UNIVERSIDADE ABERTA

Recursos Educacionais / Educational Resources

Seleção de Atributos utilizando a Análise Lógica de Dados Inconsistentes (LAID)

João Apolónia
Luís Cavique

Lisboa 2019

Resumo

O tratamento de conjuntos de dados de grande dimensão é uma questão que é recorrente nos dias de hoje e cuja tarefa não é simples, dadas as limitações computacionais, ainda, existentes. Uma das abordagens possíveis passa por realizar uma seleção de atributos que permita diminuir, consideravelmente, a dimensão dos dados sem aumentar a inconsistência dos mesmos. "*Rough Sets*" é uma abordagem que difere doutras técnicas de seleção de atributos pela sua capacidade de lidar com dados inconsistentes. Outra abordagem para redução de dados é conhecida como Análise Lógica de Dados (LAD). A Análise Lógica de Dados Inconsistentes (LAID) junta as vantagens destas duas abordagens.

Palavras chave: *Data Mining*; Seleção de atributos; LAID; Inconsistência dos dados.

Índice

1. Introdução

Data Mining consiste num conjunto de processos e métodos destinados ao tratamento e análise de dados, procurando modelos, padrões, relacionamentos entre variáveis e validá-los, aplicando esses modelos a novos conjuntos de dados.

A existência de um grande volume de observações (registos), as quais pertencem a uma de várias classes (resultados possíveis) e que poderão ter associados muitos atributos ou caraterísticas, faz com que a dimensão do conjunto de dados seja tal que, mesmo computacionalmente, torne o seu tratamento extremamente moroso e pesado. É neste contexto que surgem os métodos de seleção de atributos (*feature selection*). O intuito é escolher, entre os atributos existentes, um conjunto mínimo que permita obter resultados o mais próximo possível dos resultados do conjunto inicial, removendo atributos que sejam irrelevantes e que poderiam originar modelos de menor qualidade. Estes processos de seleção aceleram os algoritmos de *Data Mining*, melhoram a precisão das previsões e facilitam a sua compreensão (KUMAR & MINZ, 2014).

Quando o número de observações é muito menor que o de atributos, aumenta a dificuldade de todo o processo de extração de conhecimento, porque o espaço de pesquisa, normalmente, é escassamente povoado (matriz esparsa) (KUMAR & MINZ, 2014). Outros problemas prendem-se com a possibilidade de existir uma grande quantidade de atributos irrelevantes, redundantes, ou dependentes, e dados descontextualizados (*noisy data*).

Existem várias técnicas de seleção de atributos, entre as quais estão os *Rough Sets* e a Análise Lógica de Dados (LAD), cada uma com as suas virtudes e defeitos. Da combinação destas duas técnicas, surge a Análise Lógica de Dados Inconsistentes (LAID), que vai buscar o melhor de cada uma das técnicas primárias (CAVIQUE *et al.*, 2013).

Dada a escassez de estudos e aplicação de técnicas de seleção de atributos a conjuntos de dados de grande dimensão (KUMAR & MINZ, 2014), o projeto de dissertação propõe-se colmatar esta lacuna e disponibilizar a implementação de um algoritmo de seleção, com avaliação dos resultados obtidos.

2. Seleção de atributos

À medida que se dissemina o uso das TI no mundo empresarial, nas instituições oficiais e no dia a dia das pessoas, aumenta, de forma exponencial, a disponibilização de informação em todas as áreas do conhecimento humano. Apesar da grande evolução registada neste século, ao nível dos equipamentos informáticos, não tem sido suficiente para acompanhar as exigências de processamento dessa mesma informação digital. Para lidar com tamanho volume de dados é necessário a sua filtragem, através de algoritmos de pré-processamento.

A seleção de atributos é uma das principais técnicas de pré-tratamento de grandes volumes de dados. Segundo Kumar e Minz (2014), a seleção de atributos é um processo que permite detetar os atributos relevantes num conjunto de dados e eliminar os irrelevantes, redundantes ou causadores de ruído (onde o valor original foi modificado, por exemplo, no caso de entrevistas, a existência de falsos testemunhos). Esta seleção acelera os algoritmos de *Data Mining*, aumenta a precisão na realização de previsões e melhora a compreensão dos processos. O problema agrava-se quando, no conjunto dos dados, o número de atributos é muito maior que as observações realizadas, originando um espaço de pesquisa mais esparso e dificultando a diferenciação do que é relevante por contraponto com o que é ruído (KUMAR, MINZ, 2014). Uma boa seleção de atributos permite facilitar o processo de aprendizagem e compreensão do domínio em estudo, obter melhores previsões, simplificar modelos e reduzir custos.

Nos últimos anos, a seleção de atributos tem sido objeto de muitos estudos e artigos publicados. Tem sido aplicada em muitas áreas da atividade humana como na análise do ADN, deteção de perturbações, classificação de textos, ou músicas, tratamento de imagens, etc. (BOLÓN-CANEDO *et al.*, 2013). Por consequência, surgiram muitos algoritmos de seleção, sendo difícil estabelecer um critério de avaliação dos mesmos. Entre os autores, é unânime concluir que não existe um método que possa ser considerado "o melhor", dependendo de muitas variáveis do domínio em estudo.

2.1. Etapas do processo de seleção de atributos

A seleção de atributos pode ser realizada segundo duas filosofias distintas (GUYON, ELISSEEFF, 2003):

- Avaliação individual dos atributos, onde a cada atributo é atribuído um peso de acordo com a sua relevância;
- Avaliação de um conjunto de atributos selecionados obtidos iterativamente segundo um critério definido.

A avaliação individual não consegue remover atributos redundantes, porque tenderão a ter pesos semelhantes, ao contrário da avaliação de subconjuntos de atributos, que pode lidar com redundâncias, mas tem problemas inerentes à procura dos subconjuntos (BOLÓN-CANEDO *et al.*, 2013).

Na opinião de Dash e Liu (1997), o procedimento típico de seleção de atributos é composto por quatro etapas (ver Figura 2.1):

1. Procedimento heurístico de procura para selecionar um subconjunto de atributos, muito influenciado pelo método de escolha dos atributos e pelo ponto de partida;
2. Avaliação do subconjunto de atributos gerado, por comparação com o subconjunto previamente obtido, que poderá ser independente de algoritmos de mineração ou ser dependente destes e recorrer a dados de treino;
3. O processo é repetido até que se cumpra determinado critério de paragem definido, que poderá depender do número de iterações, do número de atributos escolhidos, se a alteração de algum atributo já não acrescenta valor, ou depender de alguma função de decisão (que poderá substituir a etapa de avaliação);
4. Validação do subconjunto obtido, através de vários testes, por comparação com resultados previamente obtidos ou comparando com os resultados de métodos alternativos.

A seleção de um subconjunto de atributos não é tarefa fácil, a começar pelo facto de, numa base de dados com n atributos, existirem 2^n-1 subconjuntos diferentes não vazios (${}^nC_1+{}^nC_2+\ldots+{}^nC_n$). Pelo facto de o problema de seleção ter um crescimento exponencial com o

número de atributos, obriga ao uso de estratégias, que poderão ser exaustivas, sequenciais ou aleatórias (DOAK, 1992).

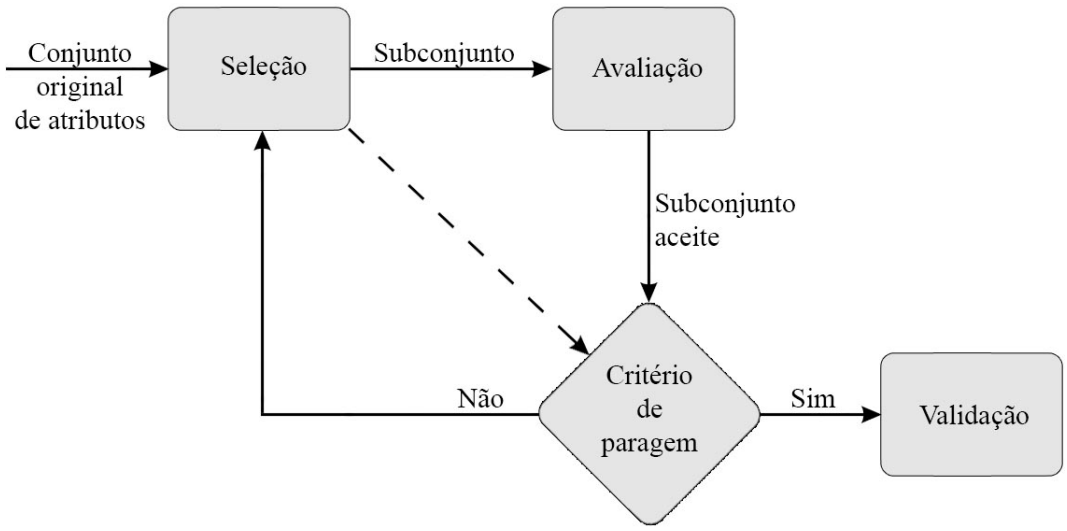

Figura 2.1 – Processo de seleção de atributos com validação.
(Adaptado de: DASH, LIU, 1997)

A pesquisa exaustiva garante a solução ótima. O algoritmo *Exhaustive Search* examina todas as possibilidades no espaço de pesquisa, mas existem outros que, efetuam a pesquisa exaustiva numa determinada vizinhança, ou dentro de certos limites, como são o caso dos algoritmos *Branch and Bound*, *Approximate Monotonicity with Branch and Bound* e *Beam Search* (DOAK, 1992).

Na pesquisa sequencial, são selecionados um ou mais atributos, num processo iterativo, onde o número máximo de iterações é $O(n)$. A pesquisa sequencial é completa, fácil de implementar, mas pode não obter o subconjunto ótimo e o espaço de pesquisa é $O(n^2)$ (KUMAR, MINZ, 2014). Pode ter muitas variantes: a *Forward Sequential Selection* começa com o subconjunto vazio e adiciona um atributo de cada vez; a *Backward Sequential Selection* começa com o conjunto total de atributos e remove um atributo de cada vez; a *PQ Sequential Selection* adiciona atributos numa iteração e remove na seguinte; a *Bi-Directional Search* efetua duas pesquisas em simultâneo, por iteração, uma para adicionar um atributo, outra para retirar (DOAK, 1992).

A pesquisa aleatória parte da um subconjunto com atributos escolhidos aleatoriamente. Durante as várias iterações, a seleção pode ser completamente aleatória, ou sequencial, mas incluindo alguma aleatoriedade na escolha (KUMAR, MINZ, 2014). A aleatoriedade tem como objetivo conseguir-se escapar a mínimos locais do espaço de pesquisa (DOAK, 1992). A ordem do espaço de pesquisa é $O(n^2)$ (KUMAR, MINZ, 2014)

No processo iterativo, após obter um subconjunto, há que avaliá-lo, para verificar se é melhor que o melhor subconjunto até então encontrado. Aqui, é essencial definir um critério de avaliação. Há que ter em conta que, um subconjunto pode preencher os requisitos de um critério e não os de outro (KUMAR, MINZ, 2014). Esta não é uma questão fácil, nomeadamente, se estivermos a lidar com bases de dados de grande dimensão, onde o número de observações é muito menor que o número de atributos (KOHAVI, JOHN, 1997), porque, neste caso, temos, relativamente, menos observações para poder avaliar a seleção obtida.

Existem, dois tipos de critérios, baseados na sua dependência, ou não, dos algoritmos de aprendizagem (KUMAR, MINZ, 2014).

Os critérios independentes não têm em conta o algoritmo de aprendizagem, recorrendo a modelos de filtragem e aos dados de treino para avaliar os subconjuntos. Medidas de distância, medidas de informação ou incerteza, medidas de probabilidade de erro, medidas de dependência, medidas de distância interclasse e medidas de consistência são alguns, dos muitos, critérios independentes existentes (KUMAR, MINZ, 2014).

Na aplicação de critérios dependentes é necessário ter o algoritmo de aprendizagem já determinado. A avaliação dos subconjuntos de atributos é feita recorrendo aos processos do algoritmo, o que aumenta a performance deste. No entanto, é computacionalmente dispendioso, porque é efetuada uma estimativa da precisão para cada subconjunto de atributos (KUMAR, MINZ, 2014).

Por fim, o critério de paragem pode corresponder a um, ou a uma combinação, dos seguintes casos (KUMAR, MINZ, 2014):

- Limitar o número de iterações;
- Definir um número mínimo de atributos selecionados;
- Definir uma taxa mínima de erro de classificação;

- Completar a pesquisa;
- Verificar se retirar, ou acrescentar, novos atributos ao subconjunto já não origina diferenças significativas.

2.2. Diferentes abordagens à seleção de atributos

Bólon-Canedo, Sánchez-Maroño e Alonso-Betanzos (2013), Kumar e Minz (2014) indicam a existência de três diferentes abordagens à seleção de atributos: *Filter*; *Wrapper*; *Embedded*.

2.2.1. Abordagem *Filter*

Os métodos que seguem esta abordagem fazem a seleção dos atributos com base na sua importância para a previsão da variável em estudo (classe), independentemente do algoritmo de aprendizagem. Estes métodos selecionam os atributos mais relevantes para determinar o valor da classe. São rápidos, eficientes e permitem boas previsões (BOLÓN-CANEDO *et al.*, 2013). O problema desta abordagem é poder selecionar atributos redundantes e não selecionar atributos que, não tendo importância por si, são importantes quando combinados com outros (KUMAR, MINZ, 2014). São frequentemente usados para pré-processamento dos dados (BOLÓN-CANEDO *et al.*, 2013). *FOCUS, ABB, Relief* (KUMAR, MINZ, 2014), *CFS, INTERACT, Consistency-based Filter, InfoGain, ReliefF, mRMR* e M_d (BOLÓN-CANEDO *et al.*, 2013) são exemplos de métodos que usam esta abordagem.

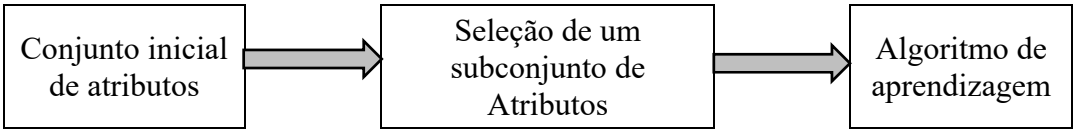

Figura 2.2 – Abordagem Filter.
(Fonte: JOHN et al., 1994)

A abordagem *Filter* tem uma boa performance, porque a filtragem redunda numa única solução, sendo mais intuitiva. No entanto, porque não tem em conta o algoritmo de aprendizagem, muitos dos atributos selecionados poderão não otimizar o modelo de previsão e os critérios de seleção são difíceis de determinar. (CAVIQUE *et al.*, 2013)

FOCUS e *Relief* são os métodos mais conhecidos que recorrem à abordagem *Filter*. *FOCUS* faz uso da existência de inconsistências, procurando grupos de atributos que minimizam a quantidade destes conflitos. Já o método *Relief* faz a avaliação de um conjunto de atributos através de um processo de avaliação baseado na diferença entre a distância das observações da mesma classe e de classe diferente, que sejam mais próximas. (CAVIQUE *et al.*, 2013)

2.2.2. Abordagem *Wrapper*

A grande diferença entre as abordagens *Wrapper* e *Filter* reside no critério de avaliação. Os métodos que seguem a abordagem *Wrapper* realizam a seleção de subconjuntos de atributos durante o processo de treino, através de um algoritmo de aprendizagem específico, ao qual recorrem para proceder à avaliação (KUMAR, MINZ, 2014), com maiores custos de processamento, logo mais lentos (BOLÓN-CANEDO *et al.*, 2013). Se o número de observações for insuficiente, aumenta o risco de *overfitting* (KUMAR, MINZ, 2014), ou seja, o modelo obtido encaixar perfeitamente nos dados de treino, mas a precisão diminuir bastante quando aplicado noutros grupos de dados. Consegue capturar as dependências entre atributos (BOLÓN-CANEDO *et al.*, 2013). Exemplos de métodos: *Wrapper-C4.5* e *Wrapper SVM* (BOLÓN-CANEDO *et al.*, 2013).

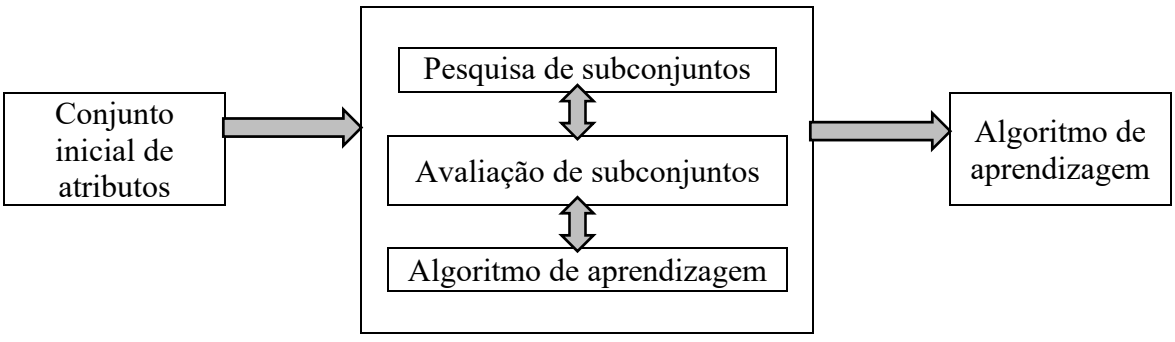

Figura 2.3 – Abordagem Wrapper.
(Fonte: JOHN et al., 1994)

Neste caso, como os critérios de seleção são determinados recorrendo ao algoritmo de aprendizagem, eles são fáceis de estimar. Mas, pela mesma razão, são computacionalmente mais pesados e pouco intuitivos. Também, não identificam dependências estatísticas entre atributos, podendo, o subconjunto determinado, não ser o que melhor explica o modelo, destruindo a semântica subjacente ao sistema. (CAVIQUE *et al.*, 2013)

2.2.3. Abordagem *Embedded*

A abordagem *Embedded* tenta combinar as vantagens das abordagens anteriores. Procura o subconjunto ótimo de acordo com o algoritmo de aprendizagem durante o processo de treino. Tem baixo custo computacional e consegue detetar dependências entre atributos (BOLÓN-CANEDO *et al.*, 2013). O algoritmo faz uso do seu próprio processo de seleção para realizar, simultaneamente, a seleção de atributos e a avaliação, ou seja, usa uma métrica própria durante a seleção de atributos (KUMAR, MINZ, 2014). *FS-Percepton* e *SVM-RFE* (BOLÓN-CANEDO *et al.*, 2013) são métodos que aplicam esta abordagem.

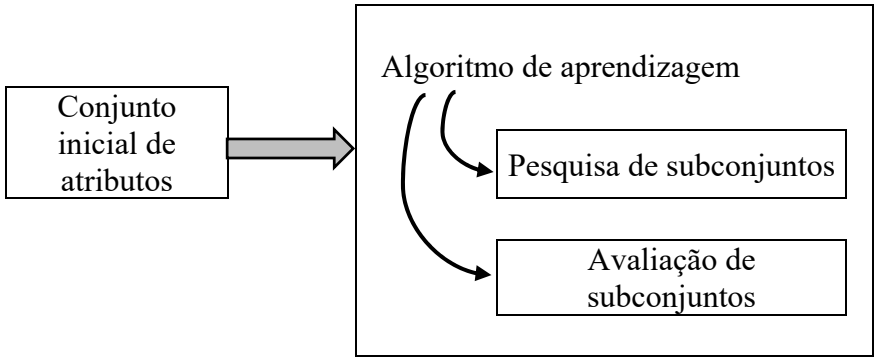

Figura 2.4 – Abordagem Embedded.
(Adaptado de: BOLÓN-CANEDO et al., 2013)

2.3. Relevância dos atributos

O ideal, no processo de seleção, para obter um subconjunto de atributos, seria que todos os atributos escolhidos fossem relevantes. Para que isto aconteça, há que definir o conceito de atributo relevante.

Kumar e Minz (2014) compilaram seis definições de relevância, de acordo com ser relevante em relação ao quê:

- Relevante para com o objetivo – um atributo x_i é relevante para um objetivo C (função que associa a cada observação a respetiva classe) se existirem duas observações (A e B) que difiram apenas no valor de x_i e $C(A) \neq C(B)$;
- Forte relevância para com os dados – um atributo x_i é fortemente relevante para com um conjunto de dados do sistema (dados de um subconjunto de atributos), se existirem duas observações (A e B) que difiram apenas no valor de x_i , relativamente ao conjunto de dados considerado, e $C(A) \neq C(B)$;
- Fraca relevância para com os dados – um atributo x_i é fracamente relevante para com os dados se existir pelo menos um subconjunto de atributos que contenha x_i, onde, nesse subconjunto, x_i não é fortemente relevante para com os dados;
- Relevância sendo uma medida de complexidade – numa amostra de dados e tendo um conjunto de objetivos C, a medida corresponde ao número de atributos relevantes, considerando a primeira definição, relativamente a um objetivo de C cujo erro sobre S seja menor e tenha menos atributos relevantes;
- Utilidade incremental – dada uma amostra de dados, um algoritmo de pesquisa e um subconjunto de atributos X', um atributo x_i é incrementalmente útil para o algoritmo, relativamente a X', se a precisão da hipótese que o algoritmo produz, usando o conjunto de atributos $\{x_i\} \cup X'$ é melhor do que a precisão alcançada apenas por X'.
- Relevância da entropia – a entropia de x para y é definida por $I(x,y) / H(y)$, onde $I(x,y) = H(x) – H(x \mid y)$ e H é a função entropia de um atributo.

A relevância de um atributo é qualificada em três níveis (KOHAVI, JOHN, 1997):

- Fortemente relevante;
- Fracamente relevante;
- Irrelevante.

Um atributo é fortemente relevante se não puder ser retirado do subconjunto de atributos sem que diminua a avaliação do subconjunto. Será fracamente relevante se, por vezes, contribuir para aumentar a avaliação do subconjunto. Um atributo é irrelevante se não for fortemente nem fracamente relevante (KOHAVI, JOHN, 1997).

2.4. Algoritmos de seleção de atributos

Têm sido propostos muitos algoritmos de seleção de atributos. A eficácia dos mesmos é difícil de determinar sem saber, de antemão, quais os atributos relevantes (BOLÓN-CANEDO *et al.*, 2013). Existem muitas medidas de performance definidas na literatura, nomeadamente, medidas de precisão, cálculo de rácios, recursos computacionais, etc. (KUMAR, MINZ, 2014). Além disso, o volume de dados pode ser enorme, existir uma grande quantidade de atributos irrelevantes ou redundantes e, ainda, uma grande quantidade de observações, ou registos, inconsistentes. Na literatura encontram-se muitos estudos comparativos de métodos de seleção.

Em consequência do acima exposto, têm surgido muitos novos algoritmos de seleção de atributos, com diferentes estratégias (KUMAR, MINZ, 2014):

- Recorrer à aplicação de vários métodos de seleção para obter melhores resultados;
- Combinar a seleção com outras técnicas, como a extração de atributos e técnicas de árvore;
- Melhorar algoritmos existentes;
- Criar novos métodos para responder a problemas que não se enquadrem em nenhum dos métodos existentes;
- Combinar dois ou mais métodos de seleção, de forma a obter o melhor de cada um.

Tabela 2.1 – Algoritmos de seleção de atributos.

			Estratégias de seleção		
			Exaustiva	Sequencial	Aleatória
Abordagens	*Filter*	Distância	B&B, BFF, BOBRO, OBLIVIIN	Relief, RelifS, SFS, Segen's, SBS	
		Informação	MDLM, CARDIE	DTM, Koller's, FG, FCBF, BSE, Dash's, SBUD	
		Dependência	Bobrowski's	CFS, RRESET, POE+ACC, DVMM, Mitra's	
		Consistência	FOCUS, ABB, MIFESI, Schlimmer's	*Rough Sets*, LAD, LAID, *Set Cover*, WINNOW	LIV, QBB, LVF
	Wrapper		BS, AMB&B, FSLC, FSBC, CARDIE, OBLIVIIN	SBS-SLASH, WSFG, WSBG, BDS, PQSS, RC, SS, Queiros's, BSE, K2-AS, RACE, SBS-W, SBS-SLASH, AICC, FSSEM, ELSA	SA, RGSS, LVW, RMHC-PF, GA, RVE
	Embedded			BBHFS, Xing's, Dash-Liu's	

(Adaptado de: KUMAR, MINZ, 2014)

3. LAID e as metodologias subjacentes

A Análise Lógica de Dados Inconsistentes (LAID) surge da combinação de duas metodologias *Filter* desenvolvidas nos finais do século XX: a Teoria dos *Rough Sets* e a LAD. A LAID herda muitas das qualidades de cada uma destas teorias, sendo capaz de lidar com dados inconsistentes e com classes não dicotomizadas, características dos *Rough Sets*, bem como, tem a eficácia computacional da LAD. (CAVIQUE *et al.*, 2011)

Neste capítulo far-se-á a descrição teórica de cada uma destas três metodologias, complementada com exemplos simples e esclarecedores.

3.1. Rough Sets

A teoria dos *Rough Sets* é uma abordagem matemática, proposta por Zdzislaw Pawlak, em 1982, para análise de dados inconsistentes (ZHANG *et al.*, 2016). Esta abordagem tem obtido bons resultados em muitos domínios, nomeadamente, na seleção de atributos, a encontrar dependências entre atributos e na identificação de padrões num conjunto de dados. Uma das grandes aplicações dos *Rough Sets* tem sido na área da inteligência artificial, nomeadamente, na extração de conhecimento, tratamento de dados, suporte a decisões e reconhecimento de padrões (PAWLAK, 2003).

Pelas suas características, os *Rough Sets* têm tido uma atenção crescente por parte da comunidade científica, corroborada pela grande quantidade de artigos e outras publicações (PAWLAK, 2003), em especial, nas áreas de *Data Mining*, reconhecimento de padrões e classificação, na indústria energética, processamento de imagens e outros dados de grande dimensão, desenvolvimento de sistemas inteligentes, medicina, gestão, engenharia e muitas outras (ZHANG *et al.*, 2016).

A teoria dos *Rough Sets* introduziu novos conceitos e notações. Por ser relativamente nova e por ter despertado o interesse de um grande número de investigadores, estes novos conceitos não estão estandardizados (SASSI, 2006), pelo que convém clarificá-los.

3.1.1. Indiscernibilidade e conjuntos elementares

Um conceito fundamental na teoria dos *Rough Sets* é o de indiscernibilidade (PAWLAK, 2002). A indiscernibilidade manifesta-se quando, num conjunto, existem elementos que não se conseguem distinguir, parecendo ser repetições do mesmo elemento (PAWLAK, 2002). Por exemplo, se duas observações tiverem os mesmos valores num conjunto de atributos, elas dizem-se indiscerníveis (PILA, MONARD, 2001). No entanto, poderão divergir se forem acrescentados novos atributos.

Um espaço aproximado é composto por um conjunto universo (U), finito, não vazio, e por uma operação de equivalência (R) entre dois elementos de U, que goza das propriedades reflexiva (xRx), simétrica (se xRy então yRx) e transitiva (se xRy e yRz então xRz) (PILA, MONARD, 2001). Dados dois elementos de U, x e y, se xRy então os dois elementos são indiscerníveis no espaço aproximado. Algumas extensões da teoria dos *Rough Sets* não exigem que a relação de equivalência goze da propriedade transitiva, sendo chamada relação de tolerância ou similaridade (PAWLAK, 2003).

A classe de equivalência de um elemento $x \in U$, denotada por $[x]_R$, é o conjunto de todos os elementos $y \in U$, tais que xRy. Conjuntos elementares de U são conjuntos cujos elementos são indiscerníveis, logo correspondem às classes de equivalência (SASSI, 2006).

Na seleção de atributos, U corresponde ao conjunto de observações ou registos de uma base de dados. O conjunto de atributos designa-se por A. Cada atributo tem um conjunto de valores possíveis, que poderão ser binários, numéricos ou literais. Duas observações são indiscerníveis se tiverem o mesmo valor para cada atributo. Ao conjunto das observações, com os atributos e respetivos valores, dá-se o nome de sistema de informação (PAWLAK, 2003).

A Tabela 3.1 apresenta um exemplo de um sistema de informação, onde $A=\{a1, a2, a3\}$ e $U=\{o1,o2,o3,o4,o5,o6,o7,o8\}$. Verifica-se que as observações $o1$, $o5$ e $o8$ são indiscerníveis, o mesmo se passando com as observações $o4$ e $o6$.

Tabela 3.1 – Representação de um sistema de informação.

Atributos / Observações	$a1$	$a2$	$a3$
$o1$	1	2	1
$o2$	1	1	1
$o3$	0	2	0
$o4$	0	3	0
$o5$	1	2	1
$o6$	0	3	0
$o7$	0	4	1
$o8$	1	2	1

A Relação de Indiscernibilidade associada ao conjunto de atributos A, $IND(A)$, é definida pela operação de equivalência (PAWLAK, 2003):

$$IND(A) = \{x,y \in U: \forall a \in A, a(x) = a(y)\} \tag{3.1}$$

Se x e y verificam a relação $IND(A)$, então são indiscerníveis, ou indistinguíveis, tendo em conta os atributos A (PAWLAK, 2003). A notação $[x]_{IND(A)}$ representa o conjunto elementar ao qual uma observação x pertence, isto é, a classe de equivalência de x.

Um conjunto elementar de um sistema de informação é um subconjunto de U que contém todos os elementos de U que verificam a Relação de Indiscernibilidade associada aos atributos A (PAWLAK, 2002). Todos os conjuntos elementares de um sistema de informação, U/A, formam uma partição de U. No sistema de informação da Tabela 3.1, tendo em conta todos os atributos, podem-se identificar cinco conjuntos elementares: $\{o1, o5, o8\}$; $\{o2\}$; $\{o3\}$; $\{o4, o6\}$; $\{o7\}$. Qualquer união finita de conjuntos elementares designa-se por conjunto definível num sistema de informação.

Em muitos casos, a cada observação de um sistema de informação está associada uma classificação de resposta designada por classe (C). A classe poderá ser considerada como um atributo de decisão. Um sistema de informação com uma classe associada designa-se por Sistema de Decisão (PAWLAK, 2003). Cada linha do sistema de decisão representa uma regra que, dada a satisfação de determinadas condições (valores dos respetivos atributos) é obtido um resultado (valor da classe) (PAWLAK, 2002).

A Tabela 3.2 apresenta o sistema de informação, definido na Tabela 3.1, com uma classe associada, constituindo um sistema de decisão.

Tabela 3.2 – Representação de um sistema de decisão.

Atributos / Observações	$a1$	$a2$	$a3$	Classe c
$o1$	1	2	1	0
$o2$	1	1	1	2
$o3$	0	2	0	1
$o4$	0	3	0	2
$o5$	1	2	1	1
$o6$	0	3	0	0
$o7$	0	4	1	2
$o8$	1	2	1	1

Duas observações com os mesmos valores para cada atributo e para a classe dizem-se redundantes:

$$x \text{ e } y \text{ são redundantes sse } (\forall a \in A,\ a(x) = a(y)) \wedge c(x) = c(y) \tag{3.2}$$

Observações inconsistentes têm os mesmos valores para cada atributo, mas classes com valores diferentes:

$$x \text{ e } y \text{ são inconsistentes sse } (\forall a \in A,\ a(x) = a(y)) \wedge c(x) \neq c(y) \tag{3.3}$$

As observações redundantes, para determinado sistema de decisão, podem ser eliminadas, ficando apenas uma delas.

Neste novo sistema de decisão, representado na Tabela 3.3, tem-se U={$o1$;$o2$;$o3$;$o4$;$o5$;$o6$;$o7$}, A={$a1$;$a2$;$a3$} e C={c}. No sistema de informação correspondente, tendo em conta todos os atributos, podem-se identificar cinco conjuntos elementares: {$o1$, $o5$}; {$o2$}; {$o3$}; {$o4$, $o6$}; {$o7$}.

Tabela 3.3 – Sistema de decisão sem observações redundantes.

Atributos / Observações	$a1$	$a2$	$a3$	Classe c
$o1$	1	2	1	0
$o2$	1	1	1	2
$o3$	0	2	0	1
$o4$	0	3	0	2
$o5$	1	2	1	1
$o6$	0	3	0	0
$o7$	0	4	1	2

3.1.2. Conjuntos aproximados

Por observação do sistema de decisão da Tabela 3.3, uma observação com os mesmos valores de atributos de $o2$ ou $o7$, pertencerá à classe 2. Uma observação com os mesmos valores de atributos de $o3$, pertencerá à classe 1. Mas, para uma observação com os mesmos valores de atributos de $o1$ e $o5$, o valor da sua classe poderia ser 0 ou 1. Da mesma forma, a uma observação com os mesmos valores de atributos de $o4$ e $o6$ poderá corresponder o valor da classe 0 ou 2. Dadas as inconsistências verificadas, a teoria dos *Rough Sets* consegue dar resposta e realizar a seleção de atributos (SASSI, 2006).

Num sistema de decisão, seja X um conjunto de observações de U com determinado(s) valor(es) da classe, $C_1 \subseteq C$:

$$X = \{x \in U: c(x) \in C_1\} \tag{3.4}$$

A partir de um determinado conjunto X, podem-se definir outros dois conjuntos (PAWLAK, 2002)(PAWLAK, 2003):

- Aproximação inferior $\underline{A}(X)$ – maior conjunto definível contido em X – conjunto das observações com valores dos atributos iguais aos dos elementos de $\underline{A}(X)$ terão, seguramente, valor de classe pertencente a C_1, isto é, pertencerão a X;

- Aproximação superior $\bar{A}(X)$ – menor conjunto definível que contém X – conjunto das observações com valores dos atributos iguais aos dos elementos de $\bar{A}(X)$ poderão ter valor de classe pertencente a C_1, isto é, serão possíveis elementos de X.

Usando a notação $[x]_{IND(A)}$ para representar o conjunto elementar ao qual uma observação x pertence, vem (PAWLAK, 2003):

$$\bar{A}(X) = \{x \in U: [x]_{IND(A)} \cap X \neq \varnothing\} \tag{3.5}$$

$$\underline{A}(X) = \{x \in U: [x]_{IND(A)} \subseteq X\} \tag{3.6}$$

Região negativa de X, $NEG(X)$, é o conjunto de todas as observações de U que não pertencem a $\bar{A}(X)$. Elas terão, seguramente, valor de classe que não pertencente a C_1, isto é, não pertencerão a X (PAWLAK, 2003).

Região fronteira de X, $FR(X)$, é o conjunto de todas as observações de U que pertencem a $\bar{A}(X)$ e não pertencem a $\underline{A}(X)$. A região fronteira corresponde ao resultado da operação de conjuntos $\bar{A}(X) - \underline{A}(X)$ (ZHANG *et al.*, 2016). Observações com valores de atributos iguais a um dos elementos de $FR(X)$ não poderão ser classificadas, com absoluta certeza, como pertencentes a X, nem como não pertencentes a X (PAWLAK, 2003).

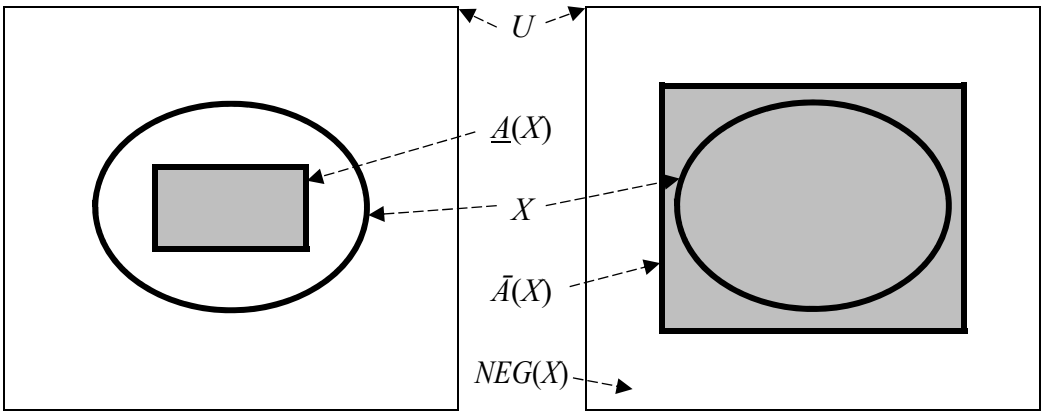

Figura 3.1 – Esquema ilustrativo da Aproximação Inferior e Superior de X.
(Adaptado de: PAWLAK, 2003)

Por exemplo, tendo em conta a Tabela 3.3, seja X o conjunto de observações de U cuja classe é 2, vem:

$$X = \{o2, o4, o7\}$$

$$\underline{A}(X) = \{o2\} \cup \{o7\} = \{o2, o7\}$$

$$\bar{A}(X) = \{o2\} \cup \{o4, o6\} \cup \{o7\} = \{o2, o4, o6, o7\}$$

$$NEG(X) = \{o1, o3, o5\}$$

$$FR(X) = \{o4, o6\}$$

Um conjunto X é impreciso ou aproximado (*rough*) se a sua região fronteira não for vazia, caso contrário, é preciso (*crisp*) (PAWLAK, 2002).

Tal como X, seja Y um conjunto de observações de U correspondentes a determinado(s) valor(es) da classe, $C_2 \subseteq C$: $Y = \{x \in U: c(x) \in C_2\}$. Segue-se a lista de propriedades dos conjuntos aproximados (ZHANG *et al.*, 2016)(PAWLAK, 2003), onde $\sim X = U - X$:

$$\bar{A}(X \cup Y) = \bar{A}(X) \cup \bar{A}(Y)$$

$$\underline{A}(X \cup Y) \supseteq \underline{A}(X) \cup \underline{A}(Y)$$

$$\bar{A}(X \cap Y) \subseteq \bar{A}(X) \cap \bar{A}(Y)$$

$$\underline{A}(X \cap Y) = \underline{A}(X) \cap \underline{A}(Y)$$

$$\bar{A}(X - Y) \subseteq \bar{A}(X) - \underline{A}(Y)$$

$$\underline{A}(X - Y) = \underline{A}(X) - \bar{A}(Y)$$

$$\sim\underline{A}(X) = \bar{A}(\sim X)$$

$$\sim\bar{A}(X) = \underline{A}(\sim X)$$

$$\sim(\underline{A}(X) \cup \underline{A}(Y)) = \bar{A}(\sim X) \cap \bar{A}(\sim Y)$$

$$\sim(\underline{A}(X) \cup \bar{A}(Y)) = \bar{A}(\sim X) \cap \underline{A}(\sim Y)$$

$$\sim(\bar{A}(X) \cup \underline{A}(Y)) = \underline{A}(\sim X) \cap \bar{A}(\sim Y)$$

$$\sim(\bar{A}(X) \cup \bar{A}(Y)) = \underline{A}(\sim X) \cap \underline{A}(\sim Y)$$

$$\sim(\underline{A}(X) \cap \underline{A}(Y)) = \bar{A}(\sim X) \cup \bar{A}(\sim Y)$$

$$\sim(\underline{A}(X) \cap \bar{A}(Y)) = \bar{A}(\sim X) \cup \underline{A}(\sim Y)$$

$$\sim(\bar{A}(X) \cap \underline{A}(Y)) = \underline{A}(\sim X) \cup \bar{A}(\sim Y)$$

$$\sim(\bar{A}(X) \cap \bar{A}(Y)) = \underline{A}(\sim X) \cup \underline{A}(\sim Y)$$

$$X \subseteq Y \Rightarrow \underline{A}(X) \subseteq \underline{A}(Y) \wedge \bar{A}(X) \subseteq \bar{A}(Y)$$

A partir das propriedades tem-se (ZHANG *et al.*, 2016)(PAWLAK, 2003):

$$\underline{A}(X) \subseteq X \subseteq \bar{A}(X)$$

$$\underline{A}(\varnothing) = \bar{A}(\varnothing) = \varnothing$$

$$\underline{A}(U) = \bar{A}(U) = U$$

$$\underline{A}(\underline{A}(X)) = \bar{A}(\underline{A}(X)) = \underline{A}(X)$$

$$\bar{A}(\bar{A}(X)) = \underline{A}(\bar{A}(X)) = \bar{A}(X)$$

Dependendo das respetivas aproximações inferior e superior, podem-se definir quatro tipos de incerteza relativamente a um conjunto X (PAWLAK, 2003):

- X é grosseiramente A-definível se e só se $\underline{A}(X) \neq \varnothing \wedge \bar{A}(X) \neq U$. Neste caso, para determinado sistema de decisão, existem observações de U que pertencem, inequivocamente, a X e existem observações de U que não pertencem, inequivocamente, a X.

- X é internamente A-indefinível se e só se $\underline{A}(X) = \varnothing \wedge \bar{A}(X) \neq U$. Neste caso, para determinado sistema de decisão, existem observações de U que não pertencem, inequivocamente, a X, mas já não se pode afirmar que existam observações de U que pertençam, inequivocamente, a X.

- X é externamente A-indefinível se e só se $\underline{A}(X) \neq \varnothing \wedge \bar{A}(X) = U$. Neste caso, para determinado sistema de decisão, existem observações de U que pertencem, inequivocamente, a X, mas já não se pode afirmar que existam observações de U que não pertençam, inequivocamente, a X.

- X é totalmente A-indefinível se e só se $\underline{A}(X) = \varnothing \wedge \bar{A}(X) = U$. Neste caso, para determinado sistema de decisão, não existem observações de U que pertençam, inequivocamente, a X, nem que não pertençam, inequivocamente, a X.

No sistema de decisão da Tabela 3.3, sendo X o conjunto de observações de U cuja classe é 2, tem-se $\underline{A}(X) = \{o2, o7\}$ e $\bar{A}(X) = \{o2, o4, o6, o7\}$. Logo, X é grosseiramente A-definível.

Existem três medidas que permitem quantificar a qualidade das aproximações de X (PAWLAK, 2003)(RISSINO, LAMBERT-TORRES, 2009):

- Coeficiente de precisão da aproximação, $\alpha_A(X) = \frac{\#\underline{A}(X)}{\#\overline{A}(X)}$, (3.7)

  mede o grau de precisão de X. Como $\#\underline{A}(X) \leq \#\bar{A}(X)$, vem $0 \leq \alpha_A(X) \leq 1$. Se $\alpha_A(X) = 1$, X é preciso, caso contrário é aproximado. O grau de imprecisão é dado por $1 - \alpha_A(X)$.

- Coeficiente de precisão da aproximação inferior, $\alpha_A(\underline{A}(X)) = \frac{\#\underline{A}(X)}{\#U}$. (3.8)

  Como $\#\underline{A}(X) \leq \#U$, vem $0 \leq \alpha_A(\underline{A}(X)) \leq 1$. Se $\alpha_A(\underline{A}(X)) = 1$, todas as observações de U pertencem, inequivocamente, a X. Se $\alpha_A(\underline{A}(X)) = 0$, nenhuma observação de U pertence, inequivocamente a X.

- Coeficiente de precisão da aproximação superior, $\alpha_A(\bar{A}(X)) = \frac{\#\overline{A}(X)}{\#U}$. (3.9)

  Como $\#\bar{A}(X) \leq \#U$, vem $0 \leq \alpha_A(\bar{A}(X)) \leq 1$. Se $\alpha_A(\bar{A}(X)) = 1$, nenhuma observação de U não pertence, inequivocamente a X. Se $\alpha_A(\bar{A}(X)) = 0$, todas as observações de U não pertencem, inequivocamente, a X.

Voltando ao exemplo da Tabela 3.3:

$$\alpha_A(X) = \frac{\#\{o2,o7\}}{\#\{o2,o4,o6,o7\}} = \frac{2}{4} = 0{,}5$$ (o grau de precisão de X é de 50%)

$$\alpha_A(\underline{A}(X)) = \frac{\#\{o2,o7\}}{\#\{o1,o2,o3,o4,o5,o6,o7\}} = \frac{2}{7} \approx 0{,}29$$ (29% das observações de U pertence, inequivocamente, a X).

$$\alpha_A(\bar{A}(X)) = \frac{\#\{o2,o4,o6,o7\}}{\#\{o1,o2,o3,o4,o5,o6,o7\}} = \frac{4}{7} \approx 0{,}57$$ (57% das observações de U possivelmente pertencem a X).

3.1.3. Grau de relevância dos atributos

Para determinar o grau de relevância de um atributo, há que, primeiro, definir a dependência entre a classe e os atributos.

Sejam A_1 e A_2 dois subconjuntos de atributos de A ($A_1 \subseteq A$ e $A_2 \subseteq A$), A_1 depende totalmente de A_2 se todos os valores dos atributos de A_1 forem univocamente determinados pelos valores dos atributos de A_2 (PAWLAK, 2003).

A dependência de um conjunto de valores da classe (C), relativamente ao conjunto de atributos (A), pode ser quantificada recorrendo ao respetivo coeficiente de qualidade da aproximação inferior 3.8 (PAWLAK, 2003), onde U/C denota a partição de U segundo os valores de C:

$$\gamma(A,C) = \frac{\#\left(\bigcup_{X \in U/C}(\underline{A}(X))\right)}{\#U} = \sum_{X \in U/C} \frac{\#\underline{A}(X)}{\#U} \quad (3.10)$$

A dependência da classe, em relação aos atributos, pode ser classificada segundo o valor de $\gamma(A,C)$ (PAWLAK, 2003):

- Se $\gamma(A,C) = 1$, há total dependência desses valores da classe em relação ao conjunto de atributos, ou seja, esses valores da classe são determinados, inequivocamente, pelos valores dos atributos.

- Se $0 < \gamma(A,C) < 1$, existe dependência parcial dos valores da classe em relação ao conjunto de atributos. Neste caso, indica que existem atributos que podem ser retirados de A, sem afetar a correspondência entre os valores dos atributos e os, respetivos, valores da classe (seleção de atributos).

- Se $\gamma(A,C) = 0$, os valores da classe são independentes do conjunto de atributos.

Tendo em conta a Tabela 3.3, vem:

$$\underline{A}(\{x \in U: c(x) = 0\}) = \underline{A}(\{o1, o6\}) = \{\}$$

$$\underline{A}(\{x \in U: c(x) = 1\}) = \underline{A}(\{o3, o5\}) = \{o3\}$$

$$\underline{A}(\{x \in U: c(x) = 2\}) = \underline{A}(\{o2, o4, o7\}) = \{o2, o7\}$$

$$\gamma(A,C) = \frac{\#\{o2,o3,o7\}}{\#\{o1,o2,o3,o4,o5,o6,o7\}} = \frac{3}{7} \approx 0{,}43$$

A partir da análise do grau de dependência de C em relação a A, conclui-se que os valores da classe dependem em 43% dos atributos.

Outro conceito importante para a seleção de atributos é a importância que cada um tem para a tomada de decisão (obter o valor da classe). Neste contexto, surge a definição de significância de um atributo (a), determinada a partir do grau de dependência de C em relação a A, por Sassi (2006):

$$\sigma_C(a) = 1 - \frac{\gamma(A-\{a\},C)}{\gamma(A,C)} \tag{3.11}$$

A significância de um atributo corresponderá ao erro que se comete na dependência de C em relação a A, quando o atributo é removido de A (SASSI, 2006). Um atributo é dispensável se a sua significância for nula (PILA, MONARD, 2001).

Designe-se por A^* o conjunto de atributos de A após retirado o atributo $a3$, isto é, $A^* = A - \{a3\}$.

Tabela 3.4 – Sistema de decisão com o atributo a3 removido.

Atributos / Observações	$a1$	$a2$	Classe c
$o1$	1	2	0
$o2$	1	1	2
$o3$	0	2	1
$o4$	0	3	2
$o5$	1	2	1
$o6$	0	3	0
$o7$	0	4	2

A partir da Tabela 3.4, vem:

$$\underline{A}^*(\{x \in U: c(x) = 0\}) = \underline{A}^*(\{o1, o6\}) = \{\}$$
$$\underline{A}^*(\{x \in U: c(x) = 1\}) = \underline{A}^*(\{o3, o5\}) = \{o3\}$$
$$\underline{A}^*(\{x \in U: c(x) = 2\}) = \underline{A}^*(\{o2, o4, o7\}) = \{o2, o7\}$$
$$\gamma(A^*,C) = \frac{\#\{o2,o3,o7\}}{\#\{o1,o2,o3,o4,o5,o6,o7\}} = \frac{3}{7} \approx 0{,}43$$

Logo,

$$\sigma_X(a3) = 1 - \frac{\gamma(A^*,C)}{\gamma(A,C)} = 1 - \frac{0{,}43}{0{,}43} = 0$$

Donde se conclui que o atributo $a3$ não tem significado para determinar os valores da classe, podendo ser retirado.

3.1.4. Seleção de atributos

Na seleção de atributos, um conjunto de atributos que permita extrair o mesmo conhecimento de uma base de dados que a totalidade dos mesmos, diz-se um reduto (PAWLAK, 2003). Seja $A^* \subseteq A$, A^* é um reduto de A, se todos os atributos de $A–A^*$ forem dispensáveis e $\forall x \in U$, $[x]_{IND(A^*)} = [x]_{IND(A)}$. O conjunto de todos os redutos de A é representado por $RED(A)$ (PILA, MONARD, 2001).

No processo de remoção de um atributo há que determinar os conjuntos elementares de cada atributo. Se os conjuntos elementares de um atributo forem iguais aos do conjunto total de atributos (A), então esse atributo será um reduto de A, ou seja, todos os outros atributos poderão ser retirados. Caso não haja nenhum atributo nestas condições, determinam-se os conjuntos elementares para todos os subconjuntos de dois atributos, depois para subconjuntos de três, e assim sucessivamente, até encontrar um, ou vários, subconjuntos de A cujos conjuntos elementares sejam iguais aos de A, sendo, esse subconjunto, um reduto de A. Se para nenhum subconjunto de A, exceto o próprio A, se obtém um reduto, isto significa que o conjunto de atributos não pode ser reduzido (PAWLAK, 2003).

Voltando ao exemplo da Tabela 3.3, os conjuntos elementares para cada atributo estão identificados na Tabela 3.5.

Tabela 3.5 – Conjuntos elementares para subconjuntos de A com um elemento.

Subconjuntos de 1 atributo	Valores possíveis	Conjuntos elementares
$\{a1\}$	0	$\{o3, o4, o6, o7\}$
	1	$\{o1, o2, o5\}$
$\{a2\}$	1	$\{o2\}$
	2	$\{o1, o3, o5\}$
	3	$\{o4, o6\}$
	4	$\{o7\}$
$\{a3\}$	0	$\{o3, o4, o6\}$
	1	$\{o1, o2, o5, o7\}$

Para todos os atributos, conforme já determinado, podem identificar-se cinco conjuntos elementares: $\{o1, o5\}$; $\{o2\}$; $\{o3\}$; $\{o4, o6\}$; $\{o7\}$. Por comparação com a Tabela 3.5, nenhum atributo tem os mesmos conjuntos elementares. Continuando, considerem-se os subconjuntos de dois atributos e, respetivos, conjuntos elementares (ver Tabela 3.6).

Tabela 3.6 – Conjuntos elementares para subconjuntos de A com dois elementos.

Subconjuntos de 2 atributos	Valores possíveis	Conjuntos elementares
{*a*1, *a*2}	*a*1=0 ∧ *a*2=1	
	*a*1=0 ∧ *a*2=2	{*o*3}
	*a*1=0 ∧ *a*2=3	{*o*4, *o*6}
	*a*1=0 ∧ *a*2=4	{*o*7}
	*a*1=1 ∧ *a*2=1	{*o*2}
	*a*1=1 ∧ *a*2=2	{*o*1, *o*5}
	*a*1=1 ∧ *a*2=3	
	*a*1=1 ∧ *a*2=4	
{*a*1, *a*3}	*a*1=0 ∧ *a*3=0	{*o*3, *o*4, *o*6}
	*a*1=0 ∧ *a*3=1	{*o*7}
	*a*1=1 ∧ *a*3=0	
	*a*1=1 ∧ *a*3=1	{*o*1, *o*2, *o*5}
{*a*2, *a*3}	*a*2=1 ∧ *a*3=0	
	*a*2=1 ∧ *a*3=1	{*o*2}
	*a*2=2 ∧ *a*3=0	{*o*3}
	*a*2=2 ∧ *a*3=1	{*o*1, *o*5}
	*a*2=3 ∧ *a*3=0	{*o*4, *o*6}
	*a*2=3 ∧ *a*3=1	
	*a*2=4 ∧ *a*3=0	
	*a*2=4 ∧ *a*3=1	{*o*7}

Pela análise da Tabela 3.6, conclui-se que os atributos *a*1 e *a*2 têm os mesmos conjuntos elementares que *A*, logo formam um reduto de *A*, e, por consequência, o atributo *a*3 pode ser retirado sem perda de informação. De igual modo, os atributos *a*2 e *a*3 formam um reduto de *A*. As Tabela 3.4 e Tabela 3.7 mostram que a teoria dos *Rough Sets* consegue lidar com as inconsistências, mas não as remove.

Tabela 3.7 – Sistema de decisão com o atributo a1 removido.

Atributos / Observações	*a*2	*a*3	Classe *c*
*o*1	2	1	0
*o*2	1	1	2
*o*3	2	0	1
*o*4	3	0	2
*o*5	2	1	1
*o*6	3	0	0
*o*7	4	1	2

3.1.5. Função de discernibilidade

O processo de seleção de atributos, visto atrás, tem um grande peso computacional, nomeadamente, no caso de sistemas de informação com milhares de atributos. A função de discernibilidade permite realizar a seleção de atributos de uma forma computacionalmente mais eficaz (PAWLAK, 2003). Para tal, há que construir a matriz de discernibilidade.

A matriz de discernibilidade, M, é uma matriz simétrica, de dimensão $n \times n$, onde n é o número de observações do sistema de informação ($\#U$) e cada elemento de M, $m(i,j)$, é obtido para um par de observações e corresponde ao conjunto de atributos de A cujos valores diferem nas duas observações (PAWLAK, 2003). Formalmente, será (PAWLAK, 2003):

$$m(i,j) = \{a \in A: a(o(i)) \neq a(o(j))\} \tag{3.12}$$

Determinada a matriz M, há que calcular a função de discernibilidade F, uma função booleana composta pela operação de conjunção dos elementos que correspondem à disjunção dos atributos constantes em cada célula da matriz M (PAWLAK, 2003) (PILA, MONARD, 2001).

$$F(A) = \wedge[\vee[a(k) \in m(i,j)]] \tag{3.13}$$

Onde $1 \leq k \leq \#A$ e $1 \leq j < i \leq \#U$.

Aplicando, à expressão obtida para $F(A)$, as propriedades da álgebra booleana, obtém-se a expressão simplificada de F, que indica a redução, ou reduções, de A.

Continuando com o exemplo do sistema de decisão da Tabela 3.3, a matriz de discernibilidade vem:

Tabela 3.8 – Matriz de discernibilidade.

	$o1$	$o2$	$o3$	$o4$	$o5$	$o6$	$o7$
$o1$	$\varnothing$						
$o2$	$\{a2\}$	$\varnothing$					
$o3$	$\{a1, a3\}$	$\{a1, a2, a3\}$	$\varnothing$				
$o4$	$\{a1, a2, a3\}$	$\{a1, a2, a3\}$	$\{a2\}$	$\varnothing$			
$o5$	$\varnothing$	$\{a2\}$	$\{a1, a3\}$	$\{a1, a2, a3\}$	$\varnothing$		
$o6$	$\{a1, a2, a3\}$	$\{a1, a2, a3\}$	$\{a2\}$	$\varnothing$	$\{a1, a2, a3\}$	$\varnothing$	
$o7$	$\{a1, a2\}$	$\{a1, a2\}$	$\{a2, a3\}$	$\{a2, a3\}$	$\{a1, a2\}$	$\{a2, a3\}$	$\varnothing$

Dada a matriz obtida no exemplo acima, obtém-se a seguinte função de discernibilidade:

$$F(A) = (a2) \wedge (a1 \vee a3) \wedge (a1 \vee a2 \vee a3) \wedge (a1 \vee a2 \vee a3) \wedge (a1 \vee a2 \vee a3) \wedge (a2) \wedge (a2) \wedge (a1 \vee a3) \wedge (a1 \vee a2 \vee a3) \wedge (a1 \vee a2 \vee a3) \wedge (a1 \vee a2 \vee a3) \wedge (a2) \wedge (a1 \vee a2 \vee a3) \wedge (a1 \vee a2) \wedge (a1 \vee a2) \wedge (a2 \vee a3) \wedge (a2 \vee a3) \wedge (a1 \vee a2) \wedge (a2 \vee a3)$$

Pelas propriedades da álgebra booleana, nomeadamente, a comutatividade e a idempotência da conjunção (retirando os operandos repetidos), vem:

$$F(A) = (a2) \wedge (a1 \vee a2) \wedge (a1 \vee a3) \wedge (a2 \vee a3) \wedge (a1 \vee a2 \vee a3)$$

Continuando a simplificar, aplicando a lei da absorção, tem-se:

$$F(A) = (a2) \wedge (a1 \vee a3)$$

Finalmente, através da propriedade distributiva da conjunção em relação à disjunção, fica:

$$F(A) = (a1 \wedge a2) \vee (a2 \wedge a3)$$

Da expressão de $F(A)$ conclui-se que os atributos $a1$ e $a2$ formam um reduto de A, tal como os atributos $a2$ e $a3$.

3.1.6. Características dos *Rough Sets*

Desde que foi proposta, a teoria dos *Rough Sets* tem tido um grande desenvolvimento e aplicação. Para tal contribuíram o facto de ter uma estrutura matemática maturada, que não necessita de grandes bases teóricas. Tem um modelo simples, fácil de compreender e aplicar. Pode processar dados inconsistentes, incompletos, vagos e de vários tipos. A teoria dos *Rough Sets* permite obter regras de decisão simplificadas e precisas (ZHANG *et al.*, 2016) e uma notável habilidade para lidar com dados qualitativos (RISSINO, LAMBERT-TORRES, 2009). Além disto, o método *Rough Sets* não precisa de informação preliminar sobre os dados, é objetivo a lidar com as inconsistências, permite avaliar a significância dos dados e obter um conjunto de regras de decisão (PAWLAK, 2003).

A teoria dos *Rough Sets* pode, facilmente, ser aplicada como complemento a outros métodos (ZHANG *et al.*, 2016) e tem aplicação na maioria dos casos reais (RISSINO, LAMBERT-TORRES, 2009). Os recentes desenvolvimentos da teoria têm passado pelo estudo da significância dos atributos, pela incorporação de algoritmos heurísticos e integração com o processamento paralelo. Um senão da teoria é estar limitada a dados discretos, sendo este um tópico para desenvolvimentos futuros (ZHANG *et al.*, 2016).

3.2. Metodologia LAD

No final da década de 1980, Hammer e a sua equipa deram a conhecer uma nova metodologia de seleção e classificação de dados, chamada Análise Lógica de Dados (LAD) (BRUNI, 2007). O seu suporte teórico é a matemática discreta, com destaque para a teoria das funções booleanas (BOROS *et al.*, 1997), razão pela qual, a LAD só trabalha com variáveis binárias. No entanto, raras são as bases de dados reais onde todos os atributos são binários. Quando tal não acontece, ou seja, existem atributos cujo domínio corresponde a valores qualitativos (variáveis literais, como a cor), ou quantitativos (variáveis discretas ou contínuas), há que aplicar um processo de conversão para valores binários (BOROS *et al.*, 1997). Nesta secção, a LAD vai ser analisada com o foco na seleção de atributos.

Recuperando as definições usadas na descrição dos *Rough Sets*, vem:

- Sistema de informação – conjunto das observações, com os atributos e respetivos valores.
- Sistema de decisão – sistema de informação com a classe associada.
- U – Universo – conjunto das observações (o_1, o_2, …, o_n), onde n é o número total de observações.
- A – conjunto dos atributos (a_1, a_2, …, a_m), onde m é o número total de atributos. $A = \{a_1, a_2, \ldots, a_m\}$
- C – classe.

Considere-se ainda:

- D_i – o domínio de cada atributo – conjuntos de todos os valores do atributo respetivo.

Relativamente à ação de converter os valores dos atributos para valores binários, ir-se-á usar a expressão *binarizar* e *binarização* para designar o ato de *binarizar*.

Dado um sistema de decisão, onde a classe é binária, podem-se definir dois conjuntos de observações, que formam uma partição do universo de observações (BOROS *et al.*, 1996):

- Conjunto de observações positivas (U^+), cujas observações (o^+) correspondem a um dos dois valores da classe;

- Conjunto de observações negativas (U^-), cujas observações (o^-) não correspondem ao valor da classe escolhido para definir U^+, ou seja, correspondem ao outro valor da classe;

A metodologia LAD não lida com inconsistências, logo, os conjuntos U^+ e U^- são bem definidos, sem ambiguidades (BOROS *et al.*, 1996). Outra particularidade é o facto de a LAD ser aplicada a sistemas de decisão com uma única classe, que só poderá ter dois valores possíveis, facilmente substituíveis pelos valores binários 0 e 1. No entanto, a LAD pode ser estendida a problemas onde o atributo classe tem mais de dois valores possíveis (SUBASI, AVILA-HERRERA, 2015).

3.2.1. *Binarização* de um sistema de decisão

Para que se possa aplicar a LAD, todos os atributos devem ser binários (BOROS *et al.*, 1996). Caso não sejam, há que *binarizá-los*. No processo de *binarização*, cada atributo não binário, a_i, será convertido num conjunto de atributos binários, b_j, onde $j = 1, \ldots, m^*$, sendo m^* a soma do número de atributos a_i iniciais que já eram binários, com o número total de atributos binários necessários para substituir os atributos a_i não binários (BRUNI, 2007).

Num sistema de decisão *binarizado*, o correspondente conjunto de atributos binários b_j, será $B = \{b_1; \ldots ; b_{m^*}\}$. As observações corresponderão a tuplos de valores binários, $o_b = (b_1; \ldots ; b_{m^*})$ (BRUNI, 2007).

O conjunto de atributos binários (B), resultantes da *binarização* de um sistema de decisão, é chamado de conjunto suporte (*support set*). Um conjunto suporte diz-se exatamente separado se cada tuplo o_b corresponder, sem ambiguidades, a uma observação positiva ou negativa. Isto é, se não existirem duas observações, uma positiva e a outra negativa, com iguais valores *binarizados* (BRUNI, 2007). Neste caso, não ocorrem inconsistências. Como a LAD não trabalha com dados inconsistentes, considerar-se-ão, sempre, conjuntos de suporte exatamente separados.

Quando um atributo a_i é qualitativo nominal, serão necessários tantos atributos binários (n_i) quanto o número total de elementos do, respetivo, domínio D_i (BOROS *et al.*, 1996). Supondo o atributo $a1$ (cor), com D_1 = {amarelo, verde, vermelho, azul}, vem $n_1 = \#D_1 = 4$. Desta forma, o atributo $a1$ será desdobrado em quatro atributos binários (b_j, j=1,…,4) correspondentes a cada uma das cores. O valor do atributo binário (b_j, j=1,…,4) para determinada observação o_b, será 1 se corresponder à cor do atributo respetivo e será 0 para as restantes cores.

Tabela 3.9 – Exemplo de um sistema de informação com um atributo qualitativo.

Observações	$a1$
$o1$	amarelo
$o2$	verde
$o3$	vermelho
$o4$	verde
$o5$	azul
$o6$	vermelho

Tabela 3.10 – Resultado da binarização do atributo qualitativo.

Observações	b_1 (amarelo)	b_2 (verde)	b_3 (vermelho)	b_4 (azul)
$o1$	1	0	0	0
$o2$	0	1	0	0
$o3$	0	0	1	0
$o4$	0	1	0	0
$o5$	0	0	0	1
$o6$	0	0	1	0

Também, se podem *binarizar* atributos qualitativos ordinais, como por exemplo, meses do ano, ou ocorrência de um fenómeno (nunca, raramente, frequentemente, sempre) (BOROS *et al.*, 1996). Neste caso, deve-se criar uma correspondência entre os valores literais ordenados e uma sequência de números naturais, após a qual se procede à *binarização* do atributo quantitativo resultante, da forma descrita a seguir.

No caso de um atributo quantitativo, o processo de *binarização* é mais trabalhoso. Há que introduzir a noção de ponto de corte (*cut-point*), $\alpha_{i,j} \in I\!R$ (BOROS *et al.*, 1996).

Dado um atributo quantitativo, não binário, a_i, considere-se o respetivo domínio finito, D_i, cujos elementos estão ordenados por ordem crescente. Determina-se um ponto de corte entre dois valores consecutivos de D_i, sempre que um pertença a uma observação positiva, $o^+ \in U^+$ e o outro pertença a uma observação negativa, $o^- \in U^-$.

Sejam $a_i(k)$ e $a_i(k+1)$ dois valores consecutivos do domínio de um atributo, tais que $a_i(k) < a_i(k+1)$, $a_i(k) \in o^+$ e $a_i(k+1) \in o^-$. Neste caso, define-se um ponto de corte, cujo valor será a média destes dois valores de D_i (BRUNI, 2007):

$$\alpha_{i,j} = \frac{a_i(k) + a_i(k+1)}{2} \tag{3.14}$$

A cada ponto de corte do atributo a_i ($\alpha_{i,j}$) vai corresponder um atributo binário (b_j), cujo valor dependerá do valor do atributo (a_i):

$$b_j = \begin{cases} 1 & \text{se } a_i \geq \alpha_{i,j} \\ 0 & \text{se } a_i < \alpha_{i,j} \end{cases}$$

Da mesma forma se define um ponto de corte entre dois valores consecutivos do domínio do atributo, onde o primeiro pertence a uma observação negativa e o segundo a uma observação positiva. Os valores dos pontos de corte não têm de pertencer ao domínio do atributo.

Estes são os atributos binários do primeiro tipo, chamados de atributos de nível. Também são definidos os atributos binários do segundo tipo, designados por atributos de intervalo. (BOROS *et al.*, 1996)

A cada par de pontos de corte do atributo a_i ($\alpha_{i,j}$ e $\alpha_{i,k}$, com $\alpha_{i,j} < \alpha_{i,k}$) vai corresponder um atributo binário ($b_{i,jk}$), cujo valor dependerá do valor do atributo (a_i):

$$b_{i,jk} = \begin{cases} 1 & \text{se } \alpha_{i,j} \leq a_i < \alpha_{i,k} \\ 0 & \text{para outros valores de } a_i \end{cases}$$

Tabela 3.11 – Exemplo de um sistema de decisão com dois atributos quantitativos.

Observações	*a*2	*a*3	classe
*o*1	6	45	Sim
*o*2	8	60	Não
*o*3	4	35	Não
*o*4	5	70	Sim
*o*5	6	50	Sim
*o*6	3	65	Não

Considerando as observações com valor de classe "Sim" como positivas, da Tabela 3.11 vem: $U^{+} = \{o1, o4, o5\}$ e $U^{-} = \{o2, o3, o6\}$. Na Figura 3.2, os sinais ✚ e ▭ identificam, respetivamente, as observações positivas e as observações negativas.

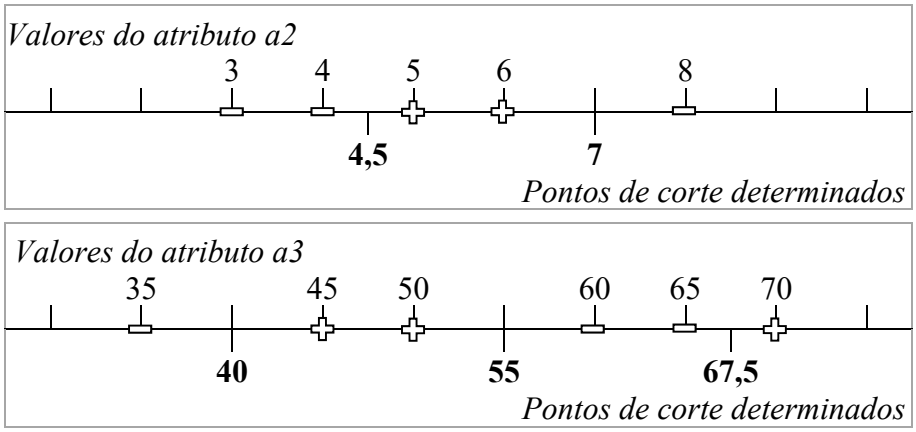

Figura 3.2 – Pontos de corte determinados a partir do exemplo da Tabela 3.11.

(Adaptado de: BRUNI, 2007)

Para o atributo *a*2 foram determinados dois pontos de corte: $\alpha_{2,1} = 4{,}5$; $\alpha_{2,2} = 7$. A estes pontos de corte corresponderão os atributos binários de nível b_5 e b_6 , cujos valores serão, respetivamente:

$$b_5 = \begin{cases} 1 \text{ se } a2 \geq 4{,}5 \\ 0 \text{ se } a2 < 4{,}5 \end{cases}; \qquad b_6 = \begin{cases} 1 \text{ se } a2 \geq 7 \\ 0 \text{ se } a2 < 7 \end{cases}.$$

e o atributo binário de intervalo b_7 , cujo valor será: $b_7 = \begin{cases} 1 \text{ se } 4{,}5 \leq a2 < 7 \\ 0 \text{ nos outros casos} \end{cases}$

Para o atributo $a3$ foram determinados três pontos de corte: $\alpha_{3,1} = 40$; $\alpha_{3,2} = 55$; $\alpha_{3,3} = 67{,}5$. A estes pontos de corte corresponderão os atributos binários de nível b_8, b_9 e b_{10}, cujos valores serão, respetivamente:

$$b_8 = \begin{cases} 1 \text{ se } a3 \geq 40 \\ 0 \text{ se } a3 < 40 \end{cases}; \qquad b_9 = \begin{cases} 1 \text{ se } a3 \geq 55 \\ 0 \text{ se } a3 < 55 \end{cases}; \qquad b_{10} = \begin{cases} 1 \text{ se } a3 \geq 67{,}5 \\ 0 \text{ se } a3 < 67{,}5 \end{cases}.$$

e os atributos binários de intervalo b_{11}, b_{12}, b_{13}, cujos valores serão, respetivamente:

$$b_{11} = \begin{cases} 1 \text{ se } 40 \leq a3 < 55 \\ 0 \text{ nos outros casos} \end{cases}; \quad b_{12} = \begin{cases} 1 \text{ se } 40 \leq a3 < 67{,}5 \\ 0 \text{ nos outros casos} \end{cases};$$

$$b_{13} = \begin{cases} 1 \text{ se } 55 \leq a3 < 67{,}5 \\ 0 \text{ nos outros casos} \end{cases}.$$

Quanto ao atributo classe, facilmente se atribui o valor 1 a "Sim" e o valor 0 a "Não".

Tabela 3.12 – Resultado da binarização de dois atributos quantitativos.

Observações	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	classe
$o1$	1	0	1	1	0	0	1	1	0	1
$o2$	1	1	0	1	1	0	0	1	1	0
$o3$	0	0	0	0	0	0	0	0	0	0
$o4$	1	0	1	1	1	1	0	0	0	1
$o5$	1	0	1	1	0	0	1	1	0	1
$o6$	0	0	0	1	1	0	0	1	1	0

O processo de *binarização* é analisado com grande detalhe no trabalho de Boros *et al.* (1997).

3.2.2. Redução do conjunto suporte

No processo de *binarização* de sistemas de decisão, obtidos a partir de casos reais de grande dimensão, normalmente, são determinados um grande número de pontos de corte, tornando impraticável o seu tratamento informático (BOROS *et al.*, 1996). Acontece que, muitos dos atributos binários não são necessários para classificar as observações. Daqui resulta a possibilidade de reduzir a dimensão do conjunto suporte com o objetivo de diminuir a complexidade do problema e melhorar a performance computacional. (BRUNI, 2007)

Para cada par de observações, uma positiva (o^+) e outra negativa (o^-), seja $I(o^+,o^-)$ o conjunto de pares ordenados (r, s) que correspondem aos índices do par de observações o^+ e o^- com representações binárias diferentes (BRUNI, 2007).

Considere-se, para cada atributo binário b_j, uma nova variável binária y_j, definida da seguinte forma (BRUNI, 2007):

$$y_j = \begin{cases} 1 & \text{se } b_j \text{ se mantém no conjunto suporte após a redução} \\ 0 & \text{se } b_j \text{ não se mantém no conjunto suporte após a redução} \end{cases}$$

Há que definir, também, um coeficiente binário, $d_{i,j}$, que indica se os valores binários do atributo b_j são diferentes, para cada par (r, s) $\in$ $I(o^+,o^-)$, identificado pelo índice i (CAVIQUE *et al.*, 2011):

$$d_{i,j} = \begin{cases} 1 & \text{se } b_j(r) \neq b_j(s) \\ 0 & \text{se } b_j(r) = b_j(s) \end{cases}$$

onde, $b_j(r)$ é o valor binário do atributo b_j na observação positiva o_r e $b_j(s)$ é o valor binário do atributo b_j na observação negativa o_s.

Ou seja, $d_{i,j} = 1$ se, para duas observações o_r e o_s, os valores binários do atributo b_j são diferentes. Caso contrário, $d_{i,j} = 0$. O conjunto de todos os $d_{i,j}$ formam a matriz disjunta M.

O processo de redução do conjunto suporte pode ser matematizado através do modelo de cobertura de conjuntos (BOROS *et al.*, 1996):

$$\min \sum_{j=1}^{m^*} y_j$$

Sujeito às condições:

$$\sum_{j=1}^{m^*} d_{i,j}.y_j \geq 1 \ , \ \forall\, i \in I(o^+,o^-) \text{, com } o^+ \in U^+ , o^- \in U^-$$

$$y_j \in \{0,1\} \quad \text{, com } j = 1, \ldots , m^*$$

Os atributos binários selecionados serão os b_j que na solução ótima correspondem aos valores de $y_j = 1$.

Considere-se o sistema de decisão composto pelo conjunto dos atributos usados atrás.

Tabela 3.13 – Sistema de decisão usado como exemplo.

Observações	$a1$	$a2$	$a3$	classe
$o1$	amarelo	6	45	1
$o2$	verde	8	60	0
$o3$	vermelho	4	35	0
$o4$	verde	5	70	1
$o5$	azul	6	50	1
$o6$	vermelho	3	65	0

Pelo processo de *binarização*, obtém-se o seguinte sistema de decisão *binarizado*:

Tabela 3.14 – Binarização do sistema de decisão usado como exemplo.

Obs.	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	c
$o1$	1	0	0	0	1	0	1	1	0	0	1	1	0	1
$o2$	0	1	0	0	1	1	0	1	1	0	0	1	1	0
$o3$	0	0	1	0	0	0	0	0	0	0	0	0	0	0
$o4$	0	1	0	0	1	0	1	1	1	1	0	0	0	1
$o5$	0	0	0	1	1	0	1	1	0	0	1	1	0	1
$o6$	0	0	1	0	0	0	0	1	1	0	0	1	1	0

Caso existam observações redundantes, devem ser retiradas uma a uma de forma a eliminar essas redundâncias.

A partir do sistema de decisão *binarizado*, há que determinar as condições às quais os valores de y_j estão sujeitos:

Tabela 3.15 – Determinação das condições da formulação do problema exemplo.

i	(o^+, o^-)	$d_{i,1}$	$d_{i,2}$	$d_{i,3}$	$d_{i,4}$	$d_{i,5}$	$d_{i,6}$	$d_{i,7}$	$d_{i,8}$	$d_{i,9}$	$d_{i,10}$	$d_{i,11}$	$d_{i,12}$	$d_{i,13}$	condição
1	($o1$,$o2$)	1	1	0	0	0	1	1	0	1	0	1	0	1	$y_1+y_2+y_6+y_7+y_9+y_{11}+y_{13} \geq 1$
2	($o1$,$o3$)	1	0	1	0	1	0	1	1	0	0	1	1	0	$y_1+y_3+ y_5+ y_7+y_8+y_{11}+y_{12} \geq 1$
3	($o1$,$o6$)	1	0	1	0	1	0	1	0	1	0	1	0	1	$y_1+y_3+y_5+y_7+y_9+y_{11}+y_{13} \geq 1$
4	($o4$,$o2$)	0	0	0	0	0	1	1	0	0	1	0	1	1	$y_6+y_7+y_{10}+y_{12}+y_{13} \geq 1$
5	($o4$,$o3$)	0	1	1	0	1	0	1	1	1	1	0	0	0	$y_2+y_3+y_5+y_7+y_8+y_9+y_{10} \geq 1$
6	($o4$,$o6$)	0	1	1	0	1	0	1	0	0	1	0	1	1	$y_2+y_3+y_5+y_7+y_{10}+y_{12}+y_{13} \geq 1$
7	($o5$,$o2$)	0	1	0	1	0	1	1	0	1	0	1	0	1	$y_2+y_4+y_6+y_7+y_9+y_{11}+y_{13} \geq 1$
8	($o5$,$o3$)	0	0	1	1	1	0	1	1	0	0	1	1	0	$y_3+y_4+y_5+y_7+y_8+y_{11}+y_{12} \geq 1$
9	($o5$,$o6$)	0	0	1	1	1	0	1	0	1	0	1	0	1	$y_3+y_4+y_5+y_7+y_9+y_{11}+y_{13} \geq 1$

A redução do conjunto suporte é obtida a partir da solução do seguinte problema de otimização:

$$\min (y_1+y_2+y_3+y_4+y_5+y_6+y_7+y_8+y_9+y_{10}+y_{11}+y_{12}+y_{13})$$

Sujeito a:

$$y_1+y_2+y_6+y_7+y_9+y_{11}+y_{13} \geq 1$$
$$y_1+y_3+ y_5+ y_7+y_8+y_{11}+y_{12} \geq 1$$
$$y_1+y_3+y_5+y_7+y_9+y_{11}+y_{13} \geq 1$$
$$y_6+y_7+y_{10}+y_{12}+y_{13} \geq 1$$
$$y_2+y_3+y_5+y_7+y_8+y_9+y_{10} \geq 1$$
$$y_2+y_3+y_5+y_7+y_{10}+y_{12}+y_{13} \geq 1$$
$$y_2+y_4+y_6+y_7+y_9+y_{11}+y_{13} \geq 1$$
$$y_3+y_4+y_5+y_7+y_8+y_{11}+y_{12} \geq 1$$
$$y_3+y_4+y_5+y_7+y_9+y_{11}+y_{13} \geq 1$$

$$y_j \in \{0,1\} \text{ , com } j=1,\ldots,13$$

Através de métodos matemáticos, ou recorrendo a algoritmos de resolução de problemas de programação linear, vem:

$$(y_1;y_2;y_3;y_4;y_5;y_6;y_7;y_8;y_9;y_{10};y_{11};y_{12};y_{13}) = (0;0;0;0;0;0;1;0;0;0;0;0;0)$$

Assim sendo, obtém-se uma solução, que corresponde a um conjunto suporte reduzido, $B_1 = \{b_7\}$. O resultado da seleção de atributos, correspondente a B_1, indica que o sistema de decisão pode ser reduzido ao conhecimento do atributo $a2$ (dependendo se o seu valor está entre 4,5 e 7, ou não), conforme indicado na Tabela 3.16.

Tabela 3.16 – Redução do sistema de decisão do exemplo.

Observações	b_7	classe
$o1$	1	1
$o2$	0	0
$o3$	0	0
$o4$	1	1
$o5$	1	1
$o6$	0	0

Agrupando as observações positivas e as negativas, vem:

Tabela 3.17 – Redução do sistema de decisão agrupado por U^+ e U^-.

		B_1
	Observações	b_7
U^+	$o1$	1
	$o4$	1
	$o5$	1
U^-	$o2$	0
	$o3$	0
	$o6$	0

Após a redução do conjunto suporte, a metodologia LAD cria padrões que serão usados no processo de classificação. Este aspeto não será, aqui, desenvolvido, por estar fora do âmbito do tema.

3.2.3. Variantes à metodologia clássica da LAD

Há situações às quais se pode aplicar o algoritmo LAD, embora os respetivos problemas tenham variantes relativamente ao modelo clássico. É o caso da existência de custos associados aos atributos, ou existirem atributos sem valor em algumas observações, ou, ainda, classes com multivalores.

Se a cada atributo binário for imputado um custo, q_j, a formalização do problema de redução do conjunto suporte vem (CAVIQUE *et al.*, 2011):

$$\min \sum_{j=1}^{m^*} q_j \, . y_j$$

Sujeito às condições:

$$\sum_{j=1}^{m^*} d_{i,j} . y_j \quad \geq 1 \quad , \quad \forall\, i \in I(o^+, o^-) \,, \text{com } o^+ \in U^+ \,, o^- \in U^-$$

$$y_j \in \{0,1\} \quad , \text{ com } j = 1, \ldots, m^*$$

A imputação de um custo a cada atributo binário quantifica a sua importância isolada para a solução final. Não tem em conta atributos que, por si só, não são significativos, mas, quando combinados, têm uma importância apreciável para a qualidade da redução do conjunto suporte (BRUNI, 2007). O recurso aos custos na formulação do problema apresenta grandes vantagens para a sua resolução computacional, nomeadamente, ao nível do tempo de execução dos algoritmos. Esta vantagem é mais notória quando o algoritmo procura uma solução através de um processo heurístico (BOROS *et al.*, 1996).

Outra variante ocorre quando existem atributos com valor omisso em algumas observações. Se, num atributo original, a_i, existem valores em falta, então nos respetivos atributos binários, b_j, também, são considerados em falta. Isto faz com que estes atributos não sejam usados. Matematicamente, corresponde a atribuir o valor zero aos respetivos coeficientes $d_{i,j}$ (BOROS *et al.*, 1996).

No caso do atributo classe de um problema ter mais de dois valores possíveis, procede-se à *binarização* da classe, como se de outro atributo se tratasse (SUBASI, AVILA-HERRERA, 2015). Este aspeto tem sido bastante estudado e existem diversas abordagens para lidar com o problema (FRIEDMAN *et al.*, 2000).

3.2.4. LAD versus *Rough Sets*

Tanto a LAD como a metodologia *Rough Sets* permitem efetuar a seleção de atributos e, ambos, são compostos por duas etapas: uma primeira etapa de transformação e uma segunda de redução do número de atributos (CAVIQUE *et al.*, 2011).

A LAD não permite a existência de inconsistências (duas observações com os mesmos valores dos atributos, mas pertencentes a classes diferentes) e só pode ser aplicada a dados binários (podendo recorrer ao processo de *binarização*) (BOROS *et al.*, 1996). Outro aspeto favorável é poder associar a cada atributo um custo, o que permite selecionar os atributos minimizando o custo total. Tudo isto faz da LAD uma metodologia mais sistematizada, robusta e sem ambiguidades, sendo fácil de interpretar (CAVIQUE *et al.*, 2013).

Já os *Rough Sets* lida com inconsistências, embora possa dificultar a interpretação dos resultados. Também, possibilita a existência de atributos classe com valores não necessariamente dicotomizados. No entanto, não permite o recurso aos custos associados a cada atributo (CAVIQUE *et al.*, 2011).

Da combinação destas duas metodologias, surge a Análise Lógica de Dados Inconsistentes (LAID), que é descrita na secção seguinte.

3.3. Metodologia LAID

Como já referido, a Análise Lógica de Dados Inconsistentes (LAID) surge da combinação das metodologias *Rough Sets* e LAD. Como características principais, a LAID permite atributos inteiros, com custos associados, lida com inconsistências e comporta classes não dicotomizadas (CAVIQUE *et al.*, 2013).

3.3.1. Remoção das redundâncias

Observações redundantes são observações com os mesmos valores dos atributos e pertencentes à mesma classe. Sendo m o número de atributos do sistema de decisão, o_x e o_y são redundantes se:

$$\begin{cases} a_i(o_x) = a_i(o_y) \ \ \forall i = 1, \dots, m \\ c(o_x) = c(o_y) \end{cases}$$

A remoção de redundâncias é simples, basta eliminar, uma a uma, as observações redundantes.

3.3.2. Remoção das inconsistências

Observações inconsistentes são observações com os mesmos valores dos atributos e pertencentes a classes diferentes. Sendo m o número de atributos do sistema de decisão, o_x e o_y são inconsistentes se:

$$\begin{cases} a_i(o_x) = a_i(o_y) \ \ \forall i = 1, \dots, m \\ c(o_x) \neq c(o_y) \end{cases}$$

Considere-se o grau de inconsistência (gr) de uma observação o_x, com $x = 1,\dots,n$, como o número de observações do Universo de dimensão n, inconsistentes com o_x. A definição formal é:

$$gr(o_x) = \#\{y=1,\dots,n: o_y \in U \wedge (a_i(o_x) = a_i(o_y), \forall i=1,\dots,m) \wedge \ c(o_x) \neq c(o_y)\}$$

Se $gr(o_x)$=0, significa que a observação o_x não tem inconsistências.

O grau de inconsistência do Universo, $gr(U)$ é igual ao maior dos graus de inconsistência de cada observação:

$$gr(U) = \text{máx}(\ gr(o_x)),\ \forall x=1,\ldots,n$$

O valor do grau de inconsistência do Universo é, sempre, menor ou igual ao número de valores diferentes da classe. Se $gr(U)$=0, significa que não existem inconsistências no sistema de decisão.

O processo de remoção das inconsistências existentes num sistema de decisão é surpreendentemente simples. O exemplo dado por Cavique *et al.* (2013) é bastante esclarecedor para se perceber a ideia subjacente. Quando um paciente vai a uma consulta, pode acontecer que o médico fique indeciso no diagnóstico, porque o conjunto dos sintomas relatados são comuns a várias doenças. Neste caso, o médico faz um ou mais testes para despistar o mal de que o paciente padece. Considerando os sintomas como um conjunto de atributos, o que o clínico faz é acrescentar novos atributos de forma a acabar com as ambiguidades existentes.

Assim sendo, quando existem inconsistências num conjunto de dados, elas são removidas através da adição de novos atributos binários, que vão testar "*je ne sais quoi*", para diferenciar as observações inconsistentes (CAVIQUE *et al.*, 2013). Para remover as inconsistências de um sistema de decisão são necessários $m^* = \lceil log_2(1+gr(U)) \rceil$ novos atributos binários, onde a função $\lceil x \rceil$ devolve a aproximação às unidades por excesso do valor x. Ou seja, é necessário um novo atributo binário, para distinguir duas observações com os mesmos valores dos atributos, mas pertencentes a classes diferentes. No caso de três, ou quatro, observações com valores dos atributos iguais, mas classes diferentes, são necessários 2 novos atributos binários.

Pelo facto de acrescentar novos atributos "*je ne sais quoi*" para retirar as inconsistências, faz com que qualquer subconjunto de U seja preciso (*crisp*), sendo a sua região fronteira vazia, porque $\underline{A}(X) = \bar{A}(X)$. Desta forma, evitam-se as dificuldades de interpretação destes conceitos, que são fundamentais na teoria dos *Rough Sets* (YAO, 2006).

A Figura 3.3 esquematiza a diferença entre as metodologias *Rough Sets* e LAID no tratamento das inconsistências. Neste exemplo, o Universo é composto por 6 observações (o_1, o_2, o_3, o_4, o_5, o_6). As observações o_1, o_4 e o_5 têm valor da classe igual a 1, as restantes observações têm valor da classe 0. As observações o_5 e o_6 são inconsistentes, ou seja, têm igual valor nos atributos, mas classe diferente. Seja X um subconjunto de U cujos valores dos atributos correspondem ao valor 1 da classe. Logo, $\underline{A}(X) = \{o_1; o_4\}$, $\bar{A}(X) = \{o_1; o_4; o_5; o_6\}$ e $FR(X) = \{o_5; o_6\}$. Como cada inconsistência corresponde a um par de observações, é necessário um novo atributo "*je ne sais quoi*" (a^*) que toma o valor 1 para a observação da fronteira cujo valor da classe é 1 (o_5) e o valor 0 para as restantes observações. Assim, após a inserção do atributo a^*, o subconjunto de U, cujos valores dos atributos correspondem ao valor 1 da classe, será X^*, onde $X^* = \underline{A}(X^*) = \bar{A}(X^*) = \{o_1; o_4; o_5\}$, e $FR(X^*) = \varnothing$. A Figura 3.3 mostra o resultado da remoção das inconsistências na LAID quando é adicionado o atributo *jnsq*.

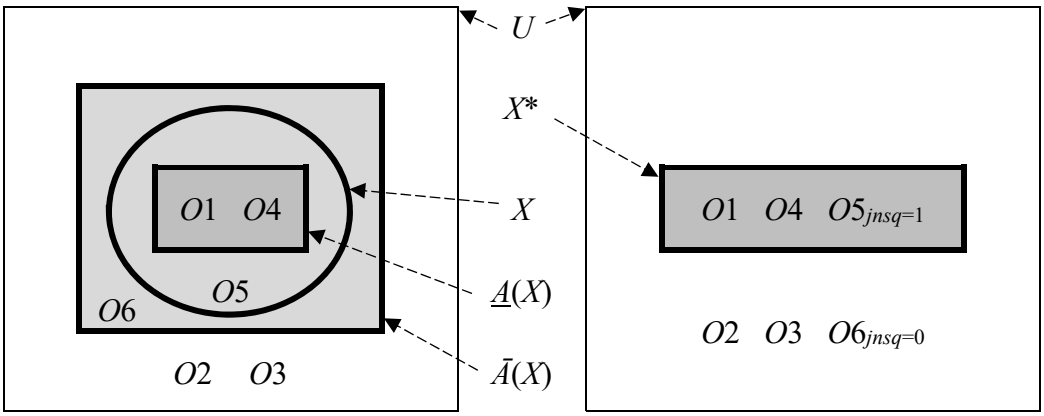

Figura 3.3 – Diferença entre os Rough Sets e a remoção das inconsistências na LAID. (Adaptado de: PAWLAK, 2003 e CAVIQUE et al., 2013)

Sistematizando todo o processo de remoção das inconsistências num sistema de decisão:

- Há que começar por remover as redundâncias.
- A quantidade de atributos "*je ne sais quoi*" é determinada por $m^* = \lceil log_2(1+gr(U)) \rceil$.
- Para cada conjunto de inconsistências, ordenam-se as observações pelos respetivos valores da classe e faz-se corresponder, a cada uma, a respetiva posição de ordem, $p_i(c(o_x))$, começando por zero.

- Para cada observação inconsistente, o valor de cada atributo "*je ne sais quoi*" é o respetivo algarismo da codificação binária de $p_i(c(o_x))$. Para as observações sem inconsistências, todos os atributos "*je ne sais quoi*" têm o valor zero.

O conjunto de todos os atributos originais, unido com o conjunto dos atributos "*je ne sais quoi*" é designado por A^*.

C*o*nsidere-se o exemplo do sistema de decisão da Tabela 3.18 para aplicar a remoção das inconsistências segundo a metodologia LAID.

Tabela 3.18 – Sistema de decisão para exemplificar a LAID.

Observações	*a*1	*a*2	*a*3	*a*4	classe
*o*1	1	0	1	0	1
*o*2	0	1	1	0	1
*o*3	0	1	0	0	1
*o*4	0	1	0	0	2
*o*5	1	0	0	1	0
*o*6	1	0	1	0	2
*o*7	0	1	0	0	0
*o*8	1	0	1	0	1

Começa-se por remover as redundâncias. As observações *o*1 e *o*8 são redundantes. Ao retirar uma delas, escolha-se *o*8, a redundância desaparece.

Tabela 3.19 – Sistema de decisão com a redundância removida.

Observações	*a*1	*a*2	*a*3	*a*4	classe
*o*1	1	0	1	0	1
*o*2	0	1	1	0	1
*o*3	0	1	0	0	1
*o*4	0	1	0	0	2
*o*5	1	0	0	1	0
*o*6	1	0	1	0	2
*o*7	0	1	0	0	0

Da observação da Tabela 3.19, identificam-se os seguintes conjuntos de observações inconsistentes: {$o1$, $o6$} e {$o3$, $o4$, $o7$}, vindo:

Tabela 3.20 – Grau de inconsistência das observações.

Observações	$o1$	$o2$	$o3$	$o4$	$o5$	$o6$	$o7$
gr(o_x)	1	0	2	2	0	1	2

Como o grau de inconsistência do universo de observações é $gr(U) = 2$, são necessários dois atributos "*je ne sais quoi*" ($a5^*$ e $a6^*$), porque $m^* = \lceil log_2(1+2) \rceil = 2$, ficando $A^*=\{a1, a2, a3, a4, a5^*, a6^*\}$.

Tabela 3.21 – Remover as inconsistências do sistema de decisão.

Observações	$a1$	$a2$	$a3$	$a4$	$a5^*$	$a6^*$	classe
$o1$	1	0	1	0	0	0	1
$o2$	0	1	1	0	0	0	1
$o3$	0	1	0	0	1	0	1
$o4$	0	1	0	0	0	1	2
$o5$	1	0	0	1	0	0	0
$o6$	1	0	1	0	1	0	2
$o7$	0	1	0	0	0	0	0

O sistema da Tabela 3.21 está pronto para aplicar o algoritmo de seleção dos atributos. Sendo a LAID um algoritmo constituído por duas fases (CAVIQUE *et al.*, 2013):

1. Gerar a matriz disjunta *M*;
2. Redução do conjunto suporte.

3.3.3. Gerar a matriz disjunta

À semelhança da LAD, a metodologia LAID recorre à determinação da matriz disjunta *M*. Mas, ao contrário da primeira, a LAID permite classes com um número ilimitado de valores diferentes (CAVIQUE *et al.*, 2013).

O processo consiste em, para cada observação, comparar os valores de cada atributo com os respetivos das observações seguintes que tenham valor de classe diferente. Se o_x e o_y forem duas observações, tais que $c(o_x) \neq c(o_y)$, cuja comparação corresponde à linha i da matriz disjunta M, os elementos da matriz M ($d_{i,j}$, com j=1,…,m+m*) serão (CAVIQUE *et al.*, 2013):

$$d_{i,j} = \begin{cases} 1 & \text{se } a_j(o_x) \neq a_j(o_y) \\ 0 & \text{se } a_j(o_x) = a_j(o_y) \end{cases}$$

Como as inconsistências foram removidas, cada linha da matriz M tem de ter, pelo menos, um valor não nulo. O número de linhas da matriz M é menor, ou igual, a $n(n-1)/2$, valor que corresponderia a comparar todas as n observações. (CAVIQUE *et al.*, 2013)

Retomando o sistema da Tabela 3.21, a matriz disjunta vem:

Tabela 3.22 – Matriz Disjunta M.

Por comparação das observações	*a*1	*a*2	*a*3	*a*4	*a*5*	*a*6*
*o*1 e *o*4	1	1	1	0	0	1
*o*1 e *o*5	0	0	1	1	0	0
*o*1 e *o*6	0	0	0	0	1	0
*o*1 e *o*7	1	1	1	0	0	0
*o*2 e *o*4	0	0	1	0	0	1
*o*2 e *o*5	1	1	1	1	0	0
*o*2 e *o*6	1	1	0	0	1	0
*o*2 e *o*7	0	0	1	0	0	0
*o*3 e *o*4	0	0	0	0	1	1
*o*3 e *o*5	1	1	0	1	1	0
*o*3 e *o*6	1	1	1	0	0	0
*o*3 e *o*7	0	0	0	0	1	0
*o*4 e *o*5	1	1	0	1	0	1
*o*4 e *o*7	0	0	0	0	0	1
*o*5 e *o*6	0	0	1	1	1	0
*o*6 e *o*7	1	1	1	0	1	0

Na matriz disjunta M, se $d_{i,j} = 1$ então o atributo da coluna j diferencia as duas observações comparadas na linha i. Pelo contrário, se $d_{i,j} = 0$ então o atributo da coluna j não distingue as duas observações comparadas na linha i. (CAVIQUE *et al.*, 2013)

3.3.4. Redução do conjunto suporte

A obtenção do menor conjunto suporte usa uma heurística proposta por Chvatal (CAVIQUE *et al.*, 2013). A redução do conjunto suporte é um processo iterativo que pretende determinar um subconjunto, S, de A^*. Neste subconjunto S, inicialmente vazio, colocam-se os atributos escolhidos em cada iteração. No final, os elementos de S correspondem à seleção dos atributos que reduzem o conjunto suporte.

Começa-se por somar os valores de cada coluna da matriz M, obtendo-se o vetor s, onde:

$$s_j = \sum_{i=1}^{n} d_{i,j} \text{ , com } j = 1,\ldots,m+m^* \tag{3.15}$$

Tabela 3.23 – Processo de redução – iteração 1.

A^*	$a1$	$a2$	$a3$	$a4$	$a5^*$	$a6^*$
$d_{i,j}$	1	1	1	0	0	1
	0	0	1	1	0	0
	0	0	0	0	1	0
	1	1	1	0	0	0
	0	0	1	0	0	1
	1	1	1	1	0	0
	1	1	0	0	1	0
	0	0	1	0	0	0
	0	0	0	0	1	1
	1	1	0	1	1	0
	1	1	1	0	0	0
	0	0	0	0	1	0
	1	1	0	1	0	1
	0	0	0	0	0	1
	0	0	1	1	1	0
	1	1	1	0	1	0
s_j	8	8	9	5	7	5

As componentes do vetor s correspondem às imagens da função que decide qual o atributo escolhido para a solução, $s_j = s(a_j)$. O atributo pretendido, a_e, é aquele que melhor explica o sistema de decisão, ou seja, aquele que cumpre a condição:

$$s(a_e) = \text{máx}(s_j), \text{ com } j = 1,\ldots,m+m^* \tag{3.16}$$

No exemplo da Tabela 3.23, o atributo escolhido é $a3$, entrando para a solução, $S=\{a3\}$. Se vários atributos cumprem a condição (3.16), existem escolhas alternativas e escolhe-se um deles.

Após escolhido o atributo a entrar na solução, a_e, e atualizada a solução S, eliminam-se as linhas da matriz M que são explicadas por esse atributo, cujo valor $d_{i,e} = 1$, bem como a coluna do próprio atributo. Reinicia-se o processo com nova iteração, até eliminar todas as linhas da matriz M.

Tabela 3.24 – Processo de redução – iteração 2.

$a1$	$a2$	$a4$	$a5^*$	$a6^*$
0	0	0	1	0
1	1	0	1	0
0	0	0	1	1
1	1	1	1	0
0	0	0	1	0
1	1	1	0	1
0	0	0	0	1
3	3	2	5	3

Na segunda iteração do exemplo (Tabela 3.24), é escolhido o atributo $a5^*$, ficando $S = \{a3, a5^*\}$. São eliminadas as 5 primeiras linhas da matriz M e a coluna do atributo escolhido.

Tabela 3.25 – Processo de redução – iteração 3.

$a1$	$a2$	$a4$	$a6^*$
1	1	1	1
0	0	0	1
1	1	1	2

Na terceira iteração do exemplo (Tabela 3.25), é escolhido o atributo *a*6*, ficando $S = \{a3, a5^*, a6^*\}$. São eliminadas as restantes linhas da matriz *M* e a coluna do atributo escolhido. O processo está terminado. A redução obtida do conjunto suporte é $\{a3, a5^*, a6^*\}$.

Tabela 3.26 – Sistema de decisão após redução do conjunto suporte.

Observações	*a*3	*a*5*	*a*6*	classe
*o*1	1	0	0	1
*o*2	1	0	0	1
*o*3	0	1	0	1
*o*4	0	0	1	2
*o*5	0	0	0	0
*o*6	1	1	0	2
*o*7	0	0	0	0

Tabela 3.27 – Valores da classe em função dos atributos selecionados.

		(*a*5*, *a*6*)			
		(0,0)	(1,0)	(0,1)	(1,1)
*a*3	0	0	1	2	
	1	1	2		

As células em branco representam valores para novas observações que não são explicados pelo novo sistema de decisão. Este conceito é diferente da noção de inconsistência, porque é resultante da falta de observações (CAVIQUE *et al.*, 2013). No exemplo, dado o diminuto número de observações, era de esperar a existência de muitas células inexplicadas.

Havendo, inicialmente, inconsistências no conjunto suporte e tendo-se acrescentado os atributos "*je ne sais quoi*" para eliminar essas inconsistências, é lógico que estes atributos sejam necessários para explicar o sistema de decisão. Daí, a redução do conjunto suporte contém, sempre, todos os atributos "*je ne sais quoi*".

3.4. Critérios de validação

Depois de aplicar um determinado método de seleção de atributos, há que definir critérios para aferir a sua performance e validar os resultados obtidos.

A performance dos métodos de seleção de atributos é determinada a partir da performance dos modelos de aprendizagem aplicados após a redução, usando métricas que variam consoante o método. Os recursos computacionais (memória e tempo despendido), a precisão, o rácio de atributos selecionados, são exemplos de métricas usadas. (BOLÓN-CANEDO *et al.*, 2013)

Para dificultar este processo, muitos conjuntos de dados podem ter atributos classe multivalor, dados descontextualizados ou errados, atributos irrelevantes ou repetidos, o número de observações muito pequeno comparado com a quantidade de atributos, etc.. Todas estas dificuldades podem tornar os estudos comparativos de tal forma complexos que são impraticáveis. (BOLÓN-CANEDO *et al.*, 2013)

A maioria dos estudos foca-se no problema que se pretende resolver. Muitos são realizados recorrendo a dados gerados artificialmente. Já que os resultados são conhecidos, o método de seleção de atributos pode ser avaliado independentemente do classificador usado e permite alterar as condições experimentais. (BOLÓN-CANEDO *et al.*, 2013)

3.4.1. Classificação

O processo de classificação tem por objetivo prever o valor da classe de uma nova observação, que não pertence ao sistema de decisão usado na seleção de atributos. Normalmente, para que tal se possa realizar, o conjunto de observações do sistema é dividido, aleatoriamente, em dois subconjuntos (CAVIQUE *et al.*, 2013):

- O conjunto de treino – as observações são usadas para selecionar os atributos e determinar o modelo de aprendizagem ou classificador;

- O conjunto de teste – as observações são usadas para testar o modelo de aprendizagem determinado.

Sendo n a dimensão do universo de observações, se o conjunto de treino tem n_{tr} observações, então o conjunto de teste tem n-n_{tr} observações. Esta divisão do sistema de informação tem a desvantagem de não usar todas as observações para determinar o modelo de aprendizagem, particularmente, no caso de existirem poucas observações.

Uma forma de contornar esta desvantagem é a chamada validação cruzada (*cross-validation*) (CAVIQUE *et al.*, 2013). Esta técnica consiste em dividir o sistema de dados em p partições. Uma das partições é usada para teste, as restantes partições constituem o conjunto de treino. O processo é repetido, trocando a partição de teste por uma das de treino, até que todas as partições tenham sido usadas para teste. O modelo é avaliado pela média das performances de cada teste. Um caso particular é o chamado *Leave-One-Out Cross-Validation*, onde as partições do sistema têm apenas uma observação (CAVIQUE *et al.*, 2013).

Num sistema de decisão, onde o atributo classe pode ter k valores diferentes, a forma de prever o valor da classe de uma nova observação o_x é através da função f (CAVIQUE *et al.*, 2013), onde:

$$f(o_x) = i: H(o_x,i) = \min(H(o_x,j), \text{com } j=0,\ldots,k\text{-}1) \tag{3.17}$$

A função $H(o_x,j)$ devolve a média das distâncias de Hamming entre o_x e todas as observações do sistema de decisão cujo valor da classe é j. A distância de Hamming (dH), entre dois vetores da mesma dimensão, é o número de coordenadas cujos valores, nesses vetores, são diferentes. Dito de outra forma, a distância de Hamming é o menor número de substituições necessárias para transformar um dos vetores no outro. A função f devolve o valor da classe cujo retorno de H é menor.

Considerando o sistema de decisão da Tabela 3.28, pretende-se prever o valor da classe de uma nova observação o_x = (1,0,1). A Tabela 3.29 esquematiza o processo de cálculo e determina o resultado final. O valor dH corresponde à distância de Hamming entre a nova observação e cada uma do sistema de decisão.

Tabela 3.28 – Novo exemplo de sistema de decisão.

Observações	$a1$	$a2$	$a3$	classe
$o1$	1	0	0	1
$o2$	0	1	0	1
$o3$	0	1	1	1
$o4$	0	0	1	0
$o5$	0	0	0	0
$o6$	1	1	0	0

Tabela 3.29 – Exemplo do cálculo da função f.

<table>
<tr><th>Observações</th><th>vetor</th><th>classe</th><th>dH</th><th>H(o_x,j)</th></tr>
<tr><td>o1</td><td>(1,0,0)</td><td rowspan="3">1</td><td>1</td><td rowspan="3">$\frac{1+3+2}{3} = 2$</td></tr>
<tr><td>o2</td><td>(0,1,0)</td><td>3</td></tr>
<tr><td>o3</td><td>(0,1,1)</td><td>2</td></tr>
<tr><td>o4</td><td>(0,0,1)</td><td rowspan="3">0</td><td>1</td><td rowspan="3">$\frac{1+2+2}{3} = 1{,}67$</td></tr>
<tr><td>o5</td><td>(0,0,0)</td><td>2</td></tr>
<tr><td>o6</td><td>(1,1,0)</td><td>2</td></tr>
<tr><td colspan="5">f(1,0,1) = 0</td></tr>
</table>

O valor da classe prevista para o_x = (1,0,1) é 0.

3.4.2. Avaliação da performance

A performance de um modelo de aprendizagem é avaliada através da correção, ou não, das previsões. Esta avaliação tem por base a matriz confusão (*confusion matrix*) (CAVIQUE *et al.*, 2013), cujas linhas contabilizam o número de observações de cada um dos valores das classes no conjunto de teste, de acordo com o sistema de decisão, e as colunas contabilizam o número de observações previstas para cada classe (SANTRA, CHRISTY, 2012).

Tabela 3.30 – Matriz confusão quando a classe é booleana.

		Classe prevista	
		0	1
Classe real	0	n_{00}	n_{01}
	1	n_{10}	n_{11}

Na Tabela 3.30 (SANTRA, CHRISTY, 2012):

- n_{00} é o número de observações do conjunto de teste cujo valor da classe é 0 e a previsão foi correta (verdadeiros negativos);
- n_{01} é o número de observações cujo valor da classe é 0 e a previsão foi incorreta (falsos positivos);
- n_{10} é o número de observações cujo valor da classe é 1 e a previsão foi incorreta (falsos negativos);
- n_{11} é o número de observações cujo valor da classe é 1 e a previsão foi correta (verdadeiros positivos).

A soma $n_{00} + n_{01} + n_{10} + n_{11}$ é a dimensão do conjunto de teste.

A matriz confusão permite determinar várias medidas de performance, nomeadamente, a taxa de cobertura (*overall accuracy*) (SANTRA, CHRISTY, 2012):

$$\text{taxa de cobertura} = \frac{n_{00} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} \tag{3.18}$$

Esta métrica indica a frequência de acerto do modelo obtido. Um modelo de aprendizagem é melhor que um modelo aleatório se a sua taxa de cobertura é maior que a taxa da classe modal (CAVIQUE *et al.*, 2013), ou seja, é maior que a taxa de qualquer valor da classe:

$$\left(\frac{n_{00} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} > \frac{n_{00} + n_{01}}{n_{00} + n_{01} + n_{10} + n_{11}}\right) \wedge \left(\frac{n_{00} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} > \frac{n_{10} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}}\right)$$

$$\Rightarrow \quad n_{11} > n_{01} \quad \wedge \quad n_{00} > n_{10}$$

Outra medida de performance é o *Cohen's Kappa-statistic*, que indica o grau de concordância dos dados (CAVIQUE *et al.*, 2013). Para tal, há que definir a taxa esperada (*expected rate*) que mede a precisão para um modelo de aprendizagem aleatório (JENNESS, WYNNE, 2005):

$$\text{taxa esperada} = \frac{(n_{00}+n_{01})\times(n_{00}+n_{10}) + (n_{10}+n_{11})\times(n_{01}+n_{11})}{(n_{00}+n_{01}+n_{10}+n_{11})^2} \tag{3.19}$$

O índice *Kappa-statistic* é calculado recorrendo à taxa de cobertura e à taxa esperada (JENNESS, WYNNE, 2005):

$$\textit{Kappa-statistic} = \frac{\text{taxa de cobertura} - \text{taxa esperada}}{1 - \text{taxa esperada}} \tag{3.20}$$

Os valores de *kappa* estão entre -1 e 1, inclusive, e permitem avaliar a performance de acordo com a escala seguinte (CAVIQUE *et al.*, 2013):

Tabela 3.31 – Escala de avaliação do índice Kappa-statistic.

Kappa-statistic	[-1,0; 0,0]	]0,0; 0,2]	]0,2; 0,4]	]0,4; 0,6]	]0,6; 0,8]	]0,8; 1,0]
Performance do modelo	má	fraca	sofrível	moderada	boa	excelente

O índice *Kappa-statistic* é comparável entre modelos de aprendizagem diferentes (JENNESS, WYNNE, 2005).

A matriz confusão, a taxa de cobertura e o índice *Kappa-statistic* são facilmente generalizáveis para sistemas com classes multivalor. No caso da classe poder assumir 3 valores, vem:

Tabela 3.32 – Matriz confusão quando a classe é ternária.

		Classe prevista		
		0	1	2
Classe real	0	n_{00}	n_{01}	n_{02}
	1	n_{10}	n_{11}	n_{12}
	2	n_{20}	n_{21}	n_{22}

$$\text{taxa de cobertura} = \frac{n_{00} + n_{11} + n_{22}}{n_{00} + n_{01} + n_{02} + n_{10} + n_{11} + n_{12} + n_{20} + n_{21} + n_{22}}$$

taxa esperada =

$$\frac{(n_{00}+n_{01}+n_{02})\times(n_{00}+n_{10}+n_{20})+(n_{10}+n_{11}+n_{12})\times(n_{01}+n_{11}+n_{21})+(n_{20}+n_{21}+n_{22})\times(n_{02}+n_{12}+n_{22})}{(n_{00} + n_{01} + n_{02} + n_{10} + n_{11} + n_{12} + n_{20} + n_{21} + n_{22})^2}$$

$$\textit{Kappa-statistic} = \frac{\text{taxa de cobertura} - \text{taxa esperada}}{1 - \text{taxa esperada}}$$

Referências

(BOLÓN-CANEDO *et al.*, 2013) BOLÓN-CANEDO, Verónica; SÁNCHEZ-MAROÑO, Noelia; ALONSO-BETANZOS, Amparo – A review of feature selection methods on synthetic data. In **Knowledge and Information Systems**. Springer. Vol. 34, nº. 3, março 2012. ISSN 0219-1377. P. 483-519. Disponível em WWW: <URL: https://www.researchgate.net/publication/257482053_A_review_of_feature_selection_methods_on_synthetic_data > (último acesso em 07-11-2018).

(BOROS *et al.*, 1996) BOROS, Endre; HAMMER, Peter L.; IBARAKI, Toshihide [*et al.*] – An Implementation of Logical Analysis of Data. In **IEEE Transactions on Knowledge and Data Engineering**. IEEE. Vol. 12-2, março-abril 2000. ISSN 1041-4347 (impressão). P. 292-306. Disponível em WWW: <URL: https://pdfs.semanticscholar.org/051d/a38e4d6e07ed36520e313a0fc2cf6219efa4.pdf > (último acesso em 07-11-2018).

(BOROS *et al.*, 1997) BOROS, Endre; HAMMER, Peter L.; IBARAKI, Toshihide; KOGAN, Alexander – Logical Analysis of Numerical Data. In Mathematical Programming. Springer. Vol. 79(1-3), outubro 1997. ISSN 0025-5610 (impressão), 1436-4646 (eletrónica). P. 163-190. Disponível em WWW: <URL: https://pdfs.semanticscholar.org/843a/0dc559ad1277c1f71e0460173697f5cbea01.pdf > (último acesso em 07-11-2018).

(BRUNI, 2007) BRUNI, Renato – Reformulation of the Support Set Selection Problem in the Logical Analysis of Data. In **Annals of Operations Research**. Springer. Vol. 150(1), março 2007. ISSN 0254-5330 (impressão), 1572-9338 (eletrónica). P. 79-92. Disponível em WWW: <URL: http://www.dis.uniroma1.it/~bruni/files/bruni04anor.pdf > (último acesso em 07-11-2018).

(CAVIQUE *et al.*, 2011) CAVIQUE, Luís; MENDES, Armando B.; FUNK, Matthias – Logical Analysis of Inconsistent Data (LAID) for a Paremiologic Study. In **15th Portuguese Conference on Artificial Inteligence - EPIA 2011**. Lisboa: EPIA, 2011. ISBN 978-989-95618-4-7. Disponível em WWW: <URL: https://repositorio.uac.pt/bitstream/10400.3/2905/1/2011%20EPIA%2057%20LAID%20v3%20final.pdf > (último acesso em 07-11-2018).

(CAVIQUE *et al.*, 2013) CAVIQUE, Luís; MENDES, Armando; FUNK, Matthias; SANTOS, Jorge – A feature selection approach in the study of azorean proverbs. In MASEGOSA, Antonio; VILLACORTA, Pablo; CRUZ-CORONA, Carlos; GARCÍA-CASCALES, M. Socorro; LAMATA, Maria; VERDEGAY, José – **Exploring Innovative and Successful Applications of Soft Computing**. Hershey PA: Igi Global, 2013. IBSN-13: 9781466447855. P. 38-58. Disponível em WWW: <URL: http://hdl.handle.net/10400.2/2795 > (último acesso em 07-11-2018).

(DASH, LIU, 1997) DASH, M.; LIU, H. – Feature Selection for Classification. In **Intelligent Data Analysis**. Elsevier, 1997. P. 131-156. Disponível em WWW: <URL: https://pdfs.semanticscholar.org/c7ed/d5641f979ebf5d2f7c3a3ee05c6741693205.pdf > (último acesso em 07-11-2018).

(DOAK, 1992) DOAK, Justin – **An evaluation of feature selection methods and their application to computer security**. Davis CA: Escholarship - Universidade da Califórnia, 01-01-1992. CSE 92-18. Disponível em WWW: <URL: https://escholarship.org/content/qt2jf918dh/qt2jf918dh.pdf > (último acesso em 07-11-2018).

(FRIEDMAN *et al.*, 2000) FRIEDMAN, Nir; LINIAL, Michal; NACHMAN, Iftach; PEÉR, Danna – Using Bayesian Networks to Analyze Expression Data. In **Journal of Computational Biology**. Mary Ann Liebert, Inc.. Vol. 7(3-4), agosto 2000. P. 601–620. Disponível em WWW: <URL: http://www.cs.huji.ac.il/~nir/Papers/FLNP1Full.pdf > (último acesso em 07-11-2018).

(GUYON, ELISSEEFF, 2003) GUYON, Isabelle; ELISSEEFF, André – An Introduction to Variable and Feature Selection. In **Journal of Machine Learning Research**. JMLR.Org. ISSN: 1532-4435. Vol. 3 março 2003. P. 1157-1182. Disponível em WWW: <URL: http://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf > (último acesso em 07-11-2018).

(JENNESS, WYNNE, 2005) JENNESS, Jeff; WYNNE, J. Judson – **Cohen's Kappa and Classification Table Metrics 2.0** - An ArcView 3x Extension for Accuracy Assessment of Spatially Explicit Models - U.S. Geological Survey Open-File Report Of 2005-1363. Flagstaff, AZ: U.S. Geological Survey, Southwest Biological Science Center, 2005, 91 p. Disponível em WWW: <URL: https://www.fs.fed.us/rm/pubs_other/rmrs_2005_jenness_j001.pdf > (último acesso em 07-11-2018).

(JOHN *et al.*, 1994) JOHN, George H.; KOHAVI, Ron; PFLEGER, Karl – Irrelevant Features and the Subset Selection Problem. In COHEN, William W.; HIRSH, Haym – **Machine Learning: Proceedings of the Eleventh International Conference**. São Francisco CA: Morgan Kaufmann Publishers, 1994. ISBN: 1-55860-335-2. P. 121-129. Disponível em WWW: <URL: https://pdfs.semanticscholar.org/a83b/ddb34618cc68f1014ca12eef7f537825d104.pdf > (último acesso em 07-11-2018).

(KOHAVI, JOHN, 1997) KOHAVI, Ron; JOHN, George H. – Wrappers for feature subset selection. In **Artificial Intelligence**. Elsevier. Vol. 97, dezembro 1997. P. 273-324. Disponível em WWW: <URL: https://ac.els-cdn.com/S000437029700043X/1-s2.0-S000437029700043X-main.pdf?_tid=c1932680-099d-11e8-b18e-00000aab0f02&acdnat=1517743560_36d04b415cf78fbc3ead78205910999c > (último acesso em 07-11-2018).

(KUMAR, MINZ, 2014) KUMAR, Vipin; MINZ, Sonajharia – Feature Selection: A literature Review. **Smart Computing Review**. Vol. 4, nº. 3, janeiro 2014. ISSN Nº. 2234-4624. P. 211-229. Disponível em WWW: <URL: www.smartcr.org/view/download.php?filename=smartcr_vol4no3p7.pdf > (último acesso em 07-11-2018).

(PAWLAK, 2002) PAWLAK, Zdzislaw – Rough set theory and its applications. In **Journal of Telecommunications and Information Technology**. National Institute of Telecommunications. Vol. 3, março 2002. P. 7-10. Disponível em WWW: <URL: https://www.itl.waw.pl/czasopisma/JTIT/2002/3/7.pdf > (último acesso em 07-11-2018).

(PAWLAK, 2003) PAWLAK, Zdzislaw – **Rough Sets** [brochura]. Gliwice: 2003, 51 p. Disponível em WWW: <URL: https://wiki.eecs.yorku.ca/course_archive/2010-11/W/4403/_media/roughsetsrep29.pdf > (último acesso em 07-11-2018).

(PILA, MONARD, 2001) PILA, Adriano D.; MONARD, M. Carolina – Teoria de Rough Sets Aplicada à Data Mining. In **II Congresso de Lógica Aplicada à Tecnologia, LAPTEC'2001**. São Paulo: Plêiade, 2001. P. 203-211. Disponível em WWW: <URL: http://conteudo.icmc.usp.br/pessoas/mcmonard/public/laptec2001A.pdf > (último acesso em 07-11-2018).

(RISSINO, LAMBERT-TORRES, 2009) RISSINO, Silvia; LAMBERT-TORRES, Germano – Rough Set Theory - Fundamental Concepts, Principals, Data Extraction, and Applications. In PONCE, Julio; KARAHOCA, Adem – **Data Mining and Knowledge Discovery in Real Life Applications**. Viena: InTech, 2009. ISBN 978-3-902613-53-0. P. 35-59. Disponível em WWW: <URL: http://cdn.intechopen.com/pdfs/5939/InTech-Rough_set_theory_151_fundamental_concepts_principals_data_extraction_and_applications.pdf > (último acesso em 07-11-2018).

(SANTRA, CHRISTY, 2012) SANTRA, A. K.; CHRISTY, C. Josephine – Genetic Algorithm and Confusion Matrix for Document Clustering. In **IJCSI International Journal of Computer Science Issues**. IJCSI. Vol. 9-1, nº. 2, janeiro 2012. ISSN 1694-0814 (eletrónico). P. 322-328. Disponível em WWW: <URL: http://www.ijcsi.org/papers/IJCSI-9-1-2-322-328.pdf > (último acesso em 07-11-2018).

(SASSI, 2006) SASSI, Renato J. – **Uma Arquitetura para Descoberta de Conhecimento em Bases de Dados: Teoria dos Rough Sets e Redes Neurais Artificiais**. São Paulo: Escola Politécnica da Universidade de São Paulo, 2006. 186 p. Tese de doutoramento. Disponível em WWW: <URL: http://www.teses.usp.br/teses/disponiveis/3/3142/tde-16032007-163930/publico/teseversaorevisada.pdf > (último acesso em 07-11-2018).

(SUBASI, AVILA-HERRERA, 2015) SUBASI, Munevver M.; AVILA-HERRERA, Juan F. – Logical Analysis of Multiclass Data. In **Latin American Computing Conference** (CLEI). IEEE, dezembro 2015, 10 p.. ISBN 978-1-4673-9143-6 (eletrónica). Disponível em WWW: <URL: http://isaim2016.cs.virginia.edu/papers/ISAIM2016_Boolean_Subasi_Herrera.pdf > (último acesso em 07-11-2018).

(YAO, 2006) YAO, Y. Y. – Neighborhood systems and approximate retrieval. In **Information Sciences**. Elsevier. Vol. 176-23, dezembro 2006. P. 3431-3452. Disponível em WWW: <URL: https://www.sciencedirect.com/science/article/pii/S0020025506000405 > (último acesso em 07-11-2018).

(ZHANG *et al.*, 2016) ZHANG, Qinghua; XIE, Qin; WANG, Guoyin – A survey on rough set theory and its applications. In **Transactions on Intelligence Technology**. CAAI. Vol. 1, outubro 2016. P. 323-333. Disponível em WWW: <URL: https://ac.els-cdn.com/S2468232216300786/1-s2.0-S2468232216300786-main.pdf?_tid=1c079c6f-951a-4472-ab65-5e5fa6d15246&acdnat=1528130321_973403687cf807226b867b1135761f43 > (último acesso em 07-11-2018).