos *26* novos atributos, sendo *10* representados na árvore gerada.

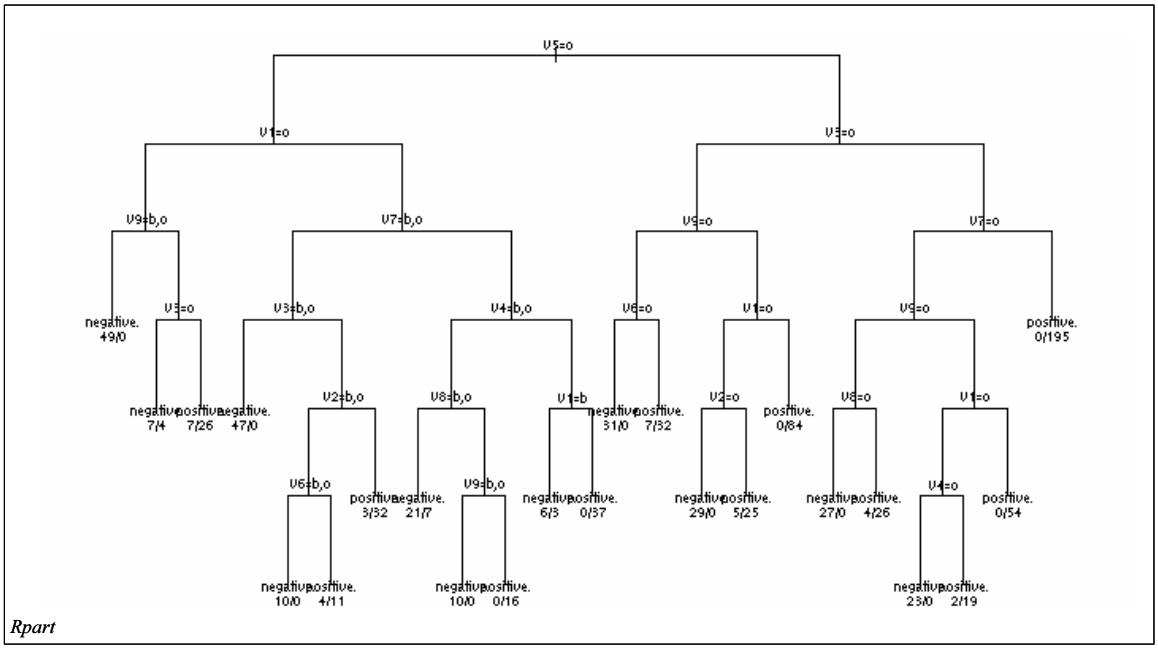


Figura 4.5: Árvore de decisão obtida no conjunto de dados TicTacToe (iteração 9)

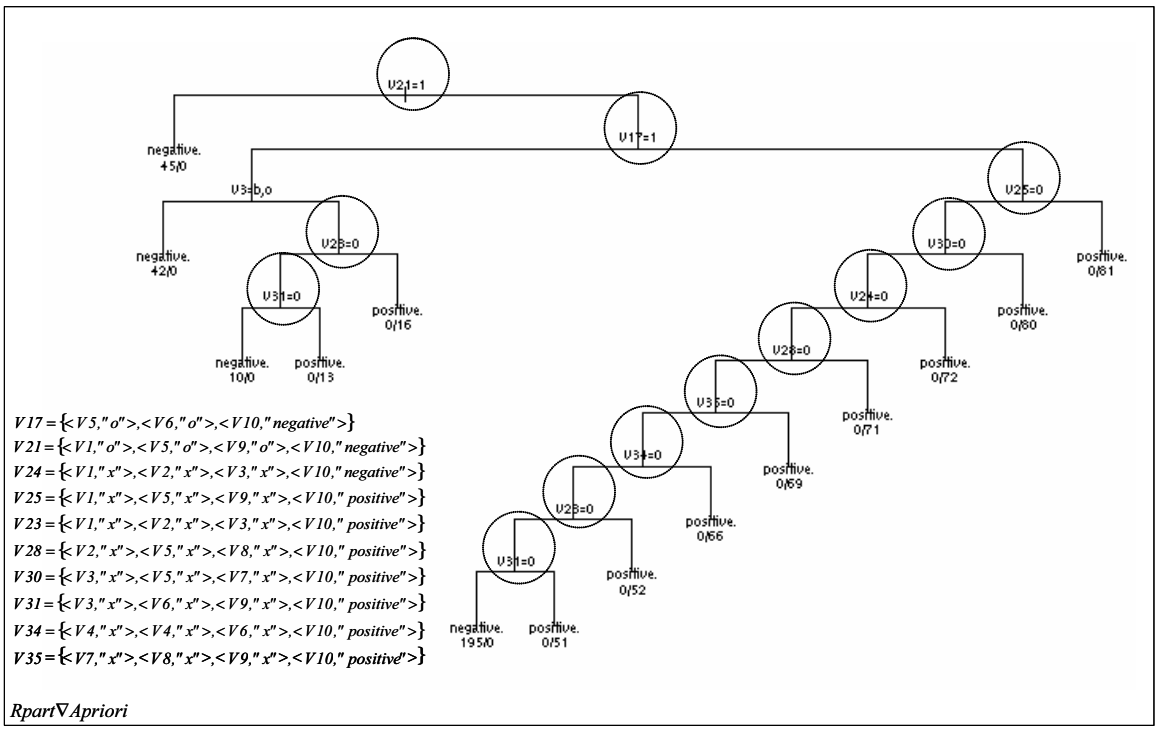


Figura 4.6: Árvore de decisão obtida no conjunto de dados (estendido) *TicTacToe* com o rácio de informação e o filtro de selecção 3 (iteração 3)

4.4.4 Coeficiente de G*ini*

Á semelhança do que sucedeu com a heurística de ordenação, ganho de informação, na aplicação do coeficiente de *Gini,* foram seleccionados todos os conjuntos de termos frequentes para estender o conjunto de dados. Como pode ser observado pela Tabela 4.9, a aplicação deste método com esta heurística proporcionou bons resultados, pois conseguimos obter ganhos de performance em quatro dos cinco conjuntos de dados e não hão houve perdas performance em nenhuma das experiências realizadas.

Para os conjuntos, *TicTacToe* e *Monk1*, o método proposto registou bons ganhos de performance e simultaneamente, uma redução da complexidade dos modelos gerados. Nos conjuntos de dados, *Promoters* e *Monk2*, os ganhos de performance foram conseguidos com um aumento da complexidade dos modelos gerados. No conjunto de dados *Monk3* apenas foi possível reduzir a complexidade das árvores de decisão.

Tabela 4.9: Tabela de resultados recorrendo ao coeficiente de *Gini* e com o filtro de ordenação 3

Conjuntos de Dados (1)	Atrib. (2)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (3)	d.p. (4)	Méd (5)	d.p. (6)	Méd (7)	d.p. (8)	G> 0 (9)	G= 0 (10)	G< 0 (11)	Arv 0 (12)	Arv 1 (13)
Tic Tac Toe	251,4	0,09	0,02	0,02	0,02	0,068	0,04	10	0	0	43	27
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	56,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	27,4	0,28	0,08	0,19	0,06	0,086	0,07	8	2	0	31	41
Monk3	100,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	7

Tabela 4.10: Testes de significância para o coeficiente de *Gini*

Conjunto de Dados	p-value
Tic Tac Toe	0,000
Promoters	0,002
Monk1	0,000
Monk2	0,004
Monk3	NA

Nas Figuras 4.7 e 4.8 estão representadas duas árvores de decisão, para o conjunto de dados *Promoters,* uma para o conjunto de dados original e outra para o conjunto de dados estendido pelos novos atributos. A árvore de decisão que resulta da aplicação da

aplicação do método proposto, coloca um novo atributo na raiz da árvore, conferindo-lhe assim, um maior poder discriminativo.

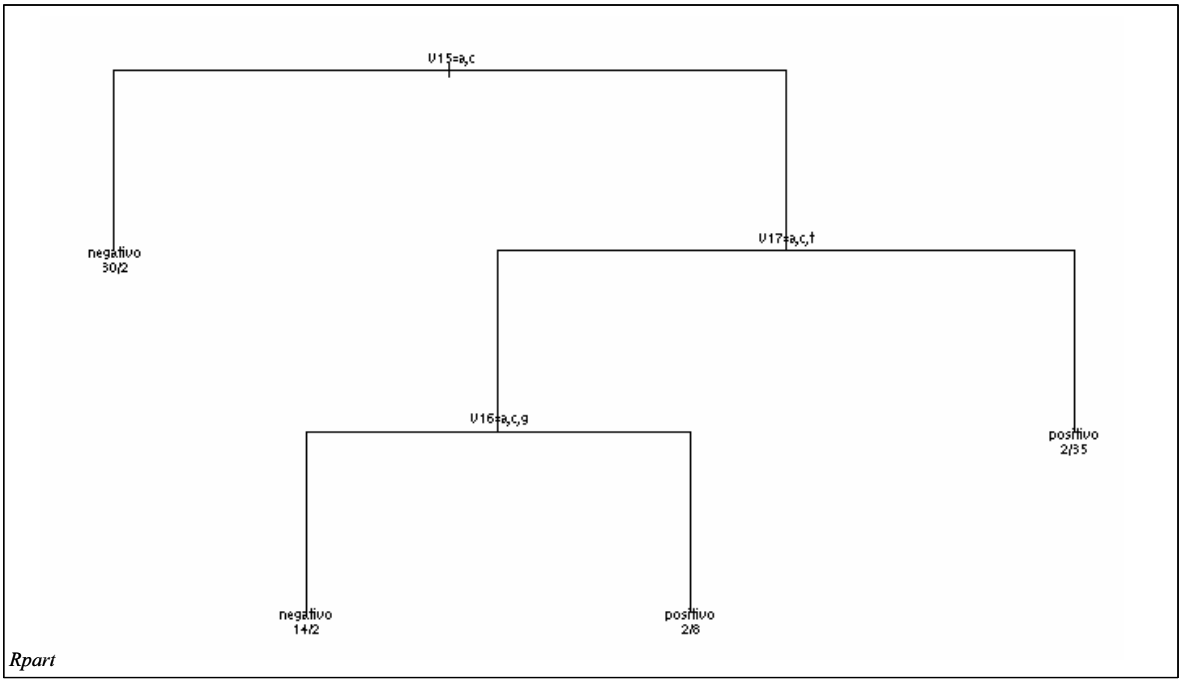


Figura 4.7: Árvore de decisão obtida no conjunto de dados (original) *Promoters* (iteração 5)

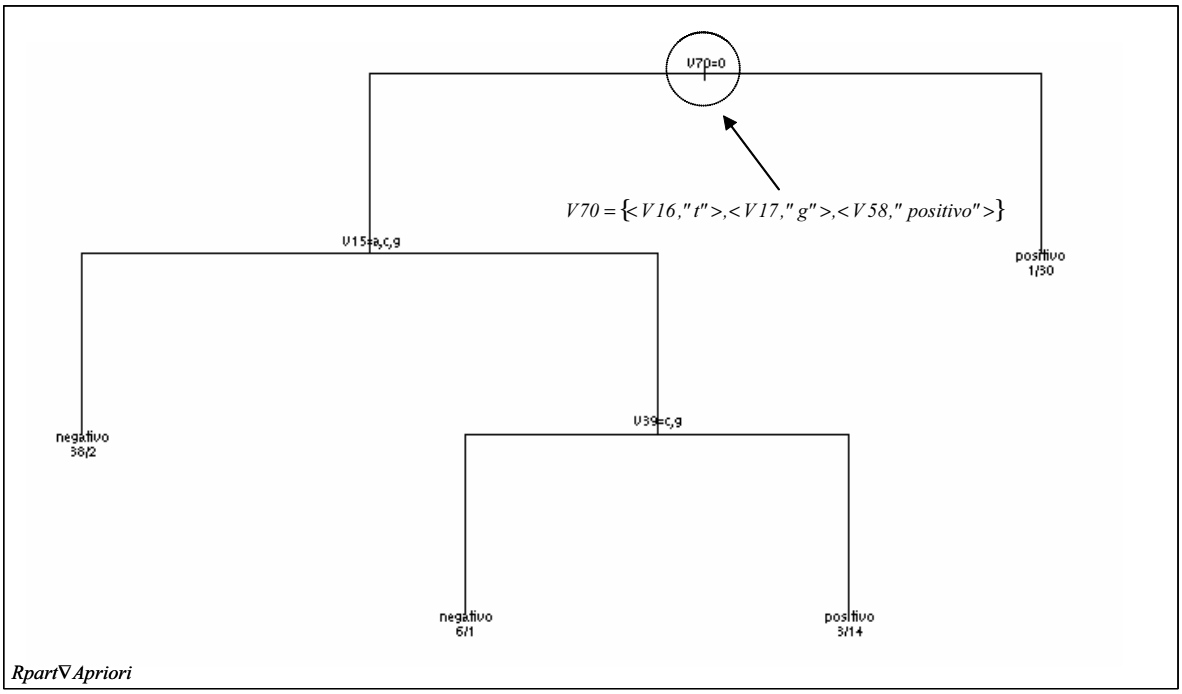


Figura 4.8: Árvore de decisão obtida no conjunto (estendido) *Promoters* com o coeficiente de *Gini.* e o filtro de selecção *3* (iteração 5)

4.4.5 Índice de associação de *Kruskal-Goodman*

Os resultados obtidos com o índice de associação de *Kruskal-Goodman* estão na linha dos obtidos com as anteriores heurísticas. De acordo com os resultados descritos na Tabela 4.11, obtivemos ganhos de performance em todos os conjuntos de dados à excepção do conjunto *Monk3*. Não existiram perdas de performance no conjunto das *50* experiências realizadas e em *14* destas obtivemos resultados nulos. Nos conjuntos de dados, *TicTacToe* e *Monk1*, foi possível obter ganhos de performance e redução da complexidade das árvores. Nos conjuntos, *Monk2* e *Promoters*, houve ganhos de performance e aumento da complexidade das árvores (Tabela 4.11).

A comparação estatística das diferenças entre os dois tipos de erro (Tabela 4.12), permite-nos concluir que os nossos modelos produzem resultados significativamente diferentes para um nível de confiança de *95%*

Tabela 4.11: Resultados recorrendo o índice de *Kruskal-Goodman* com o filtro de selecção *3*

Conjuntos de Dados (1)	Atrib. (2)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (3)	d.p. (4)	Méd (5)	d.p. (6)	Méd (7)	d.p. (8)	G> 0 (9)	G= 0 (10)	G< 0 (11)	Arv 0 (12)	Arv 1 (13)
Tic Tac Toe	3,8	0,09	0,02	0,06	0,03	0,034	0,04	8	2	0	43	37
Promoters	15,8	0,31	0,11	0,10	0,11	0,215	0,13	9	1	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,20	0,06	0,074	0,08	9	1	0	31	42
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

Tabela 4.12: Testes de significância para os resultados obtidos com o índice de *Kruskal-Goodman*

Conjunto de Dados	p-value
Tic Tac Toe	0,036
Promoters	0,000
Monk1	0,000
Monk2	0,015
Monk3	NA

O modelo gerado na iteração *9* para o conjunto de dados *Monk2* é apresentado na Figura 4.10 e árvore de decisão para o conjunto de dados original está representada na Figura 4.9.Foram seleccionados quatro novos atributos para integrar a nova árvore de decisão e um foi posicionado na raiz da árvore.

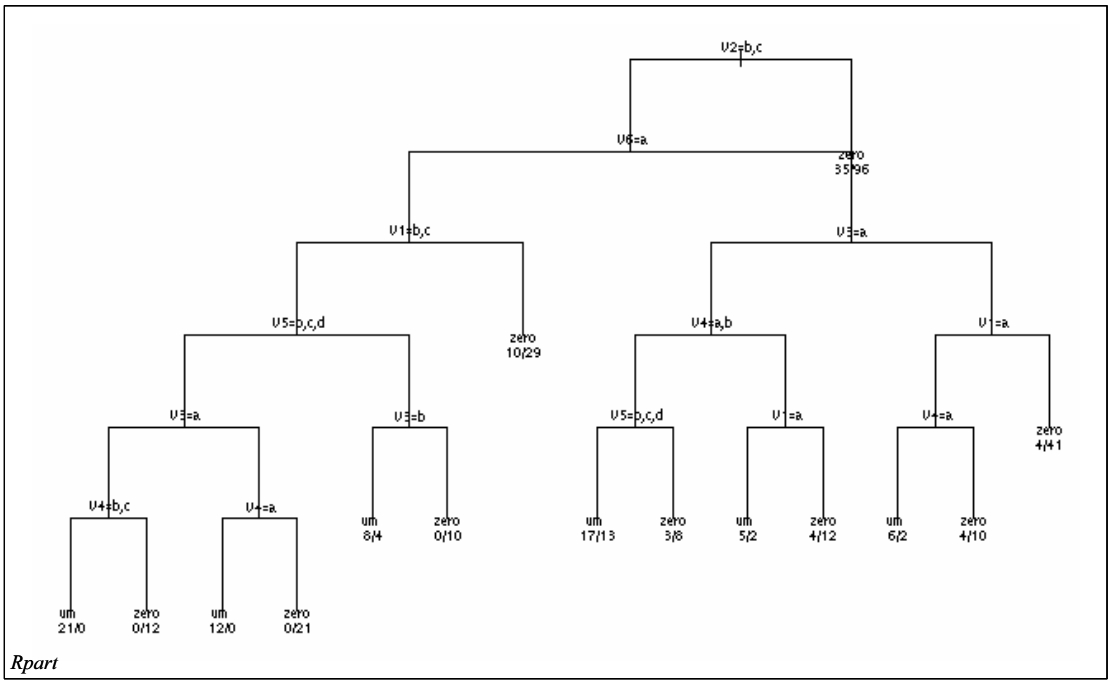


Figura 4.9: Árvore de decisão obtida no conjunto *Monk2* (original) na iteração *8*

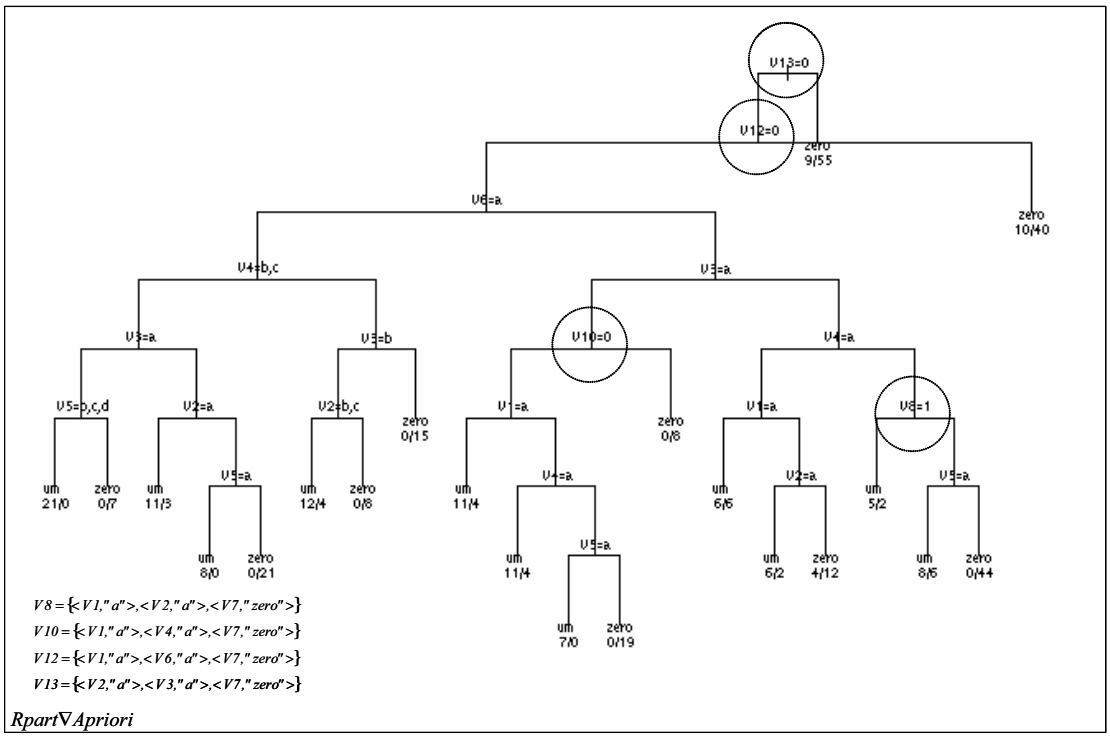


Figura 4.10: Árvore decisão no conjunto (estendido) *Monk2* com o índice de *Kruskal-Goodman* e o filtro de selecção *3* (iteração *8*)

4.4.6 Coeficiente de interesse

Nas experiências realizadas com o coeficiente de interesse apenas recorremos ao filtro de selecção *2* e que nestas experiências é igual ao filtro 3. De acordo com os resultados apresentados na Tabela 4.13, obtivemos ganhos médios de performance em todos os conjuntos de dados à excepção do conjunto *Monsk3*. Os ganhos são reduzidos em dois dos conjuntos de dados, *TicTacToe* e *Monk2*, cerca de *1,2* e *2,8* pontos percentuais, respectivamente. Nos restantes conjuntos de dados, os ganhos foram significativos, *22,9* pontos percentuais no conjunto *Monk1* (*100%* de exactidão) e *19,7* pontos percentuais no conjunto de dados *Promoters*.

Para os conjuntos de *TicTacToe* e *Monk2*, as diferenças entre os dois tipos de erros não são estatisticamente significantes, para um nível de confiança de *95%*, e portanto, os resultados obtidos pelo modelo não são suficientemente robustos (Tabela 4.14).

Tabela 4.13: Resultados recorrendo ao coeficiente de interesse com o filtro de selecção número 2

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,08	0,04	0,014	0,05	7	0	3	43	40
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	6,0	0,28	0,08	0,25	0,09	0,028	0,12	4	1	5	31	41
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

Tabela 4.14: Testes de significância para os resultados obtidos com o coeficiente de interesse

Conjunto de Dados	t- pairwise tests p-value
Tic Tac Toe	0,430
Promoters	0,002
Monk1	0,000
Monk2	0,472
Monk3	NA

4.4.7 Coeficiente de *Laplace*

A utilização do coeficiente de *Laplace* como heurística de ordenação permitiu-nos obter ganhos de performance em todos os conjuntos de dados à excepção do conjunto *Monk3*, como pode ser observado pela Tabela 4.15. Das 50 experiências realizadas, obtivemos perdas de performance em 3 experiências, resultados nulos em 4 e ganhos de performance nas restantes. Todas as experiências com perdas de performance foram obtidas com o conjunto de dados *Monk2*. Nos conjuntos de dados *TicTacToe* e *Monk1* conseguimos ganhos de performance e uma redução da complexidade das árvores de decisão. Nos outros conjuntos, *Promoters* e *Monk2*, os ganhos de performance foram conseguidos à custa do aumento de complexidade das árvores de decisão.

A avaliação das diferenças entre os erros obtidos pelos dois modelos através dos testes de significância permitiu-nos concluir que, estes não são estatisticamente diferentes para os conjuntos, *Monk2* e *Monk3*, sendo-o contudo, para os restantes conjuntos de dados. (Tabela 4.16).

Tabela 4.15: Resultados recorrendo ao coeficiente de *Laplace*

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,02	0,03	0,067	0,03	10	0	0	43	38
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,26	0,07	0,021	0,10	5	1	4	31	40
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

Tabela 4.16: Testes de significância para os resultados obtidos com o coeficiente de *Laplace*

Conjunto de Dados	P-Value
Tic Tac Toe	0,000
Promoters	0,002
Monk1	0,000
Monk2	0,528
Monk3	NA

4.4.8 Coeficiente de correlação

Este método integrando o coeficiente de correlação como heurística de ordenação dos conjuntos de termos frequentes permitiu obter ganhos de performance nos conjuntos de dados: *Promoters*; *Monk1* e *Monk2* (Tabela 4.17). No conjunto de dados *TicTacToe* perdemos performance face ao método convencional de aprendizagem de árvores de decisão e no conjunto *Monk3*, os resultados foram nulos, à semelhança dos resultados obtidos com as restantes heurísticas.

Os testes de significância demonstraram que os resultados obtidos por este método, não são estatisticamente diferentes dos obtidos pelo modelo de aprendizagem convencional de árvores de decisão, para os conjuntos: *TicTacToe*, *Monk2* e *Monk3*. Para o conjunto de dados *Promoters*, os dois tipos de erros são estatisticamente diferentes para um nível de confiança de *95%*.

Tabela 4.17: Resultados recorrendo ao coeficiente de correlação

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,10	0,05	-0,008	0,05	3	2	5	43	41
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,26	0,06	0,021	0,10	5	1	4	31	41
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

Tabela 4.18 Testes de significância para os resultados obtidos com o coeficiente de correlação

Conjunto de Dados	P-Value
Tic Tac Toe	0,613
Promoters	0,002
Monk1	0,000
Monk2	0,526
Monk3	NA

4.5 Resumo dos resultados

O método que propusemos foi experimentado com oito possíveis heurísticas de ordenação de conjuntos de termos frequentes. Em algumas, foram obtidos melhores resultados do que em outras, mas em termos gerais, com todas foram obtidos bons resultados, reforçando o interesse da descoberta de padrões nos dados como uma ferramenta importante para a construção de modelos mais robustos.

Na Tabela 4.19 estão descritos os resultados obtidos para cada uma das heurísticas de ordenação. Os modelos gerados foram avaliados na sua performance e na sua complexidade. A performance dos modelos foi medida pelo nível de exactidão das suas estimativas. A relevância dos resultados obtidos foi aferida pelos testes de significância descritos na Tabela 4.20.

Os maiores níveis de exactidão foram conseguidos com a heurística **Rácio de Informação** e de uma forma geral, foram as medidas baseadas na entropia, aquelas que proporcionaram melhores resultados. Além destas últimas, o coeficiente de *Gini* e o índice de *Kruskal-Goodman* também proporcionaram bons resultados.

A análise efectuada aos resultados obtidos para cada conjunto de dados permite-nos concluir que todas as heurísticas de ordenação obtiveram níveis de exactidão máximos no conjunto de dados *Monk1.* Nos conjuntos de dados, *TicTacToe* e *Monk2*, apesar das diferentes heurísticas terem obtido melhores resultados comparativamente ao modelo gerado a partir do conjunto de dados original, alguns dos resultados não foram considerados estatisticamente diferentes dos originais para um nível de confiança de *95%.* Finalmente, para no conjunto de dados *Monk3* nenhum modelo conseguiu obter níveis de exactidão melhores do que o dos modelos originais.

A performance dos modelos segundo a sua complexidade varia de heurística para heurística, não se podendo eleger aquela heurística que produz as árvores menos complexas. Para os conjuntos de dados *TicTacToe*, *Monk1* e *Monk3*, foi possível obter árvores menos complexas do que aquelas obtidas com os conjuntos de dados originais. Nos restantes conjuntos de dados o este método implicou a construção de árvores mais complexas.

A aplicação deste método com a heurística rácio de informação permitiu obter bons resultados comparativamente aos obtidos com as outras heurísticas. Nos conjuntos de dados, *TicTacToe* e *Monk3*, e com as heurísticas, ganho de informação e coeficiente de *Gini*, foram obtidas as árvores de decisão menos complexas do que as obtidas com o recurso ao rácio de informação.

Tabela 4.19: Resumo da performance dos modelos com as diferentes heurísticas de ordenação

Conjuntos de dados / Dimensões de análise		Arv. Decisão rpart	Modelo: Heurísticas de Avaliação							
			Entropia filtro 3	Ganho filtro 3	Rácio filtro 3	Gini filtro 3	Lambda filtro 3	Interesse filtro 2	Laplace filtro 2	Correlação filtro 2
Tic tac toe	Exactidão	0,911	0,945	0,979	**0,993**	0,979	0,945	0,925	0,978	0,903
	Complexidade	43,4	39,4	27,4	36,6	27,4	36,6	39,8	38,4	40,6
Promoters	Exactidão	0,690	0,895	0,886	**0,905**	0,886	0,905	0,886	0,886	0,886
	Complexidade	5,6	7,0	6,8	7,0	6,8	7,0	6,8	6,8	6,8
Monk 1	Exactidão	0,771	1,000	1,000	**1,000**	1,000	1,000	1,000	1,000	1,000
	Complexidade	15,6	11,0	9,0	9,0	9,0	11,0	9,0	11,0	11,0
Monk 2	Exactidão	0,724	**0,813**	**0,810**	**0,810**	**0,810**	0,799	0,752	0,745	0,745
	Complexidade	31,2	41,0	40,6	40,6	40,6	41,8	41,4	40,0	41,0
Monk 3	Exactidão	0,986	0,986	0,986	0,986	0,986	0,986	0,986	0,986	0,986
	Complexidade	9,0	9,0	7,0	8,8	7,0	9,0	9,0	9,0	9,0

Tabela 4.20: Resumo dos testes de significância

Conjuntos de Dados	Heurísticas de Avaliação							
	Entropia filtro 3	Ganho filtro 3	Rácio filtro 3	Gini filtro 3	Lambda filtro 3	Interesse filtro 2	Laplace filtro 2	Correlação filtro 2
Tic tac toe	0,049	0,000	0,000	0,000	0,036	**0,430**	0,000	0,613
Promoters	0,001	0,002	0,000	0,002	0,000	0,002	0,002	0,002
Monk 1	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Monk 2	0,004	0,004	0,004	0,004	0,015	**0,472**	**0,528**	**0,526**
Monk 3	NA	NA	NA	NA	NA	NA	NA	NA

Para avaliar em simultâneo as duas componentes de avaliação dos modelos gerados, recorremos à construção de um gráfico de quadrantes que resume os resultados obtidos. O quadrante do topo esquerdo do gráfico representa a situação em que é possível melhorar a performance em termos de previsão e ao mesmo tempo reduzir a complexidade do modelo. Já o quadrante do topo direito implica uma melhoria de performance e um aumento da complexidade dos modelos. Os quadrantes inferiores implicam perdas de performance em termos de previsão, sem (esquerdo) e com (direito) redução da complexidade dos modelos.

Como exemplo da aplicação do gráfico, escolhemos as experiências realizadas com a heurística rácio de informação. Na Figura 4.11 está representado o gráfico de quadrantes e podemos observar que todos os conjuntos de dados estão acima do eixo das abcissas, significando isto que não houve perdas de performance com a aplicação do deste método. Na variedade de resultados obtidos verificamos três situações distintas: a primeira é para os conjuntos de dados, *TicTacToe* e *Monk1*, com ganhos de performance e redução da complexidade dos modelos gerados; a segunda, para os conjuntos, *Promoters* e *Monk2*, em que existem ganhos de performance e aumento da complexidade dos modelos; e por último, para o conjunto *Monk3* em que apenas existiu um pequeníssima redução da complexidade dos modelos.

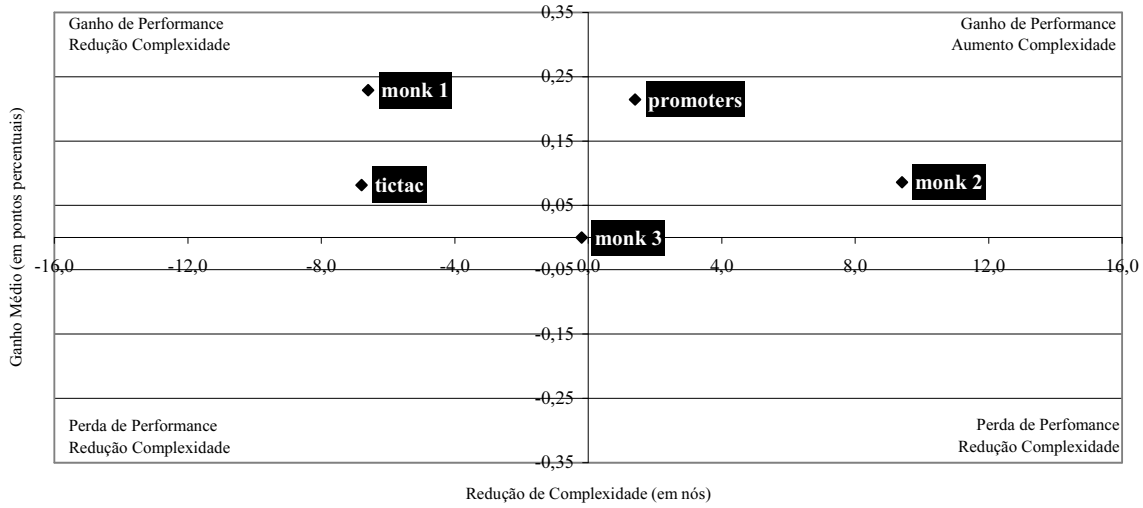


Figura 4.11: Gráfico de quadrantes para os resultados do modelo com o rácio de informação

Para além da análise dos ganhos médios de performance parece-nos útil perceber como é que o método se comportou em todas as experiências realizadas. Na tabela 4.21 apresentamos um resumo dos resultados das experiências realizadas, discriminando-as pela natureza dos resultados obtidos. A divisão é feita entre aquelas que obtiveram ganhos superiores a zero (> 0), ganhos nulos ($= 0$) ou perdas (< 0). Para as heurísticas, **Rácio de informação**, **Ganho de Informação**, **Coeficiente de *Gini*** e **Índice de associação de *Kruskal-Goodman***, não houve perdas de performance em nenhuma das experiências realizadas.

Tabela 4.21: Resumo do tipo de ganhos obtido em cada conjunto de dados e com cada heurística

Conjunto de Dados	Tipo de Resultados	Heurísticas de Avaliação							
		Entropia filtro 3	Ganho filtro 3	Rácio filtro 3	Gini filtro 3	Lambda filtro 3	Interesse filtro 2	Laplace filtro 2	Correlação filtro 2
Tic tac toe	**ganho > 0**	8	10	10	10	8	7	10	3
	ganho = 0	0	0	0	0	2	0	0	2
	ganho < 0	2	0	0	0	0	3	0	5
Promoters	**ganho > 0**	9	8	9	8	9	8	8	8
	ganho = 0	1	2	1	2	1	2	2	2
	ganho < 0	0	0	0	0	0	0	0	0
Monk 1	**ganho > 0**	10	10	10	10	10	10	10	10
	ganho = 0	0	0	0	0	0	0	0	0
	ganho < 0	0	0	0	0	0	0	0	0
Monk 2	**ganho > 0**	5	8	8	8	9	4	5	5
	ganho = 0	1	2	2	2	1	1	1	1
	ganho < 0	4	0	0	0	0	5	4	4
Monk 3	**ganho > 0**	0	0	0	0	0	0	0	0
	ganho = 0	10	10	10	10	10	10	10	10
	ganho < 0	0	0	0	0	0	0	0	0
TOTAL	**ganho > 0**	32	**36**	**37**	**36**	**36**	29	33	26
	ganho = 0	12	14	13	14	14	13	13	15
	ganho < 0	6	**0**	**0**	**0**	**0**	8	4	9
		50	**50**	**50**	**50**	**50**	**50**	**50**	**50**

4.6 Discussão

Após a definição do método e a sua experimentação, identificamos um conjunto de pontos que podem ser discutíveis, nomeadamente, o porquê do recurso aos conjuntos de termos frequentes em detrimento das regras de associação, o uso de filtros de poda e de selecção de conjuntos de termos frequentes, as heurísticas de ordenação; a morosidade do processo e a aplicabilidade dos modelos.

Recurso aos conjuntos de termos frequentes

As regras de associação são construídas tendo por objectivo encontrar relações do tipo "Um cliente que compra X também comprará Y", ou seja, regras da forma "**Se … então …**". São avaliadas pelo suporte, que representa a sua significância estatística, e pela confiança, que é a probabilidade condicionada de um conjunto ocorrer dado que outro ocorreu. Estas regras representam um tipo de conhecimento existente nos dados mas que não o único.

No método que propomos apenas recorremos aos conjuntos de termos frequentes para aferir as relações existentes entre os dados, pois através destes temos acesso ao seu

suporte estatístico, que pode ser usado como estratégia de poda de conjuntos. Não recorremos às regras de associação, pois para o nosso estudo aquelas não trazem nada de novo. A relação de causalidade existente nas regras de associação, não nos é útil, assim como o não é o parâmetro de confiança. Pretendemos analisar a relação existente entre os termos originados pelos valores do atributo classe e os termos originados pelos restantes atributos, e essa relação pode ser dada apenas pelos conjuntos de termos frequentes.

O recurso à informação contida nos conjuntos de termos frequentes em detrimento da informação contida nas regras de associação já foi estudado por outros autores. Brin *et al.* (1997) propuseram medir a significância das regras de associação através da definição de uma fronteira entre conjuntos de termos frequentes correlacionados e não correlacionados. Os algoritmos de aprendizagem de regras de associação dividem-se em duas fases, a primeira consiste na geração de conjuntos de termos frequentes e a segunda, na descoberta de regras de associação. Concentraram-se na primeira fase do algoritmo e definiram estratégias para reduzir o número de conjuntos termos frequentes gerados. Recorreram ao teste de qui-quadrado para estabelecer essa fronteira entre os conjuntos de termos correlacionados e não correlacionados. Uma das razões que invocaram para o recurso a esta estratégia foi a de pouparem tempo e espaço de processamento.

Filtros e heurísticas de ordenação

Recorremos a filtros de poda com o objectivo de reduzir o número de conjuntos de termos frequentes a serem analisados, sem contudo, eliminar conjuntos de termos que possam ter interesse para a análise. Paralelamente às experiências realizadas no grupo de conjuntos de termos filtrados, foram realizadas outras, com todos os conjuntos frequentes. Os resultados provaram que a redução do número de conjuntos de termos frequentes não implicou uma redução do seu interesse. As técnicas usadas para eliminar os conjuntos de termos frequentes, a eliminação de conjuntos de termos frequentes que não contivessem um dos termos igual a um dos valores possíveis do atributo classe e a

aplicação do teste de qui-quadrado, já foram usados noutros de trabalhos de investigação, nomeadamente, por Brin *et*. *al* (1997).

O recurso a heurísticas de ordenação está relacionado com a necessidade de, aferir o grau de interesse de cada conjunto de termos frequente e de estabelecer uma relação de ordem entre os vários conjuntos. Testamos um conjunto alargado de heurísticas para seleccionarmos aquela que melhor se adapte a este método.

As técnicas usadas como filtros de selecção de conjuntos de termos frequentes são mais discutíveis. A primeira opção a que recorremos é lógica e não oferece grande contestação, já que se pretende seleccionar os conjuntos de termos frequentes que gerem atributos mais interessantes que os atributos originais. A segunda estratégia de filtro implica seleccionar no máximo um número de conjuntos de termos frequentes igual ao número de atributos originais. A lógica subjacente a este filtro, é a de que se espera, que o espaço necessário para especificar um problema recorrendo a novos atributos, que resultam da combinação de valores dos atributos originais, tenha no máximo o mesmo número de atributos que os originais. Com esta última estratégia poderemos estar a introduzir atributos irrelevantes no conjunto de dados. O filtro de selecção *3* é uma combinação dos dois primeiros e consiste em recorrer ao filtro *2* sempre com o filtro *1* não se conseguiam seleccionar conjuntos de termos frequentes. Os resultados obtidos recorrendo a cada um destes filtros podem ser observados nos Apêndices B a I.

Aplicabilidade do método proposto

A avaliação da aplicabilidade do método é de mais fácil entendimento nos conjuntos de dados artificiais, por sabermos à partida qual é o conceito objectivo subjacente a cada problema. Sendo assim, nos parágrafos seguintes, discutiremos a aplicabilidade do método para os 3 conjuntos de dados artificiais (*Monk1*, *Monk2* e *Monk3*).

O método proposto teve um bom desempenho no conjunto de dados *Monk1*. Sabemos que o conceito objectivo deste conjunto de dados é $(A_1 = A_2) \vee (A_5 = 1)$ e como

o método proposto, os novos atributos só podem ser representados por conjunções, basta encontrarmos relações do tipo: Se $A_1 = A_2$ então classe *1*; ou Se $A_5 = 1$ então classe *1*, para conseguirmos bons resultados. Nas experiências realizadas com o coeficiente de entropia, verificamos isto com clareza. Dos dez novos atributos construídos, dois foram usados na construção da árvore, *V12* e *V13*, ambos contendo o conceito objectivo.

$$V12 = \{<V1,"c">,<V2,"c">,<V7,"um">\}$$
$$V13 = \{<V1,"a">,<V2,"a">,<V7,"um">\}$$

A adequação do método ao conceito objectivo do problema em concreto, permitiu-nos obter um nível de exactidão máximo de exactidão e reduzir a complexidade da árvore.

O conceito objectivo do conjunto de dados *Monk2* é ter exactamente, dois atributos com o valor igual a *1* e os restantes com um valor diferente. Recorremos às experiências realizadas com o índice de *Kruskal-Goodman* para obter exemplos de conjuntos de termos frequentes seleccionados para a construção de árvores de decisão. Verificamos que os quatro atributos gerados a partir dos conjuntos de termos frequentes abaixo descritos estão de acordo com o conceito objectivo deste problema.

$$V8 = \{<V1,"a">,<V2,"a">,<V7,"zero">\}$$
$$V10 = \{<V1,"a">,<V4,"a">,<V7,"zero">\}$$
$$V12 = \{<V1,"a">,<V6,"a">,<V7,"zero">\}$$
$$V13 = \{<V2,"a">,<V3,"a">,<V7,"zero">\}$$

Estes conjuntos de termos frequentes são mais difíceis de encontrar, pois dois dos atributos têm de conter o primeiro valor e os restantes têm de ser diferentes.

No conjunto de dados *Monk3*, o conceito objectivo é $(A_5 = 3 \cap A_4 = 1) \cup (A_3 \neq 4 \cap A_2 \neq 3)$. No método que propomos apenas conseguimos extrair conceitos com base em atributos representados da forma $atr_i = valor$. Sendo assim, é possível extrair a primeira parte do conceito, mas não a segunda e daí os fracos resultados obtidos. Para além da representação de conceitos subjacente a este método não ser adequada para representar o conceito objectivo do problema em concreto,

também *5%* das observações contidas neste conjunto de dados têm as classificações trocadas, dificultando ainda mais as tarefas de classificação.

Morosidade do processo

Este método proposto contém operações morosas, pois cada conjunto de termos frequente tem de ser projectado sobre o conjunto de treino para ser avaliado, quer para as estratégias de filtro inicial, quer para a sua selecção. Este tipo de operações em conjuntos de dados com milhares de instâncias ou com dezenas de atributos pode implicar um tempo computacional muito elevado.

Um exemplo das possíveis alterações será a de implementar uma heurística dentro do algoritmo *Apriori* para reduzir o número de conjuntos de termos frequentes gerados. Esta morosidade do processamento poderá ser reduzida em trabalhos futuros

5 Conclusão

O objectivo que pretendemos alcançar com este estudo foi de encontrar um método que permita melhorar a performance dos modelos obtidos na aprendizagem de árvores de decisão, no que respeita à exactidão das previsões e à complexidade das teorias geradas. Na secção 5.1 descrevemos um resumo do método proposto e dos resultados obtidos. As principais contribuições desta tese são descritas na secção 5.2 e finalmente, na secção 5.3 são endereçados alguns pontos para trabalhos futuros.

5.1 Resumo do método proposto e dos resultados obtidos

O funcionamento do método que propusemos está esquematizado na figura 5.1. A partir de um conjunto de dados, são extraídos conjuntos de termos frequentes pelo recurso a um algoritmo de aprendizagem de regras de associação. Estes conjuntos de termos frequentes são depois filtrados, ordenados e seleccionados para serem usados na construção de novos atributos binários. Após esta fase, o conjunto de dados é estendido pelos novos atributos e é classificado pelo algoritmo de aprendizagem de árvores de decisão.

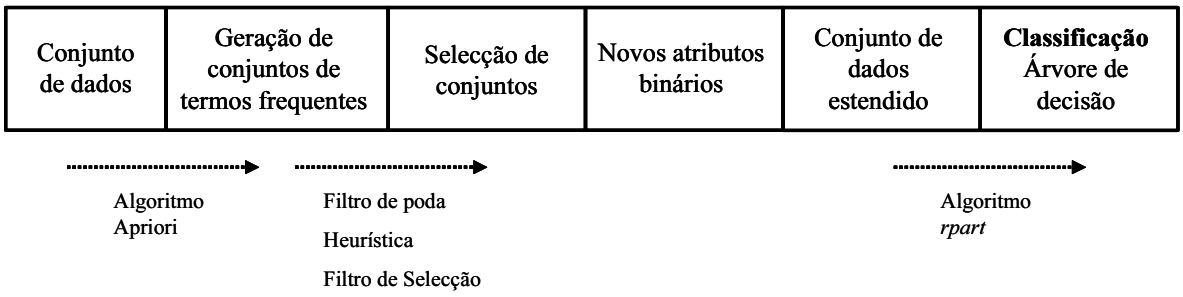


Figura 5.1: Arquitectura resumida do método proposto

Foram experimentadas oito heurísticas de ordenação de conjuntos de termos frequentes e em todas obtivemos melhores ou iguais resultados, relativamente aos obtidos pelos modelos gerados pelo algoritmo de aprendizagem de árvores de decisão

sobre os conjuntos de dados originais. Em cinco dessas heurísticas obtivemos ganhos em todos os conjuntos de dados à excepção do *Monk3*. Os maiores níveis de exactidão foram conseguidos com a heurística rácio de informação e de uma forma geral, foram as medidas baseadas na entropia, aquelas que proporcionaram os melhores resultados. Além destas últimas, as experiências realizadas com o coeficiente de *Gini* e com o índice de *Kruskal-Goodman* também proporcionaram bons resultados.

A conclusão que se pode retirar após a realização de todas experiências, descritas em pormenor no capítulo 4, é a de confirmar o interesse que a aprendizagem de regras de associação e mais concretamente, os conjuntos de termos frequentes poderão ter, na melhoria da performance dos modelos gerados por algoritmos de aprendizagem de árvores de decisão, em termos da sua capacidade de previsão e também na sua simplificação.

Depois das experiências e testes realizados, propomos a aplicação do método obedecendo aos seguintes passos:

(i) Extracção dos conjuntos de termos frequentes a partir do processamento do algoritmo *Apriori*;

(ii) Filtro de poda dos conjuntos de termos frequentes. A primeira estratégia será eliminar os conjuntos que não contenham como termo um dos valores possíveis para o atributo classe. Para os conjuntos não filtrados, devemos aplicar o teste de qui-quadrado para eliminar aqueles em que não se possa rejeitar a hipótese de independência entre a sua projecção sobre o conjunto de treino e o atributo classe. Este passo de pós-processamento irá reduzir substancialmente o número de conjuntos de termos frequentes disponíveis;

(iii) Selecção dos conjuntos de termos frequentes recorrendo ao rácio de informação como heurística de ordenação. Para cada conjunto de termos frequente é calculado o rácio de informação que lhe está associado e todos os conjuntos são ordenados por ordem decrescente de interesse. Depois será aplicado um filtro que consiste em seleccionar todos os

conjuntos cujo valor do rácio informação que lhes esteja associado seja superior ao valor máximo dessa heurística para os atributos originais. Para os casos em que o filtro anterior não retenha quaisquer conjuntos, então deveremos seleccionar tantos conjuntos quantos os atributos originais;

(iv) A fase final deste método consiste em estender o conjunto de dados original pelos novos atributos construídos e classificá-lo pela aplicação do algoritmo de aprendizagem de árvores de decisão.

O método que propomos contém alguns pontos que podem ser discutíveis. Como por exemplo, recorremos aos conjuntos de termos frequentes e não às regras de associação, pois apenas pretendemos saber como é os atributos se relacionam com o atributo classe e para isso, não necessitamos da informação contida nas regras de associação. Outro ponto discutível, e o facto de para calcular os testes de qui-quadrado e as heurísticas de ordenação, temos de recorrer a tabelas de contingência que resumam a informação entre a projecção do conjunto de termos frequente sobre o conjunto de treino e o atributo classe. Esta é uma operação morosa que pode condicionar a aplicação deste método a conjuntos de dados muito extensos. Como referimos na secção 4.6, estas operações podem ser reduzidas, se em trabalhos futuros optarmos pelo desenvolvimento de algoritmos de aprendizagem que incorporem essas estatísticas.

5.2 Contribuições deste estudo

Neste estudo é apresentada uma nova abordagem de combinação de modelos, de regras de associação e de árvores de decisão. As principais contribuições desta tese são as seguintes:

- Metodologia para construção de árvores multivariadas para conjuntos de dados com atributos nominais, que é uma área relativamente pouco explorada. Esta é uma metodologia inovadora, pois recorremos aos conjuntos de termos frequentes obtidos para construir os novos atributos. Estes novos atributos

representam uma extensão da linguagem de representação das regras de associação;

- Construção de modelos com maiores níveis de exactidão. A avaliação experimental do método proposto mostrou-nos que para os conjuntos de dados em análise, este método conseguiu melhorar os níveis de exactidão dos modelos;

- Maior interpretabilidade dos modelos gerados, por estes serem mais simples. Como os novos atributos construídos representam combinações de termos dos restantes atributos, a sua selecção para integrar a construção das árvores de decisão permite obter árvores de decisão mais simples;

- Redução da fragmentação de conceitos nas árvores de decisão.

5.3 Trabalho Futuro

No tempo disponível para a preparação da tese, não foi possível abarcar todas áreas que gostaríamos. Sendo assim, enumeramos alguns pontos que constituirão uma base para trabalhos futuros:

- Melhorar os filtros de selecção dos conjuntos de termos frequentes para que não sejam excluídos conjuntos de termos interessantes;

- Comparar o método proposto que recorre aos conjuntos de termos frequentes com o outro método recorrendo directamente às regras de associação para construir novos atributos;

- Adaptar o método proposto para que se possamos usar conjuntos de dados com atributos contínuos. É necessário construir um passo para discretizar estes atributos;

- Construir um algoritmo de aprendizagem baseado no algoritmo *Apriori* que tenha incorporado uma heurística para gerar apenas os conjuntos mais interessantes;

- Outra possibilidade será estender o conjunto de dados ao longo da construção da árvore de decisão. Para todos os subconjuntos de dados resultantes de um nó de decisão seria necessário extrair conjuntos de termos frequentes e estender os subconjuntos de dados com novos atributos binários;

- Construir um algoritmo de aprendizagem que incorpore todos os passos deste método.

Este trabalho abre novos caminhos de investigação numa área ainda pouco explorada: a combinação de regras de associação com árvores de decisão. O recurso a este tipo de combinações faz com que se abram fronteiras a uma nova classe de árvores de decisão que incorporem testes multi-variados de atributos nominais.

Apêndice A – Conjuntos de dados

A.1 *TicTacToe*

Neste conjunto de dados contém um conjunto de configurações finais do jogo *TicTacTtoe*, e se assume que "*x*" jogou primeiro. O conceito objectivo é "ganhar x" (é verdade quando "x" é uma das 8 possíveis formas de criar um "três em linha".

É constituída por 958 instâncias e 9 atributos, cada um destes, corresponde a um quadrado do tabuleiro *TicTac*. Os atributos são todos nominais e contêm os valores, "x" que representa as jogadas do jogador *x*, "o" as jogadas do jogador o e "b" significa um quadrado em branco. Os atributos representam a seguinte informação:

1. Quadrado esquerdo da linha de cima: *{x, o, b}*
2. Quadrado do meio da linha de cima: *{x, o, b}*
3. Quadrado direito da linha de cima: *{x, o, b}*
4. Quadrado esquerdo da linha intermédia: *{x, o, b}*
5. Quadrado do meio da linha intermédia: *{x, o, b}*
6. Quadrado direito da linha intermédia: *{x, o, b}*
7. Quadrado esquerda da linha de baixo: *{x, o, b}*
8. Quadrado do meio da linha de baixo: *{x, o, b}*
9. Quadrado direito da linha de cima: *{x, o, b}*
10. Classe: {positivo, negativo}

Neste conjunto de dados não existem valores omissos e *65%* das instâncias são positivas ("ganhar *x*").

A.2 *Monks*

Estes conjuntos de dados são artificiais e foram originalmente desenvolvidos para comparar as performances entre algoritmos. Os robots são caracterizados por seis atributos:

1 Forma da cabeça (*A1*) = {"redonda", "quadrada", "octogonal"} = *{1, 2, 3}*

2 Forma do corpo (*A2*) = {“redonda”, “quadrada”, “octogonal”} = *{1, 2, 3}*

3 Sorridente (*A3*) = {“sim”, “não”} = *{1, 2}*

4 Com (*A4*) = {“espada “, “balão”, “bandeira”} = {1, 2, 3}

5 Cor do casaco (*A5*) = {“vermelho”, “amarelo”, “verde”, “azul”} =*{1, 2, 3, 4}*

6 Gravata? (*A6*) = {“sim”, “não”} = *{1, 2}*

Os conceitos objectivos para os três conjuntos de dados são os seguintes:

- MONK-1: $(A_1 = A_2) \vee (A_5 = 1)$;
- MONK-2: ter exactamente dois valores do conjunto

$$\{a_1 = 1, a_2 = 1, a_3 = 1, a_4 = 1, a_5 = 1, a_6 = 1\};$$

- MONK-3: $(A_5 = 3 \wedge A_4 = 1) \vee (A_5 \neq 4 \wedge A_2 \neq 3)$.

Os conjuntos, *Monk1* e *Monk3*, contêm atributos irrelevantes. Existe ruído no conjunto de dados *Monk3*: *5%* das classes estão trocadas. Cada conjunto de dados contém 432 instâncias e não existem valores omissos.

A.3 *Promoters*

Este conjunto de dados foi desenvolvido para ajudar a avaliar um sistema de aprendizagem híbrido (“KBANN”) que usa exemplos para indutivamente refinar conhecimento pré-existente. Contém *106* instâncias e *59* atributos. As instâncias estão igualmente distribuídas pelas classes. A classe (+) representa um *promoter* e a classe (-) não *promoter*.

Apêndice B – Resultados obtidos com o coeficiente de entropia

B.1 Resultados

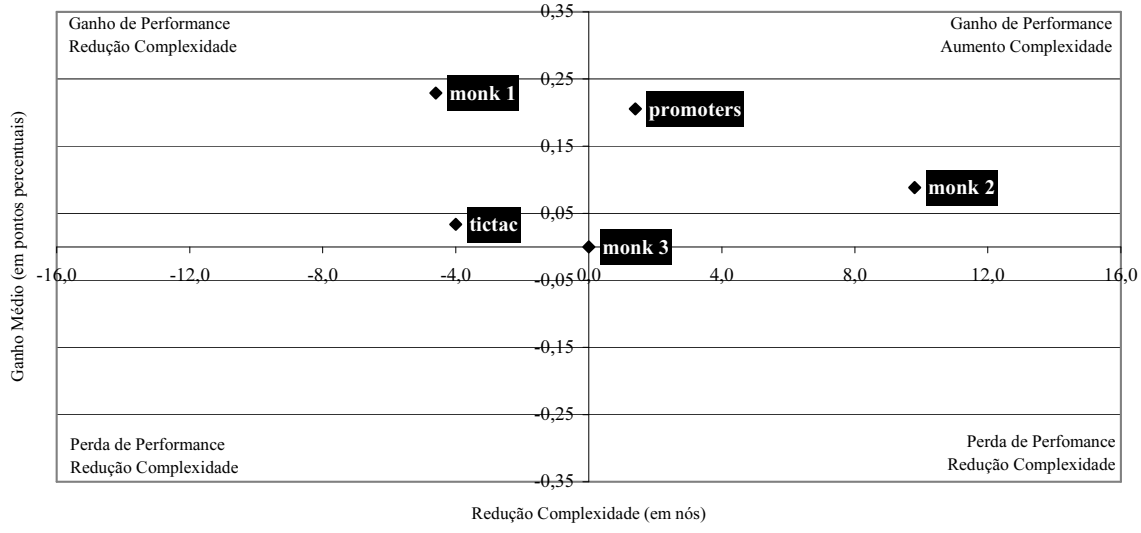
FILTRO DE SELECÇÃO 1

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	0,5	0,09	0,02	0,06	0,03	0,027	0,04	5	5	0	43	40
Promoters	1,3	0,31	0,11	0,13	0,11	0,178	0,14	8	2	0	6	7
Monk1	0,0	0,23	0,11	0,23	0,11	0,000	0,00	0	10	0	16	16
Monk2	26,6	0,28	0,08	0,19	0,06	0,088	0,07	8	2	0	31	41
Monk3	0,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

FILTRO DE SELECÇÃO 2

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,05	0,04	0,036	0,05	8	0	2	43	40
Promoters	25,0	0,31	0,11	0,11	0,13	0,196	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,25	0,07	0,021	0,10	8	2	0	31	40
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

B.2 Gráfico de quadrantes



Apêndice C – Resultados obtidos com o do ganho de informação

C.1 Resultados

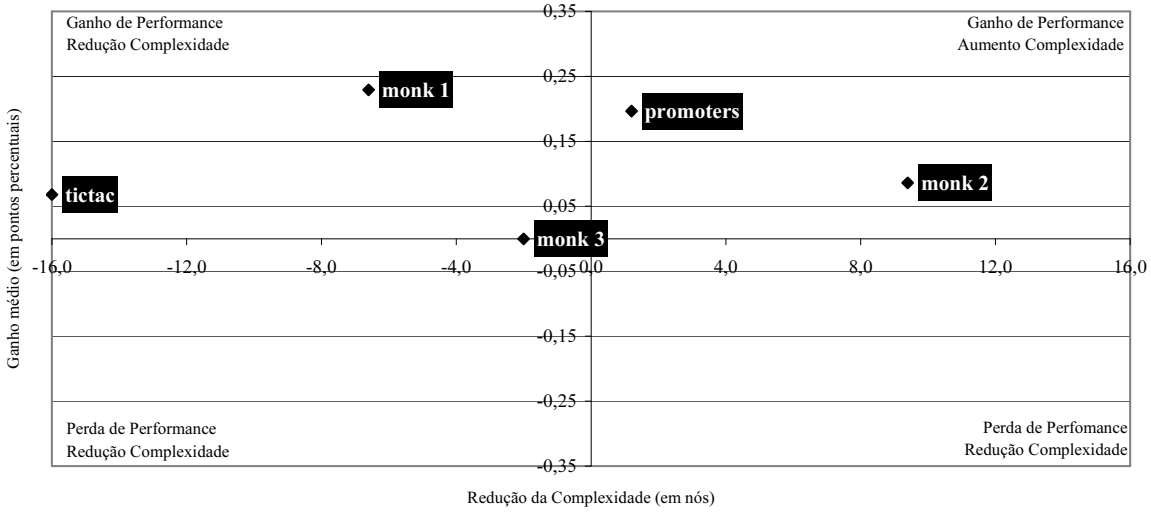
FILTRO DE SELECÇÃO 1

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	251,4	0,09	0,02	0,02	0,02	0,068	0,04	10	0	0	43	27
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	56,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	27,4	0,28	0,08	0,19	0,06	0,086	0,07	8	2	0	31	41
Monk3	100,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	7

FILTRO DE SELECÇÃO 2

Conjuntos de Dados (1)	Atrib. (2)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (3)	d.p. (4)	Méd (5)	d.p. (6)	Méd (7)	d.p. (8)	G> 0 (9)	G= 0 (10)	G< 0 (11)	Arv 0 (12)	Arv 1 (13)
Tic Tac Toe	9,0	0,09	0,02	0,05	0,04	0,036	0,05	8	0	2	43	40
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	0	11
Monk2	6,0	0,28	0,08	0,26	0,07	0,021	0,10	5	1	4	31	40
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

C.2 Gráfico de quadrantes



Apêndice D – Resultados obtidos com o rácio de informação

D.1 Resultados

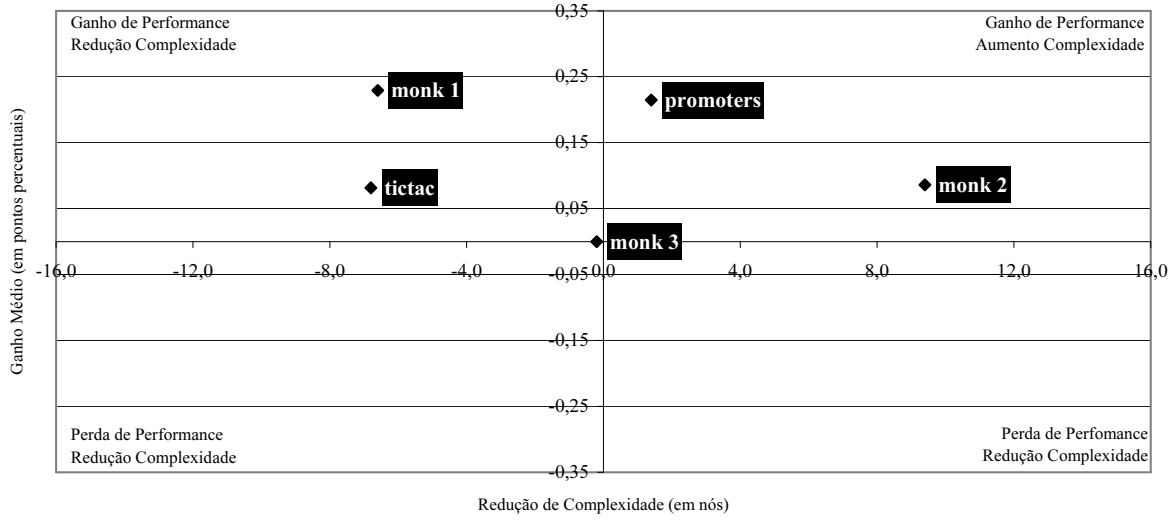
FILTRO DE SELECÇÃO 1

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	26,3	0,09	0,02	0,01	0,01	0,082	0,03	10	0	0	43	37
Promoters	18,1	0,31	0,11	0,10	0,11	0,215	0,13	9	1	0	6	7
Monk1	31,6	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	27,4	0,28	0,08	0,19	0,06	0,086	0,07	8	2	0	31	41
Monk3	30,8	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

FILTRO DE SELECÇÃO 2

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,06	0,04	0,028	0,05	7	1	2	43	40
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,26	0,07	0,021	0,10	5	1	4	31	40
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

D.2 Gráfico de quadrantes



Apêndice E – Resultados obtidos com o coeficiente de *Gini*

E.1 Resultados

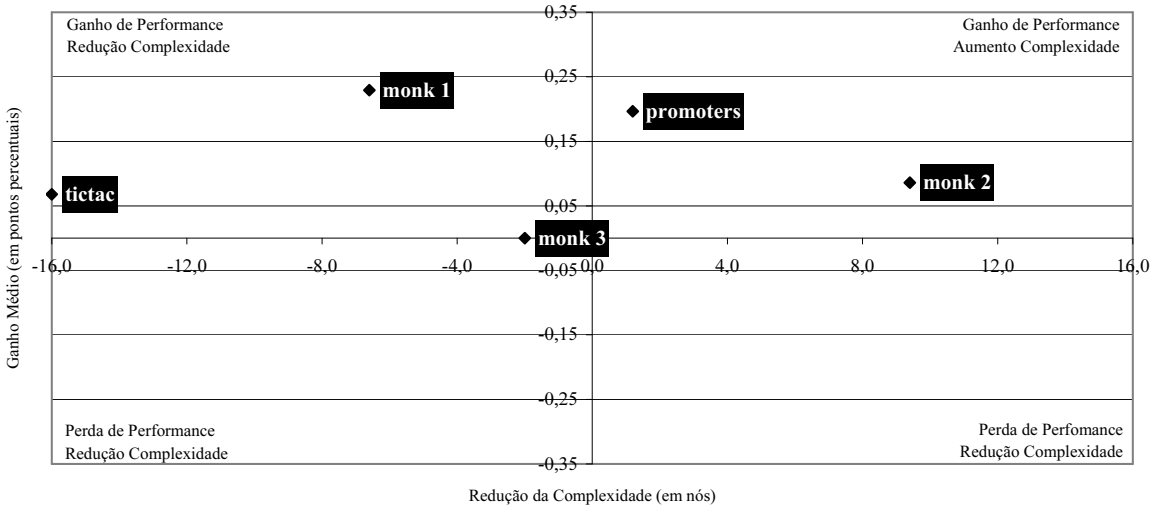
FILTRO DE SELECÇÃO 1

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	251,4	0,09	0,02	0,02	0,02	0,068	0,04	10	0	0	43	27
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	6
Monk1	56,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	9
Monk2	27,4	0,28	0,08	0,19	0,06	0,086	0,07	8	2	0	31	41
Monk3	100,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	7

FILTRO DE SELECÇÃO 2

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões									Complexidade das Árvores	
		Erro Original		Erro Estendido		Ganho		Tipo de Resultados				
		Méd (2)	d.p. (3)	Méd (4)	d.p. (5)	Méd (6)	d.p. (7)	G> 0 (8)	G= 0 (9)	G< 0 (10)	Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,07	0,03	0,018	0,05	7	3	0	43	39
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,26	0,06	0,021	0,10	5	1	4	31	41
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

E.2 Gráfico de quadrantes



Apêndice F – Resultados obtidos com o índice de associação *kruskal-Goodman*

F.1 Resultados

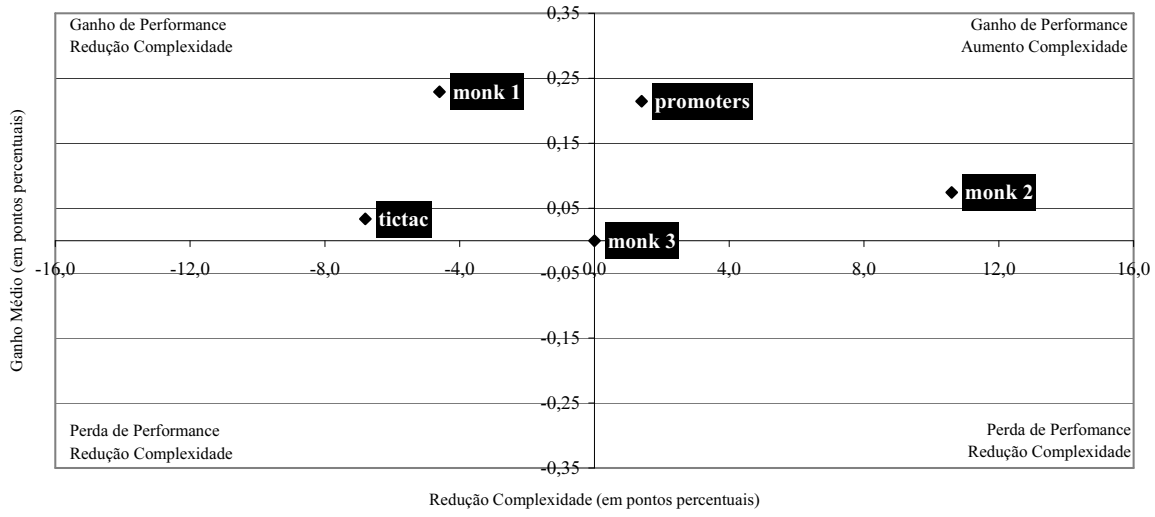
FILTRO SELECÇÃO 1

Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões: Erro Original Méd (2)	Erro Original d.p. (3)	Erro Estendido Méd (4)	Erro Estendido d.p. (5)	Ganho Méd (6)	Ganho d.p. (7)	Tipo de Resultados G> 0 (8)	G= 0 (9)	G< 0 (10)	Complexidade das Árvores Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	2,0	0,09	0,02	0,06	0,03	0,034	0,04	7	2	1	43	37
Promoters	0,4	0,31	0,11	0,24	0,18	0,066	0,10	4	6	0	6	6
Monk1	0,0	0,23	0,11	0,23	0,11	0,000	0,00	0	10	0	16	16
Monk2	0,0	0,28	0,08	0,28	0,08	0,000	0,00	0	10	0	31	31
Monk3	0,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

FILTRO SELECÇÃO 2

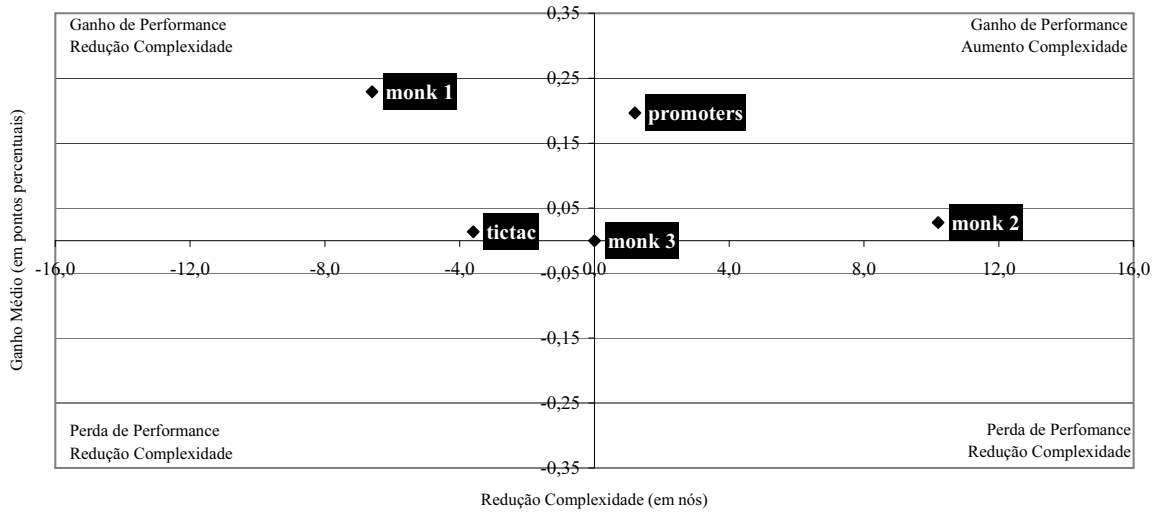
Conjuntos de Dados	Atrib. (1)	Exactidão das Previsões: Erro Original Méd (2)	Erro Original d.p. (3)	Erro Estendido Méd (4)	Erro Estendido d.p. (5)	Ganho Méd (6)	Ganho d.p. (7)	Tipo de Resultados G> 0 (8)	G= 0 (9)	G< 0 (10)	Complexidade das Árvores Arv 0 (11)	Arv 1 (12)
Tic Tac Toe	9,0	0,09	0,02	0,08	0,04	0,014	0,05	7	0	3	43	40
Promoters	25,0	0,31	0,11	0,11	0,13	0,197	0,14	8	2	0	6	7
Monk1	6,0	0,23	0,11	0,00	0,00	0,229	0,11	10	0	0	16	11
Monk2	6,0	0,28	0,08	0,20	0,06	0,074	0,08	9	1	0	31	42
Monk3	6,0	0,01	0,02	0,01	0,02	0,000	0,00	0	10	0	9	9

F.2 Gráfico de quadrantes



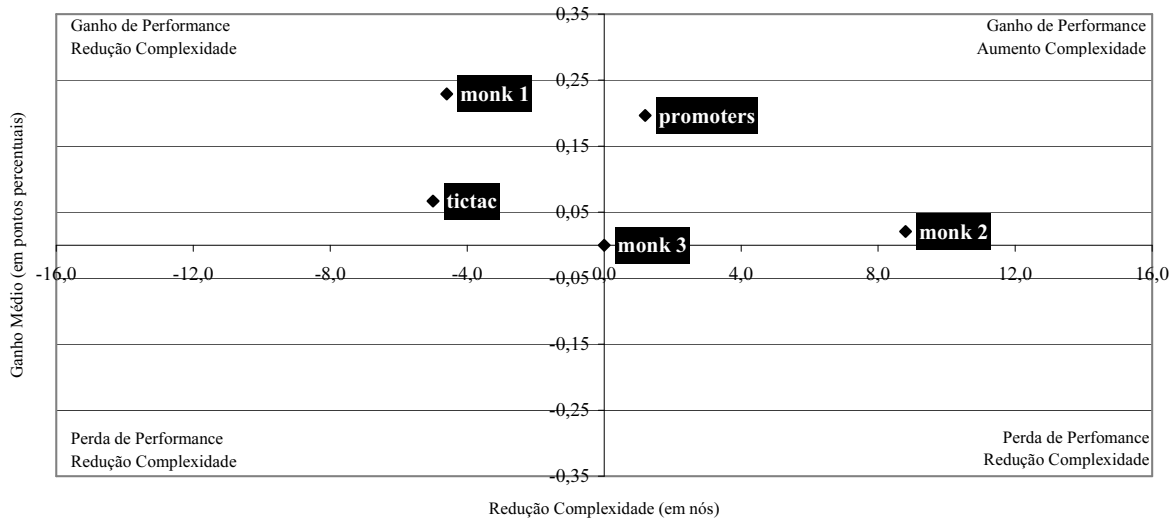
Apêndice G – Resultados obtidos com coeficiente de interesse

G.1 Gráfico de quadrantes



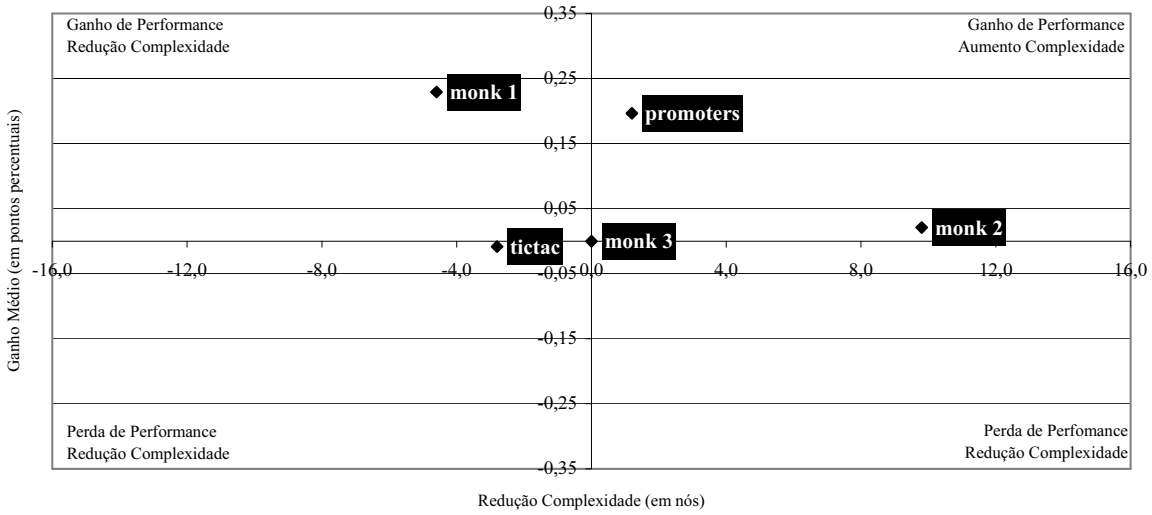
Apêndice H – Resultados obtidos com coeficiente de *Laplace*

H.1 Gráfico de quadrantes



Apêndice I – Resultados obtidos com o coeficiente de correlação

I.1 Gráfico de quadrantes



Bibliografia

Agrawal, R.; T. Imielinski; e A. Swami (1993). “Mining association rules between sets of items in large databases”. In Proc. of the ACM SIGMOD Conference on Managment of Data, Washington, USA

Agrawal, R. e R. Strikant (1994). “Fast Algorithms for Mining Association Rules”. In Proc. of the Inteligence Conference on Very Large Databases, Santiago, Chile

Bayardo, R.J. e R. Agrawal (1999). “Mining the most Interesting Rules”. In Proc of ACM SIGKDD Conference on Knowledge Discovery and Data Mining, San Diego, California, USA

Berry, Michael J. A. e Gordon Linoff (1997). *Data Mining Techniques - For Marketing Sales and Customer Support*. John Willey & Sons, New York, USA

Berzal, Fernando Galiano (2002). *ART - Un método alternativo para la construcción de arboles de decisión*, Tesis Doctoral, Deparatemento de Ciência de Computación e Inteligencia Artifical, Universidad de Granada, Espanha

Brin, S.; R. Montwani; S. D. Ullman; e S. Tsur (1997). “Dynamic itemset counting and implications rules for market basket data”. In. Proc. of ACM SIGMOD Intelligence Conference on Managment of Data, Tucson, Arizona, USA

Blake, C.; E. Keogh; e C.J. Merz (1999). UCI Repository of machine learining databases [http://www.ucs.uci.edu/~mlearn/MLRepository.html]. Department of Computer Science, University of California, Irvine, USA

Breiman, L.; J. Friedman; R. Olshen; e C. Stone (1984). *Classification and Regression Trees*. Wadsworth International Group, California, USA

Breiman, L (1996). “Bagging Preditors”. *Machine Learning*, 24 (2),123-140

Brin, S.; R. Motwani; e C. Silverstein (1997). "*Beyond Market Baskets: Generalizing Association Rules to Correlations*". In Proc. of ACM SIGMOD International Conference on Management of Data, Tucson, Arizona, USA

Brodley, C. E. e P. E. Utgoff (1995). "Multivariate Decision Trees". *Machine Learning*, 19 (1), 45-77

Brodley, C. E. e P. E. Utgoff (1992). "*Multivariate Decision Trees versus Univariate Decision Trees*". COINS Technical report, 92 (8), University of Massachusetts, Amberst, USA

Clark, P. e R. Bowell (1991). "Rule Induction with CN2: Some Recent Improvments". In Proc. Eurpean Work Session on Learning (EWL91), Berlin, Germany

Domingos, Pedro (1997). *A Unified approach to concept Learning*, Ph.D. Thesis, University of California, Irvine, Caifornia, USA

Everitt, B.S (1977). *The analysis of contingency tables*. Chapman and Hall, New York, USA

Fayyad, U.M.; G. Piatesky-Shapiro; P. Smyth (1996). "From data mining to knowledge discovery: An overview". In U.M. Fayyad, G. Piatesky-Shapiro, P. Smyth & R. Uthurusamy (eds), *Adavances in Knowledge Discovery and Data Mining,* AAAI Press., Californina, 1-34

Gama, João (1999). *Combining classification algorithms*, Tese de Ph.D. Universidade do Porto, Faculdade de Ciências, Porto, Portugal

Hilderman, R. J.; H. J. Hamilton; e B. Barber. (1999). "Ranking the interstingness of summaries from data mining systems". In Proc. of International Florida Artificial Intelligence Research Symposium (FLAIRS´99), Orlando, USA

Hipp, J.; U. Gutnzer; e G. Nakhaeizadeh (2000) "Algorithms for Association Rule Mining – A General Survey and Comparison", In Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, Massachusetts, USA

Jovanoski, V. e Nada Lavrac (2001). “Classification rule learning with apriori-c”, In Proc. of 10th Conference on Articial Intelligence (EPIA2001), Porto, Portugal

Liu, Bing; W. Hsu; e Y. Ma (1998). “Integrating classification and association rule mining”. In Proc. of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 99), New York, USA

Liu, Bing; W. Hsu; e Y. Ma (1999). “Pruning and sumarizing the discovered associations”. In Proc. of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 99), San Diego, CA, USA

Michalski, R.S. (1978). “Pattern recognition as knowledge- guided computer inductuion”. *Technical Reports 927*, Department of Computer Science, The University of Illinois at Urbana- Campaign, Urbana, Illionis, USA

Michalski, R.S. (1983). “A theory and methodology of inductive learning”, *Artifial Intelligence*, 20, 111-161

Ming, Ting Kai e H. Witten (1999). “Issues in stacked generalization”. *Journal of Artificial Inteligence Research*, 10, 271- 289

Mitchell, Tom (1997). *Machine Learning*. Mac Graw Hill Companies, Inc., New York, USA

Piatetsky- Shapiro, Gregory (1991). “Discovery, analysis and presentation of strong rules”. In Gregory Piatetsky- Shapiro and William Frawley, editors, *Knowledge Discovery in Databases*. MIT Press, Cambridge, Massachusetts, USA

Quinlan, J.R. (1987). “Simplifying decision trees”. *International Journal of Man-Machine Studies*, 27, 221- 234.

Quinlan, J.R. (1993).*C4.5: Programs for machine learning*. Morgan Kaufmann Publishers, San Mateo, California, USA

Rao, R. B.; D. Gordon; e W. Spears.(1995). “For every genralization action, is there really an equal and opposite reaction? Analysis of the conservation law for

generalization performance". In Proc. of International Conference on Machine Learning, Nashville, Tennesse, USA

Savasere A.; R. Omiecinski; e S. Navathe (1995). "An efficient algorithm for mining association rules in large databases". In Proc. of 21th intelligence conference on very large databases (VLDB´95), Zurich, Switzerland

Sarsfield Cabral e Rui Guimarães (1997). *Estatística*. McGraw Hill Portugal

Schaffer, Cullen (1993). "Selecting a classification method by cross validation". *Machine Learning*, 13(1), 135- 143

Schaffer, Cullen (1994). "A conservation law for generalization performance". In Proc.of International Conference on Machine Learning, San Francisco, California, USA

Schapire, Robert (1990). "The strength of weak learnability". *Machine Learning*, 5, 197-227

Sreerama, K. M. (1998). "Automatic construction of decision trees from data: a multi-disciplinary survey". *Data Mining and Knowledge Discovery*, 2(4), 345--389

Tan, Pang-Nin e Vipin Kumar (2000). "Interestingness measures for association patterns: a perspective". Workshop on Post Processing in Machine learning and Data Mining (KDD2000), Boston, Massachusetts, USA

Terabe, M.; T. Washio; H. Motoda; O. Katai; e T. Sawaragi (2002). "Attribute generation based on association rules", *Knowledge and Information Systems*, 4(3), 329-349

Toivonen H. (1996), "Sampling large databases for association rules". In Proc. of International Conference of Very Large Data Bases, Bombay, India

Zaki, Mahommed Javeed (1998). *Scalable data Mining for rules*, Ph.D. Thesis. University of Rochester, New York, USA

Zheng, Z.(1996). *Construction new attributes for decision tree Learning*. PhD Thesis. University of Sydney, Australia

Witten, Ian e Eibe Frank (1999). *Data mining: practical machine learning tools with Java implementations.* Morgan Kaufmann Publishers, San Francisco, USA

Wolpert, D (1992). "Stacked generalization". *Neural Networks*, 5