Universidade do Minho
Escola de Engenharia
Departamento de Sistemas de Informação

Data Mining via Redes Neuronais Artificiais e Máquinas de Vectores de Suporte

Armando Jorge Ribeiro da Cruz

Dissertação submetida à Universidade de Minho
para obtenção do grau de Mestre em Sistemas de Informação, elaborada sob a orientação do Professor Doutor Paulo Alexandre Ribeiro Cortez
2007

“Machines take me by surprise with great frequency.”

Alan Turing

Agradecimentos

Uma dissertação é o resultado de conhecimentos e vivências adquiridas ao longo de uma vida. É resultado, também, de conselhos, exemplos e orientações de docentes e colegas. Não é uma realização cujo mérito seja de uma só pessoa.

Quero, por isso, agradecer em primeiro lugar a todo o Departamento de Sistemas de Informação da Universidade do Minho, em especial ao corpo docente do Mestrado em Sistemas de Informação, pelos conhecimentos que me transmitiram, e pelo exemplo de competência e excelência.

Agradeço também ao meu orientador, pedra angular deste trabalho, pela atenção e pela compreensão das minhas dificuldades, sem nunca descurar o empenho no cumprimento dos objectivos inicialmente propostos para a dissertação.

Por último, agradeço à minha família por toda a ajuda que me deram, pela força e por acreditarem em mim. Agradeço em particular à minha mulher, pela paciência, dedicação e amor.

Resumo

O interesse nas áreas da Descoberta de Conhecimento em Bases de Dados e *Data Mining* emergiu devido ao rápido desenvolvimento das Tecnologias de Informação e Comunicação, levando a que, hoje em dia, grandes quantidades de dados estejam armazenados em computadores. Os peritos humanos são limitados, podendo falhar na identificação de detalhes importantes. Em alternativa, podem utilizar-se ferramentas de descoberta automática com vista à extracção de conhecimento de alto nível a partir de dados em bruto. Dada esta necessidade, foram propostas diversas técnicas de *Data Mining*.

Nesta dissertação, pretende-se esclarecer quais as vantagens e capacidades de dois modelos de *Data Mining* com capacidade de aprendizagem não linear: as Redes Neuronais Artificiais (RNAs) e as Máquinas de Vectores de Suporte (MVSs). Em particular, pretende-se saber qual o desempenho destas técnicas quando aplicadas a tarefas de classificação e regressão, comparando-as com outras técnicas, i.e. Árvores de Decisão/Regressão. Assim, fez-se uma análise de ferramentas de *software* que implementam os modelos referidos, tendo-se escolhido duas aplicações de utilização livre (i.e. o ambiente de programação R e o Weka) para conduzir as experiências efectuadas. Como casos de estudo, foram utilizados diversos problemas do mundo real, retirados do repositório público UCI.

Os resultados obtidos revelam que as MVSs obtêm em geral um melhor desempenho em previsão, sendo seguidas pelas RNAs. No entanto, tal melhoria é conseguida à custa de um maior esforço computacional.

Abstract

The interest in the fields of Knowledge Discovery in Databases (KDD) and Data Mining emerged due to the rapid development of the Information and Communication Technologies, which made available vast amount of data to be stored in computers. Human experts have limitations and may fail in identifying important details. As an alternative, automatic discovery tools can be used in order to obtain high level knowledge from raw data. Considering this need, several Data Mining techniques have been proposed.

This dissertation intends to infer about the advantages of two non-linear Data Mining models: Artificial Neural Networks (ANN) and Support Vector Machines (SVM). In particular, it pretends to measure their performance when applied to classification and regression tasks, being compared with other techniques, i.e. Decision/Regression Trees. Thus, an analysis was performed over a wide range of software tools that implement the referred models. From this set, two open-source applications (i.e. the R programming environment and the Weka) where selected to conduct the experiments. Several real world problems from the UCI public repository where used as benchmarks.

The results show that in general the SVM achieves better forecasts, followed by the ANN. Nevertheless, this increase in performance is achieved with a higher computational effort.

Índice

Índice de Figuras

Índice de Tabelas

Capítulo 1

Introdução

Phillipe Dreyfus, engenheiro da *École Supérieure de Phisique et de Chèmie Industrielle,* introduziu a palavra **informática** em 1962 como resultado da combinação das palavras **infor**mação e auto**mática**, definindo-a como uma técnica do tratamento automático e racional da informação, considerando a informação como suporte dos conhecimentos e da comunicação, nos domínios técnico, económico e social [24]. A Academia Francesa definiu **informática** em 1967 de forma semelhante mas concedendo-lhe o estatuto de ciência [4]. Ao mesmo tempo definiu **computador** como a máquina automática capaz de realizar operações matemáticas e lógicas com aplicações científicas, administrativas ou contabilísticas [4].

A interacção entre o computador, suporte de toda a informação passível de ser digitalizada, os programas capazes de tratar essa informação, e a componente humana das organizações que se relaciona directamente com esses programas, está na base do conceito das **Tecnologias da Informação (TI)** [39], também chamadas de **Tecnologias da Informação e Comunicação** [39]. As TI trouxeram a promessa de melhoria do tratamento da informação, quer na sua recolha e no armazenamento, quer na selecção e distribuição. As organizações necessitadas de conhecimento organizacional para a implementação de estratégias, melhoria dos processos e dos produtos, e para a satisfação dos clientes, reconheceram as oportunidades que as TI proporcionaram, e por isso integraram-nas nos seus **Sistemas de Informação** organizacionais [37].

De facto, as TI provaram ser muito eficazes na recolha e armazenamento da informação, permitindo um crescimento desmesurado dos volumes de informação armazenada, tornando difícil a extracção de conhecimento recorrendo a gráficos e tabelas ou métodos estatísticos [16]. A utilização de sistemas de ***Business Intelligence*** permite combinar a recolha e armazenamento de dados com ferramentas de análise, com o objectivo principal de disponibilizar informação relevante para a tomada de decisão [37]. Estes sistemas incluem *Data Warehouses* (armazéns de dados), que são repositórios de informação organizacional em formato adequado à análise. Incluem também, os

sistemas ***On-Line Analytic Processing*** **(OLAP)** que permitem uma análise multi-dimensional dos dados**,** e o uso de técnicas de ***Data Mining*** para extracção de conhecimento [37]. Neste contexto surge a **Descoberta de Conhecimento em Bases de Dados** com o objectivo de desenvolver métodos e técnicas que, a partir de informação guardada em bases de dados, consigam descobrir novo conhecimento, novos padrões e novas tendências, tendo a utilidade e a novidade como critérios de validação [14][13].

O processo de Descoberta de Conhecimento em Bases de Dados inclui diversas etapas, tais como o pré-processamento e o *Data Mining*, sendo esta última etapa deveras importante, pois é onde efectivamente se procuram as relações entre os dados [21]. No entanto, na prática o termo *Data Mining* acabou por se tornar mais conhecido[1], pelo que é comum referir-se à designação *Data Mining* como um sinónimo do processo completo de Descoberta de Conhecimento [20]. Assim, e ao longo desta dissertação, muitas das vezes será utilizada esta noção mais abrangente. Convém referir ainda que o *Data Mining* passou a ser considerado pelas organizações como uma tecnologia crucial, a par de ferramentas OLAP, sendo mesmo considerado como capaz de dotar as organizações de capacidade cognitiva [10].

1.1. Motivação

O interesse nas áreas da Descoberta de Conhecimento em Bases de Dados e *Data Mining* emergiu devido ao rápido desenvolvimento dos métodos electrónicos de armazenamento de dados. Este enorme volume e complexidade da informação, bem como as limitações humanas condicionam a extracção de conhecimento. Todavia, como alternativa podem utilizar-se ferramentas de descoberta automática ou de *Data Mining*, tais como as **Redes Neuronais Artificiais** ou **Máquinas de Vectores de Suporte**, que permitem ultrapassar estas limitações através de uma análise aos dados em bruto, extraindo informação de alto nível, de forma a auxiliar a tomada de decisão.

As **Redes Neuronais Artificiais (RNAs)**, modelos conexionistas inspirados no funcionamento do cérebro humano, são uma técnica deveras popular na área do *Data Mining*, tendo sido utilizadas com ênfase crescente a partir de 1986, quando foi proposto o algoritmo de treino de retropropagação. Por sua vez, as **Máquinas de**

[1] Provavelmente por ser um termo mais sonante e mais pequeno.

Vectores de Suporte (MVSs)[2]**,** são mais recentes, tendo sido propostas por Vapnik e seus colaboradores em 1995. Quer as RNAs quer as MVSs permitem efectuar uma modelação não linear de dados. Contudo, é apontada uma vantagem teórica às MVSs em relação às RNAs, pois existe uma garantia da obtenção de uma solução óptima. Por outro lado, convém referir que sendo uma técnica mais recente, a utilização de MVSs em *Data Mining* é escassa, sendo portando necessário esclarecer quais as suas verdadeiras capacidades.

1.2. Objectivos

Dado o crescente interesse nesta TI, são diversas as ferramentas computacionais de *Data Mining* que actualmente se encontram disponíveis. Existem aplicações comerciais, com licenças do tipo *free*, *shareware* e *open source*. Todas têm em comum o facto de disponibilizarem diversas técnicas de *Data Mining*, sendo que as diferenças nos resultados da sua aplicação dependem dos dados em causa, das facilidades de pré-processamento disponibilizadas, ou então são devidos a pequenas diferenças na forma como os algoritmos são implementados. Por conseguinte, não é trivial a comparação de resultados entre técnicas semelhantes de ferramentas diferentes, mas é importante fazê-la para permitir que as escolhas de futuras técnicas (e/ou ferramentas) a aplicar sejam mais esclarecidas.

As RNAs são técnicas que também conseguiram grande relevo pelos seus bons resultados, no entanto, a dificuldade em compreender os modelos criados é vista como uma desvantagem. As MVSs são relativamente recentes e são tidas como uma alternativa às RNAs, com a vantagem sobre estas de garantirem uma solução óptima, além de se apoiarem na Teoria Estatística da Aprendizagem.

Convém referir, ainda, que existem outras técnicas utilizadas no processo de *Data Mining*. Por exemplo, as Árvores de Decisão/Regressão são técnicas que cultivaram com mérito uma posição de relevo, quer pelos bons resultados que são conseguidos com a sua aplicação, quer pela facilidade de tradução para linguagem humana das regras que conseguem induzir. Também os vários algoritmos que as implementam foram deveras optimizados, e apresentam-se aos utilizadores praticamente sem diferenças de

[2] Tradução adoptada do inglês *Support Vector Machines*.

ferramenta para ferramenta. Por estas razões, tendem a ser utilizadas como uma referência para comparação (*benchmarking*) para avaliação de novas técnicas.

Com esta dissertação pretende-se então esclarecer as vantagens e desvantagens das RNAs e MVSs, quando aplicadas ao processo de Descoberta de Conhecimento/*Data Mining*, comparando-as com outras técnicas (e.g. Árvores de Decisão/Regressão). Em particular, será dado um maior destaque à capacidade de previsão dos modelos.

1.3. Organização

Esta dissertação está organizada em seis capítulos. O primeiro faz a introdução ao tema da dissertação, esclarece a motivação e os objectivos, e a organização da mesma. O segundo refere-se à Descoberta de Conhecimento em Bases de Dados, esclarecendo a sua relação com o *Data Mining,* assim como esclarecendo o que é de facto o *Data Mining*. No capítulo seguinte, faz-se a descrição de várias técnicas de *Data Mining* e, no quarto capítulo, procede-se a uma análise de ferramentas que disponibilizam as técnicas de RNAs ou MVSs, seleccionando-as entre as aplicações disponíveis actualmente. No quinto capítulo, são realizadas algumas experiências que visam comparar os resultados da utilização de RNAs, MVSs e Árvores de Decisão/Regressão. Finalmente, no sexto capítulo, são tecidas algumas conclusões, é realizada a discussão sobre a dissertação e são, também, apontados caminhos de trabalho futuro.

Capítulo 2

Descoberta de Conhecimento em Bases de Dados e *Data Mining*

O aparecimento dos computadores e das TI permitiu que se conseguisse recolher e armazenar enormes quantidades de informação. No entanto, é tão grande e tão complexa a informação que actualmente se consegue recolher, que se corre o risco de não conseguir extrair todo o conhecimento que a informação encerra, ou por falta de tempo, ou por incapacidade de encontrar algum sentido, dada a sua complexidade. Daí que tenha surgido a área da Descoberta de Conhecimento em Bases de Dados com o objectivo de solucionar este problema.

Neste capítulo esclarece-se o que é a Descoberta de Conhecimento em Bases de Dados, e qual a sua relação com *Data Mining*. Fala-se das várias etapas do processo de Descoberta de Conhecimento em Bases de Dados, segundo Fayyad et al. [14] e segundo Adriaans et al. [1]. Refere-se a relação entre Sistemas de Informação e *Data Mining* e enumeram-se as várias aplicações de *Data Mining*. Finalmente, aborda-se em pormenor a etapa de *Data Mining*, os objectivos e os vários métodos, e a metodologia CRISP-DM.

2.1. O que é Descoberta de Conhecimento em Bases de Dados

A Descoberta de Conhecimento em Bases de Dados, tradução do inglês ***Knowledge Discovery in Databases*** **(KDD)**, tem como objectivo desenvolver métodos e técnicas de extracção de conhecimento de alto nível a partir de informação guardada em bases de dados [14]. Como meios físicos, usa computadores e redes de computadores. Utiliza, também, bases de dados e/ou *Data Warehouses*.

2.2. Processo de KDD

O processo de Descoberta de Conhecimento em Bases de Dados, doravante designado pela sigla KDD, pode ser definido como o processo que permite identificar padrões e/ou modelos que sejam novidade, potencialmente úteis e compreensíveis [13]. Segundo

Fayyad et al. [14], o KDD é um processo interactivo e iterativo, no qual se podem distinguir nove etapas, a saber:

- Definir o domínio de aplicação e pré-conhecimento relevante, bem como o objectivo do ponto de vista do cliente;
- Criar um conjunto de dados alvo;
- Limpeza e pré-processamento de dados;
- Redução e projecção de dados;
- Escolha do método de *Data Mining* adequado;
- Escolha do algoritmo (técnica) de *Data Mining* apropriado;
- *Data Mining*, ou seja, procura de padrões relevantes nos dados;
- Interpretação dos padrões obtidos;
- Utilização e/ou documentação do conhecimento obtido.

Por sua vez, segundo Adriaans et al. [1], a extracção de conhecimento de bases de dados é constituído pelas seguintes etapas:

- Definição do problema;
- Enriquecimento dos dados;
- Codificação dos dados;
- *Data Mining*;
- Relatório e divulgação do novo conhecimento.

Como facilmente se pode observar, estas sete etapas podem ser englobadas nas nove anteriormente referidas. A maior diferença que se pode verificar é o facto de Fayyad e seus colaboradores subdividirem a etapa de *Data Mining* em três: escolha do método de *Data Mining* adequado, escolha do algoritmo de *Data Mining* adequado e o *Data Mining* propriamente dito. Aliás, Fayyad et al. [14] resumem o processo nas seguintes etapas (Figura 1):

- **Selecção** – após definição do âmbito e dos objectivos do processo, faz-se a recolha dos dados com características que se considerem úteis;

- **Pré-processamento** – tratamento de dados errados e omissões, resolução de distribuições de dados não uniformes, etc;

- **Transformação** – procura das características mais importantes dos dados de modo a reduzir o número de variáveis ou modificar a forma de uma dada variável;

- ***Data Mining*** – selecção de métodos e técnicas para extracção de padrões dos dados;

- **Interpretação e avaliação** – visualização do conhecimento extraído para tornar possível a sua interpretação e sua avaliação, sendo possível retomar o processo em qualquer uma das etapas anteriores para uma nova iteração.

O KDD é um processo que implica iterações entre as várias etapas, bem como intervenção do utilizador ao longo do processo. Tal motiva a criação de ferramentas que suportem todo o processo e que permitam uma fácil recolha de dados a partir de bases de dados ou *Data Warehouses* [16].

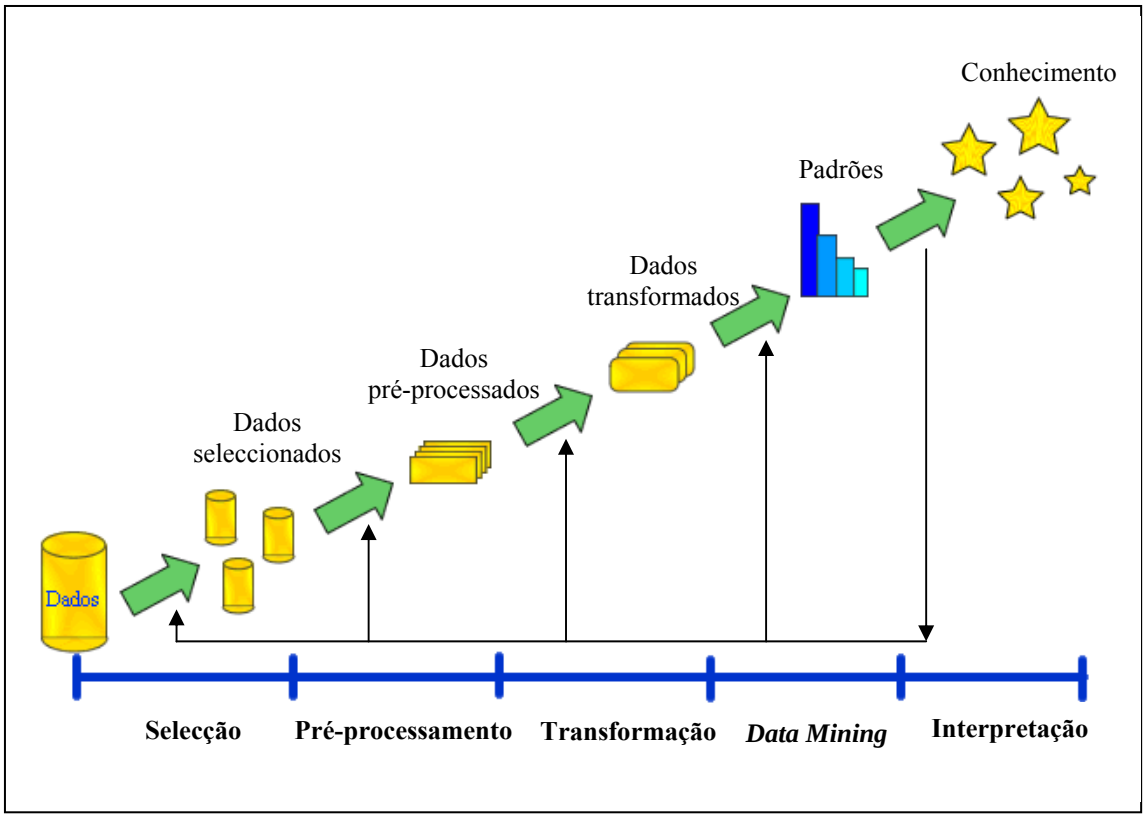

Figura 1 – Processo de KDD segundo Fayyad et al. [14].

Como se pode observar na Figura 1, o *Data Mining* é mais uma etapa do processo de KDD. King [21] descreve o *Data Mining* como sendo um processo de descoberta de relações ocultas entre os dados. Ou seja, *Data Mining* pode ser visto como um sub-processo na Descoberta de Conhecimento, envolvendo métodos e técnicas, sendo fortemente suportado pela computação. A automatização da Descoberta de Conhecimento levou à criação de várias ferramentas de *Data Mining*, algumas das quais serão abordadas mais à frente.

2.3. Sistemas de Informação e *Data Mining*

Os **Sistemas de Informação** nas organizações actuais podem ser de vários tipos, o que cria, não raras vezes, alguma confusão. No entanto, em geral mantêm algumas características comuns: todos tratam de informação, estão relacionados com organizações e com TI, quer porque os sistemas são implementados com base em computadores e afins, ou porque de alguma forma beneficiam do seu uso [8].

As organizações servem-se das novas tecnologias para a recolha e armazenamento da informação. Os **Sistemas de Gestão de Bases de Dados (SGBD)** têm assim, um papel importante nas organizações. Contudo, o conhecimento que se pode obter destes sistemas está limitado ao facto de só permitirem questões[3] simples. Os sistemas OLAP permitem análises um pouco mais complexas. Todavia, exigem uma elevada interactividade, sendo a extracção de conhecimento da responsabilidade do utilizador. Os modernos sistemas de *Business Intelligence* já incluem *Data Mining* para colmatar esta lacuna. Por conseguinte, as ferramentas comuns de análise de dados limitam-se a confirmar ou refutar hipóteses propostas pelo utilizador, dependendo o sucesso da análise da perícia do utilizador. Ao invés, no processo de KDD/*Data Mining* a produção das hipóteses é em geral automatizada. Este facto leva a um processo mais rápido, autónomo, que pode ser aperfeiçoado com resultados potencialmente mais fiáveis.

2.4. Aplicações do *Data Mining*

O *Data Mining* pode ser utilizado num vasto campo de utilizações e, embora se destaque em aplicações comerciais e científicas, têm surgido novas aplicações relacionadas principalmente com as novas TI. De seguida, serão apresentados alguns exemplos de utilizações.

2.4.1. *Customer Relationship Management* (CRM)

Para as empresas é mais prioritário manter os clientes do que angariar novos clientes. De facto, há estudos que apoiam essa estratégia. Nesta área o *Data Mining* pode ajudar na obtenção de conhecimento sobre o perfil dos clientes e na antevisão das suas necessidades. Desta forma, permite fornecer conhecimento que permita à empresa a criação de novos serviços, manter serviços personalizados, bem como criar outros produtos e serviços que permitam ou proporcionem a fidelização dos clientes [9].

2.4.2. Suporte à Decisão

Os gestores têm que tomar decisões que afectam o futuro da empresa. As decisões têm por base previsões do futuro, nomeadamente tendências, preços, situação futura do

[3] Tradução do inglês *query*.

mercado, segmentação, etc. O *Data Mining* pode fazer a diferença, fornecendo melhores previsões, identificando segmentos de consumidores e padrões de consumo, e ainda, outras ajudas importantes à decisão [36].

2.4.3. Finanças

Na área das finanças tem sido aplicado o *Data Mining* para detecção de fraudes, análise de créditos, previsões, etc. O sector bancário destaca-se nesta área, pelo largo leque de aplicações que vai desde o estudo dos comportamentos de utilizadores de cartões roubados, detecção dos mesmos, até ao CRM [36].

2.4.4. Investigação Científica

Trata-se de uma área onde existem bases de dados com enormes quantidades de informação, embora também haja situações onde os dados disponíveis são escassos. O uso de *Data Mining* vai desde a visualização da informação até à previsão [15]. Destaca-se a aplicação na medicina, na fase de diagnóstico, na identificação das melhores terapias, na pesquisa de novas formas de tratamento, etc [18][32]. Destaca-se, também, a aplicação na biologia como, por exemplo, na análise do genoma humano [17].

2.4.5. Outras Aplicações

Recentemente, tem-se dado um grande destaque ao *Text Mining*, ou seja, na extracção de conhecimento de grandes volumes de dados sob a forma de texto [19]. Outras aplicações de interesse acrescido são: o *Web Mining*, a aplicação de *Data Mining* à *Web* [3]; o *Biblio Mining*, a descoberta de conhecimento em bibliotecas [28]; e ainda , a descoberta de conhecimento em bases de dados multimédia [47].

2.5. *Data Mining*

Data Mining é a etapa de descoberta no processo de KDD (embora seja muitas vezes confundido com o próprio processo). Esta etapa envolve muitas iterações e é de importância capital no KDD. A etapa de *Data Mining* confronta-se com várias

dificuldades que podem comprometer o seu sucesso. Por exemplo, muitas vezes a informação é incompleta, ou até contém ruído. Estas dificuldades devem ser resolvidas nas etapas anteriores.

2.5.1. Métodos de *Data Mining*

Genericamente, podem-se distinguir dois objectivos principais no *Data Mining* [14]:

- **Verificação** – verificar a hipótese do utilizador; e

- **Descoberta** – procura de novos padrões, que se subdivide em:
  - **Previsão** – procura de padrões que permitam prever o futuro; e
  - **Descrição** – procura de padrões que apresentem o conhecimento de forma compreensível.

Assim, para o objectivo de previsão, os problemas são tratados como pertencentes a uma das seguintes classes:

- **Classificação** – encontrar uma função que faça o mapeamento dos dados em classes pré-definidas (e.g. diagnóstico de uma dada doença a partir de um conjunto de sintomas);

- **Regressão** - encontrar uma função desconhecida cuja saída (ou variável dependente) tem um domínio de valores reais (e.g. previsão do valor de uma acção da bolsa com base em indicadores financeiros).

Para o objectivo de descrição há vários métodos, tais como:

- **Segmentação (*Clustering*)** - procura de um número finito de conjuntos (ou *clusters*) que descrevam os dados;

- **Sumariação** – procura de uma descrição compacta de um conjunto ou sub-conjunto de dados;

- **Dependência** – procura de um modelo que descreva as relações entre variáveis;

- **Detecção de desvios** – descobrir alterações significativas nos dados.

2.5.2. Metodologias de *Data Mining*

Existem duas metodologias principais de *Data Mining*: **CRISP-DM (*Cross Industry Standard Process for Data Mining*)** e **SEMMA (*Sample, Explore, Modify, Model, Assessment*)**. A metodologia CRISP-DM foi desenvolvida por um consórcio composto por NCR Systems Engineering Copenhagen, DaimlerChrysler AG, SPSS Inc. e OHRA Verzekeringen en Bank Groep B.V [49]. Por sua vez, a SEMMA foi desenvolvida pela empresa SAS, cuja área de negócio é o *Business Inteligence* e o Suporte à Decisão [36]. O CRISP-DM tem uma maior aceitação a nível mundial, pelo que se irá descrever com mais algum detalhe.

No guia publicado pelo consórcio [49] o processo de *Data Mining* é encarado como um projecto com um ciclo de vida iterativo, tal como se pode observar na Figura 2. Esse ciclo de vida consiste em seis fases cuja sequência não é rígida, mas sim dependente do resultado de cada fase. Na Figura 2 representam-se com setas as relações mais frequentes entre as várias fases.

A primeira fase da metodologia CRISP-DM é a **compreensão do negócio,** que se centra na compreensão dos objectivos do ponto de vista do negócio, definição do problema de *Data Mining* e na criação de um plano preliminar. Segue-se a **compreensão dos dados** com a identificação de problemas na qualidade dos dados, a identificação de sub-conjuntos de dados de interesse e a formulação de hipóteses. A **preparação dos dados** é a terceira fase e prende-se com a produção do conjunto de dados (ou *dataset)* que será utilizado. Para isso, recorre-se a tarefas como a selecção de atributos, a transformação e a limpeza de dados. A quarta fase é a **modelação**, na qual são utilizadas várias técnicas de *Data Mining*, calibradas de forma a obter os melhores resultados. Na quinta fase faz-se **a avaliação** dos resultados obtidos na fase anterior. Procede-se, também, a uma revisão dos passos dados durante a modelação para garantia de que os resultados cumprem os objectivos do negócio. Só depois, deve ser tomada uma decisão quanto ao

uso dos resultados em questão. Por último, temos a fase do **desenvolvimento** que pode ir desde a produção de relatórios até à repetição de todo o processo de *Data Mining*. Esta fase é, geralmente, levada a cabo pelo cliente, mas o guia sobre CRISP-DM [49] faz a ressalva de que o analista deve certificar-se que o cliente compreendeu todas as acções que terá de executar para utilizar os modelos criados.

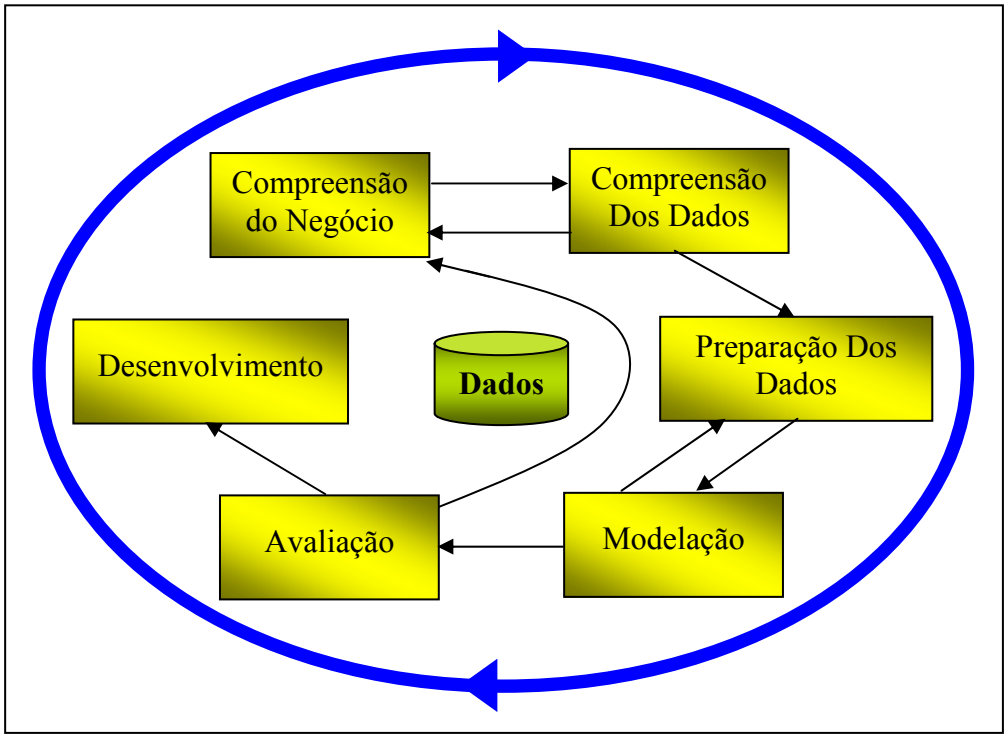

Figura 2 – Metodologia CRISP-DM [49].

2.6. Sumário

O crescimento desmesurado da quantidade de dados armazenados nos computadores actuais, proporcionado pelas TI, motivou o aparecimento da Descoberta de Conhecimento em Bases de Dados, visando a automatização da extracção de conhecimento. A motivação prende-se com o facto de, paralelamente à dimensão das bases de dados, crescer também a complexidade das relações entre os dados. Estes dois factores são deveras limitadores da capacidade humana de extracção de conhecimento útil das bases de dados.

A Descoberta de Conhecimento em Bases de Dados concretiza-se num processo, conhecido pela sigla KDD, do inglês ***Knowledge Discovery in Databases***, com várias

etapas ou sub-processos. Fayyad et al. [14] resume o processo de KDD nas seguintes etapas: Selecção, Pré-processamento, Transformação, *Data Mining*, e Interpretação. A etapa de *Data Mining* distingue-se por ser responsável pela efectiva descoberta de padrões nos dados ou pela verificação de hipóteses.

A descoberta de padrões divide-se em previsão e descrição. A descrição pode ser conseguida com os métodos de segmentação, sumariação, dependência, ou detecção de desvios. A previsão pode ser conseguida com métodos de regressão ou com métodos de classificação. Serão estes dois últimos que farão parte das experiências a realizar nesta dissertação.

A reconhecida importância do conhecimento organizacional levou à integração nos Sistemas de Informação organizacionais funcionalidades de *Data Mining*, permitindo a sua aplicação às mais variadas áreas, incluindo o *Business Intelligence*, Finanças e o CRM. Mas as aplicações do *Data Minig* não se limitam à informação organizacional, tal como se pode constatar nas recentes aplicações à *Web* e a dados multimédia.

A etapa de *Data Mining* torna-se não raras vezes, numa tarefa deveras complexa. Daí que tenham sido propostas diversas metodologias de *Data Mining*, das quais se destaca o CRISP-DM.

Capítulo 3

Técnicas de *Data Mining*

As técnicas de *Data Mining* assentam em algoritmos criados para atingirem os objectivos descritos no capítulo anterior. A mesma técnica pode ser implementada por algoritmos substancialmente diferentes, o que pode levar a resultados diversos, mediante o algoritmo utilizado. Existe também a possibilidade de usar técnicas diferentes para atingir os mesmos objectivos. Todavia, os resultados vão variar de técnica para técnica, o que obriga a uma selecção criteriosa das mesmas. As técnicas distinguem-se pela forma de representação do conhecimento (modelo) e pelo algoritmo de procura dos parâmetros internos do modelo [11].

Actualmente, a variedade de técnicas disponíveis é muito elevada, pelo que, neste capítulo, são descritas apenas algumas: as Árvores de Decisão/Regressão, a Indução de Regras, as Redes Neuronais Artificiais, e as Máquinas de Vectores de Suporte. São descritos também, os métodos de avaliação de modelos que serão utilizados nas experiências a realizar nesta dissertação.

3.1. Árvores de Decisão/Regressão

Desenvolvidas inicialmente pela Universidade de Michigan, com origem na área de Aprendizagem Automática[4], as Árvores de Decisão/Regressão são representações gráficas de regras de classificação [36] (Figura 3). A estrutura que cada representação segue é constituída por:

- **Nó raiz** – nó com o primeiro teste;
- **Nós internos** – cada um possui um teste a um atributo dos dados e têm duas ou mais sub-árvores que correspondem às respostas possíveis;
- **Ramos** – contendo valores dos atributos;
- **Folhas** – que representam as classes.

[4] Tradução adoptada para o termo *Machine Learning*.

Os algoritmos de indução de Árvores de Decisão/Regressão vão construindo a árvore de modo recursivo a partir de dados de treino, dividindo os dados em subconjuntos até que esses representem uma só classe/valor ou respeitem determinados critérios.

No âmbito dos objectivos de *Data Mining*, as árvores são mais adequadas à classificação, produzindo resultados de forma mais rápida que outras técnicas. Também podem ser utilizadas para outros objectivos mesmo que com resultados inferiores a outras técnicas, isto porque as regras que induzem possuem a vantagem de uma fácil tradução para linguagem humana. Logo são mais compreensíveis quando comparadas com outras técnicas. Para cada objectivo existem diferentes algoritmos disponíveis, tal como apresentado na Tabela 1 [36].

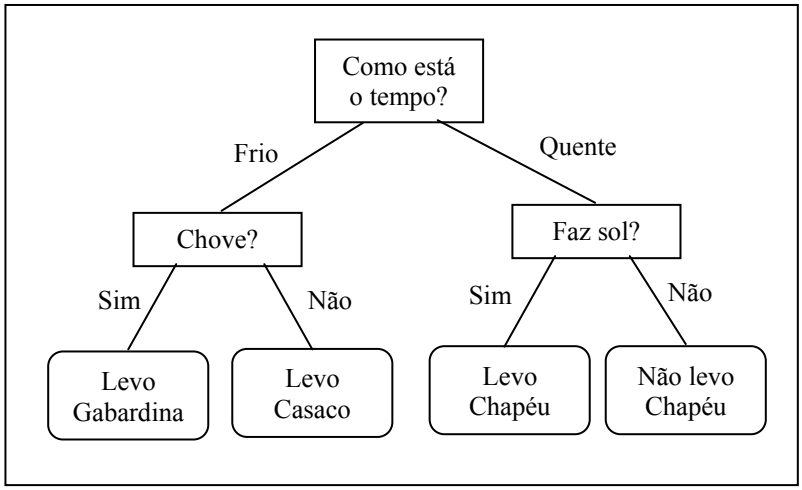

Figura 3 – Exemplo de Árvore de Decisão sobre o que vestir.

Tabela 1 – Objectivos de *Data Mining* e algoritmos de Decisão /Regressão (retirado de [36]).

Objectivos de DM	Vantagens	Desvantagens	Algoritmos
Classificação Segmentação Previsão Visualização	Construção simples e rápida. Adequada a problemas com muitas dimensões. Fácil representação e visualização.	Necessários muitos dados para descobrir estruturas complexas.	CART CHAID AID See5 ID4 ID6 C4.5 C5

Os algoritmos de indução de Árvores de Decisão/Regressão muitas vezes constroem estruturas com mais ramificações que o necessário, pelo que se torna imperativo

"podar" a estrutura. A poda pode ser feita durante a aprendizagem, o que torna o processo mais complexo. Além disso, determinadas podas só podem ser decididas após a construção da árvore, pelo que a maior parte dos algoritmos faz a poda no final da construção da árvore. Após a poda, surge o problema de determinar taxas de erro da árvore. Se houver dados em elevado número, pode-se reservar parte deles para teste após a construção da árvore (técnica de *reduced-error pruning*). Caso os dados sejam escassos, pode-se utilizar um esquema de ***Cross-Validation***. Neste caso, os dados são divididos em N blocos de dimensão semelhante. A aprendizagem faz-se com recurso a N iterações, em que a cada iteração são utilizados N-1 blocos para aprendizagem e o outro para teste, sendo este diferente a cada iteração.

Como desvantagem desta técnica, pode apontar-se o desempenho obtido em problemas com uma elevada não linearidade. Nestas situações, as Árvores de Decisão/Regressão tendem a ser ultrapassadas por técnicas com uma maior complexidade de aprendizagem, tais como as Redes Neuronais ou Máquinas de Vectores de Suporte.

3.1.1. Algoritmos de Implementação de Árvores de Decisão/Regressão

Em 1963, Morgan e Sonquist [26] propõem um método de criação de árvores através da separação dos dados em grupos sucessivos com base nos seus valores. Este método foi denominado de ***Automatic Interaction Detection* (AID)**. Hartigan [10] em 1975 introduz o teste do chi-quadrado para identificar variáveis independentes ao método **AID** dando origem ao **CHAID**. Este novo método permitiu diminuir o crescimento inicial das árvores facilitando a poda. Breiman et al. [6] desenvolveram em 1984 o algoritmo ***Classification and Regression Trees* (CART)**. Este algoritmo cria árvores binárias pelo que, se o teste de uma variável aponta para mais que dois valores, os ramos do nó de teste terão que incluir sub-testes. Tal facto leva a árvores de grande extensão. Ross Quinlan [33] desenvolveu o algoritmo ***Iterative Dichtomizer* (ID)** que teve várias versões e serviu de base ao **C4.5** que, por sua vez, também tem tido várias versões, entre as quais algoritmos para regressão. Tal como o **CHAID**, estes algoritmos identificam as variáveis independentes, mas recorrendo aos conceitos de "entropia" e "ganho de informação". A entropia está relacionada com a distribuição dos valores de uma variável, ou seja, entropia elevada quer dizer que há uma distribuição mais

uniforme, ao invés uma entropia pequena quer dizer que há um valor predominante [10]:

Entropia (Variável) = $-\Sigma_i$ [P(Variável$_i$ x $\log_2$(P(Variável$_i$))]

O ganho de informação indica a capacidade de uma variável em separar os casos de treino [10]:

Ganho (Casos, Variável) = Entropia (Casos) - Σ_i [P(Variável$_i$ x Entropia(Variável$_i$)]

3.1.2. Indução de Regras

Trata-se de uma técnica muito conhecida, geralmente utilizada para representar o conhecimento associado às Árvores de Decisão/Regressão [36]. A representação é escrita utilizando uma sintaxe do tipo: **SE** *condição* **ENTÃO** *acção*. São reconhecidas vantagens importantes na **Indução de Regras**, das quais se destaca a facilidade de explicação e compreensão das regras (Tabela 2).

Tabela 2 – Objectivos de *Data Mining* e algoritmos de Indução de Regras (retirado de [36]).

Objectivos de DM	Vantagens	Desvantagens	Algoritmos
Classificação Segmentação Previsão Associação	Construção simples e rápida Fácil representação.	Necessários muitos dados para descobrir estruturas complexas.	CHAID C4.5

3.2. Redes Neuronais Artificiais

A disciplina de Inteligência Artificial almeja o desenvolvimento de paradigmas computáveis capazes de realizar tarefas cognitivas que só são realizadas pelo cérebro humano [11]. Como consequência, surgiram as **Redes Neuronais Artificiais (RNAs)** inspiradas, precisamente, no funcionamento do cérebro humano.

Tanto o nome Inteligência Artificial, como o de Redes Neuronais Artificiais, tendem a criar grandes expectativas. Contudo, convém referir que as RNAs são modelos simplificados do sistema nervoso central, passíveis de serem implementadas por

software ou *hardware*, e capazes de realizar tarefas (como classificação e regressão) após um período de treino.

3.2.1. Do Perceptrão às RNAs

O Perceptrão, conjunto de neurónios artificiais[5] e sinapses que os interligam, foi proposto pela primeira vez por Rosenblatt em 1962 [43]. Cada neurónio calcula a média ponderada dos sinais que recebe como entrada (Figura 4). As ponderações ou pesos estão armazenados nas respectivas sinapses. O resultado desta ponderação é usado como entrada de uma função de activação (inicialmente linear), e por sua vez, a saída desta é o resultado do neurónio. Esta saída poderá ou não ser processada por outro neurónio.

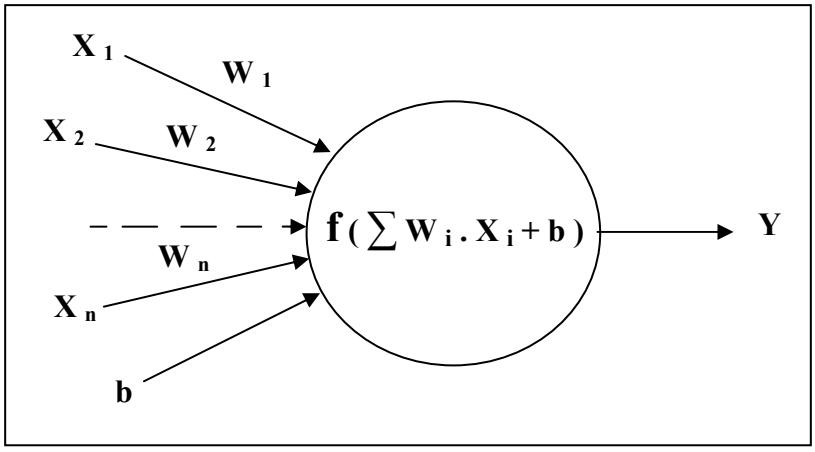

Figura 4 – Neurónio artificial de um Perceptrão.

As funções mais utilizadas como funções de activação são:

- Linear → y = x
- Tangente Hiperbólica → y = tanh(x)
- Logística → $y = 1/(1+e^{-x})$

O termo **Redes Neuronais Artificiais** só se vulgarizou na década de 1980, altura em que surgiram vários tipos como as ***Multi-Layer Perceptrons* (MLPs)**, as ***Radial Basis Functions* (RBFs)** e as ***Probablistic Neural Network* (PNN's)** [30][46].

[5] Referidos nesta dissertação apenas por neurónios

3.2.2. Topologia

Entende-se como topologia de uma RNA, o modo como os neurónios se interligam. A topologia é condicionada pelo facto da aprendizagem da rede ser ou não supervisionada. No caso de ser supervisionada, a rede é constituída tipicamente por uma camada de neurónios de entrada, uma ou mais camadas de neurónios intermédios, e uma camada de saída. Este tipo de redes, pode ainda ser dividido em **redes recorrentes**, caso existam ciclos na rede, ou **redes unidireccionais,** caso não existam ciclos, tal como acontece nas MLPs (Figura 5). As MLPs são constituídas por neurónios artificiais e por sinapses que os interligam e permitem a transmissão de sinais entre eles, tal como as outras RNAs, mas têm uma topologia em camadas e as conexões são unidireccionais [7].

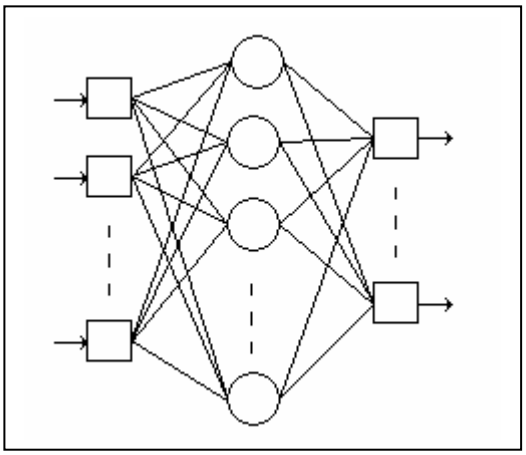

Figura 5 – Exemplo de uma rede *Multi-Layer Perceptron*.

3.2.3. Aprendizagem

Um sistema de inteligência artificial tem que ser capaz de cumprir três exigências [11]:

- Armazenamento do conhecimento;
- Aplicação do conhecimento ou raciocínio; e
- Aumento do conhecimento ou aprendizagem.

Nas RNAs, o armazenamento é realizado nas sinapses sob a forma de "pesos", que não são mais do que uma ponderação que afectará os valores produzidos pelos neurónios. Assim, a determinado conhecimento corresponde uma ou mais distribuições diferentes

de pesos, tornando cada rede única. Fornecendo sinais de entrada na rede, ela dará uma resposta que corresponderá à aplicação do conhecimento que ela possui. Os pesos e sua distribuição são obtidos através de um processo de aprendizagem da rede.

3.2.3.1. Paradigmas de aprendizagem

A aprendizagem automática tem três paradigmas fundamentais:

- **Aprendizagem supervisionada** – é assumida a presença de um professor, são conhecidas as respostas e a aprendizagem é feita com recurso a casos constituídos por dados de entrada e dados de saída.

- **Aprendizagem por reforço** - também é assumida a presença de um professor, mas neste caso não são conhecidas as respostas. A aprendizagem faz-se com recurso a "recompensas", caso as respostas sejam correctas, e "punições", caso não o sejam.

- **Aprendizagem não supervisionada** – não há professor nem são conhecidas as respostas. A aprendizagem é feita descobrindo padrões nos dados de entrada.

Nas RNAs, o paradigma de aprendizagem mais comum é o de aprendizagem supervisionada. Assim, o processo de aprendizagem ou treino da rede consiste em ajustar os valores dos pesos das sinapses de modo que as entradas na rede produzam as saídas correctas. Para tal, são utilizados algoritmos de treino dos quais o mais conhecido é o *Backpropagation*.

3.2.3.2. Algoritmos de aprendizagem

No algoritmo *Backpropagation*, em cada ciclo de aprendizagem, os erros obtidos pelas diferenças entre as saídas da rede e os valores de treino são propagados para trás, desde os neurónios de saída até aos de entrada, ocorrendo depois um reajuste dos pesos das conexões. O treino termina usualmente quando se obtém o mínimo erro à saída, no caso de regressão, ou o mínimo de classificações erradas. O algoritmo *Backpropagation* é

computacionalmente pesado e de convergência lenta pelo que surgiram melhorias a este, tais como o *QuickPropagation* desenvolvido por Fahlman em 1998 [12], ou o RPPROP (*Resilient Backpropagation*) proposto por Riedmiller [35].

3.2.3.3. O algoritmo *Backpropagation*

O algoritmo *Backpropagation*, embora não seja perfeito, permitiu uma forma automática de ajustamento dos pesos das sinapses. Os outros algoritmos mencionados são melhorias, do *Backpropagation*, pelo que a compreensão deste é relevante para a compreensão do processo de aprendizagem.

Neste algoritmo é fundamental o cálculo do erro à saída de um neurónio, erro este que, para um neurónio i, é dado por:

$$e_i(k) = y_i(k) - d_i(k)$$

em que:

- e_i → erro do neurónio i
- y_i → saída do neurónio i
- d_i → saída desejada do neurónio i
- k → entrada em causa

O erro total é $e(k) = \sum_{i=1}^{N} \frac{1}{2} e_i^2(k)$, em que *N* é o número de neurónios.

Após o cálculo do erro, é necessário actualizar os pesos das várias sinapses, recorrendo-se à regra Δ [12]:

$$\Delta w_{ij} = w_{ij}(k+1) + w_{ij}(k) = -\eta \frac{\partial e(k)}{\partial w_{ij}}$$

em que:

- w_{ij} → peso da sinapse ij
- η → controlo da aprendizagem (de 0 a 1)

Os pesos são então ajustados de acordo com $w_{ij}(k+1) = w_{ij}(k) - \eta.y_i(k).e_j(k)$, onde k designa a iteração actual do algoritmo. Isto quer dizer que o novo peso de uma sinapse w_{ij} será resultado da diferença do peso anterior e do produto do factor de controlo da aprendizagem pela saída do neurónio *i* (início da sinapse) e pelo erro do neurónio *j* (fim da sinapse). O efeito do factor de controlo da aprendizagem é o de aumentar ou diminuir o efeito do erro no peso da sinapse, pelo que pode levar a uma convergência mais ou menos rápida, conforme o seu valor for mais próximo de 1 ou de 0. Se a convergência for mais rápida, poderá cair num mínimo local ou nunca convergir devido à variabilidade dos pesos. Uma busca mais lenta poderá levar ao mínimo global mas consumirá bastante mais tempo. A escolha deste valor é de importância fundamental, já que permite o controlo directo da capacidade de generalização da rede. Numa versão que permite um maior controlo do processo, é acrescentado um novo termo, designado por *momentum* [48], onde o cálculo do novo peso é dado por:

$$w_{ij}(k+1) = w_{ij}(k) - \eta.y_i(k).e_j(k) + \mu\Delta wij(k-1)$$

A diferença está na introdução de um termo que corresponde ao ajuste da iteração anterior aferida de um factor μ, chamado de inércia ou *momentum*, que ajuda à convergência para um mínimo global, pois permite que parte do ajuste da iteração anterior vá reflectir-se no ajuste actual.

3.2.3.4. Critérios de paragem

À primeira vista pode ter-se a ilusão de que quantas mais iterações forem executadas melhor será a aprendizagem. Tal pode ser falso porque, nesse caso, poder-se-á cair numa situação em que a rede adapta-se muito bem aos casos de aprendizagem, mas responderá mal a outros casos. Está-se perante uma situação de sobre-ajustamento ou *overfitting*. Portanto, os critérios de paragem têm que conseguir um equilíbrio entre a precisão e a generalização. Na busca desse difícil equilíbrio, vários critérios de paragem podem ser utilizados, tais como [48]: erro máximo, gradiente do erro, número de iterações, ou por validação cruzada[6], sendo este último método considerado merecedor de maior confiança, mas computacionalmente mais pesado. Quando se opta pelo critério

[6] Tradução adoptada para o termo *cross-validation*.

do número de iterações corre-se o risco de sobre-ajustamento, especialmente se esse número for muito elevado. Quanto aos dois outros critérios, dependem muito da escolha do método de cálculo de erro.

3.2.4. Uso de RNAs

Antes de iniciar o treino de uma RNA, devem ser abordadas várias questões para criar as melhores condições ao sucesso da aprendizagem. Em primeiro lugar, o tamanho e a divisão dos dados. Na situação ideal, os dados devem ser em quantidade suficiente para reflectir todas as possíveis variações de respostas diferentes. Neste caso, os dados dividem-se em três partes [2]: uma para **treino**, que servirá para a actualização dos pesos das sinapses; uma para **teste**, que serve para verificação da resposta da rede a dados não usados para treino; e uma parte para **validação**, que deve ter casos diferentes dos anteriores, e permitirá aferir qual a melhor rede obtida pelo treino. Depois, há que responder às várias questões relacionadas com o processo de KDD, nomeadamente selecção, pré-processamento, transformação dos dados, etc. (abordado na Secção 2.2). Finalmente, surgem as questões de parametrização da própria rede. A inicialização dos pesos (normalmente aleatória), definição das funções de activação dos neurónios, número de camadas intermédias e número de neurónios por camada, definição dos factores de aprendizagem η e μ, e critérios de paragem. Na Tabela 3 apresentam-se os principais parâmetros de uma RNA e a sua influência no treino da mesma. Todas estas questões podem ser críticas para o sucesso da rede obtida, mas introduzem grande complexidade. Felizmente existem no mercado várias ferramentas desenvolvidas para assistir o utilizador ao longo de todo o processo (ver capítulo 4).

3.2.5. Redes RBF

Uma rede *RBF* é uma variante das RNAs apenas com uma camada intermédia, em que essa camada é constituída por neurónios que implementam, cada um deles, uma ***Radial Basis Function* (RBF)** [5]. A camada de entrada é não linear, enquanto que a de saída é linear. Uma função RBF é uma função que se caracteriza por ser monotónica à medida que se afasta do seu centro. Uma função RBF típica é a função Gaussiana (Figura 6):

$$f(x) = \exp\left(-\frac{(x-c)^2}{r^2}\right)$$

Tabela. 3 – Efeito dos valores extremos dos parâmetros de treino de uma RNA (retirado de [30]).

Parâmetro	Muito grande	Muito pequeno
Nº de camadas intermédias	*Overfitting.*	Não consegue obter o conhecimento.
η	Instabilidade em torno da solução óptima.	Treino muito lento.
μ	Reduz o risco de mínimo local, acelera o treino, aumenta o risco de instabilidade.	Sem efeito de supressão do risco de mínimo local, treino muito lento.
Nº de ciclos	Má generalização.	Não consegue obter o conhecimento.
Conjunto de treino	Baixo erro, boa generalização.	Fraca generalização ou limitada.
Conjunto de teste	Confirmação da capacidade de generalização.	Incapaz de confirmar a capacidade de generalizaçã.o

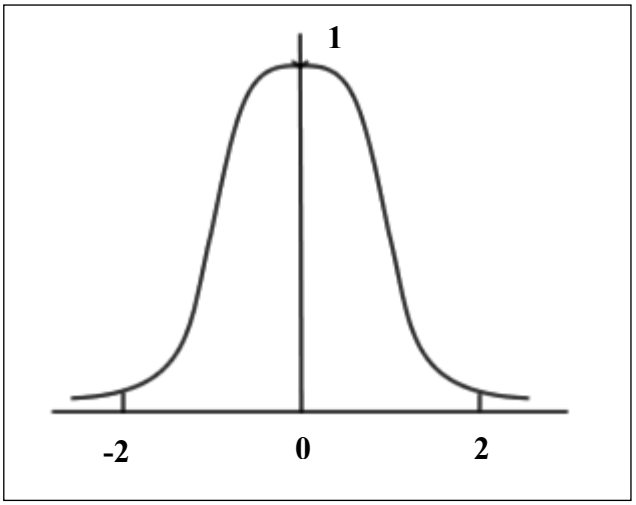

Figura 6 – Função Gaussiana para c=0 e r=1.

em que c é o centro da função, e r o raio. A função base é da forma $f(\|\vec{x}-\vec{c}\|)$, em que $\vec{c}$ é um vector de comparação ou referência, e $\vec{x}$ é o vector a comparar, resultando da função uma distância ou desvio [41]. O treino de uma rede RBF consiste em minimizar o somatório dos desvios verificados nos diversos neurónios da camada intermédia. As redes RBF têm a propriedade de conseguirem uma aproximação não linear do tipo

"hiper-esfera", conseguida pelas RNAs comuns só com recurso a várias camadas, e têm sido aplicadas com sucesso a grande diversidade de problemas [41].

3.2.6. Outras Redes

As RNAs, e em particular as MLPs e RBF, são usadas em situações de aprendizagem supervisionada, mas há redes diferentes capazes de bons resultados noutras situações. Por exemplo, no caso de aprendizagem não supervisionada existem as redes de **Kohonen**, as quais, na realidade, são constituídas por um conjunto de redes que se auto-organizam agrupando-se em *clusters*.

Para aprendizagem por reforço há redes de **Hebb**, **Hopfield** e **Máquinas de Boltzmann**, qualquer delas com a desvantagem de o processo de aprendizagem ter tendência a tornar-se extremamente demorado.

3.3. Máquinas de Vectores de Suporte para Classificação

Apoiando-se na Teoria Estatística da Aprendizagem, as *Support Vector Machines*, doravante designas pelo termo **Máquinas de Vectores de Suporte (MVSs),** surgiram pela primeira vez na década de setenta do século passado com o trabalho de Vapnik em 1979 [42]. Só recentemente a comunidade científica lhes tem dado mais importância. Tratam-se de algoritmos que podem ter algumas variações mediante a aplicação, criados inicialmente para classificação de dados, recentemente começaram a ser aplicadas também em regressão [44].

O nome de "vectores de suporte" advém do algoritmo "suportar-se" em alguns dados para definir a distância entre as classes. As figuras seguintes ilustram o conceito. Na Figura 8 está representado um conjunto de dados do tipo $(x_i;y_i)$, com $x_i \in \Re^n$, e $y_i \in \{-1;+1\}$, em que os pontos a cheio correspondem a $y_i=-1$ e os outros a $y_i=+1$. No exemplo aqui apresentado, a classificação é óbvia pois visualmente podemos estabelecer várias linhas de fronteira para separar as classes (Figura 7).

Uma linha de separação é do tipo $w \cdot x_i + b = 0$ (equação de uma recta neste exemplo), com w a inclinação da linha, e b a abcissa de início da linha. Qualquer uma das linhas

representadas na Figura 7 poderia ser uma linha de separação desde que satisfaça as seguintes condições [40]:

$$(w.x_i) + b \geq +1 \Rightarrow y_i = +1$$
$$(w.x_i) + b \leq -1 \Rightarrow y_i = -1$$

que pode ser resumido em $y_i.[(w.x_i) + b] \geq +1; i = 1,...,n$.

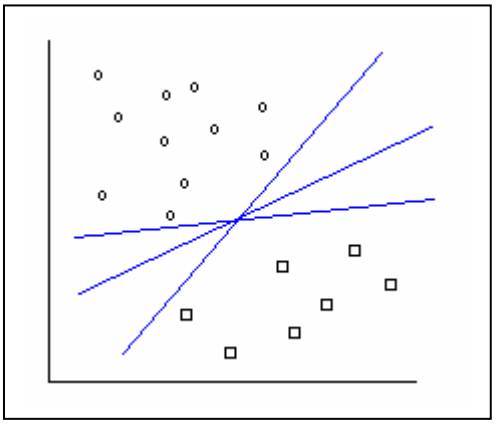

Figura 7 - Possíveis linhas de fronteira.

Do ponto de vista prático não basta que a linha de separação respeite estas condições, porque poderá ficar muito próxima de alguns dados, potenciando classificações erradas. Por isso, é necessário escolher a linha de separação que garanta a maior distância aos dados (Figura 8).

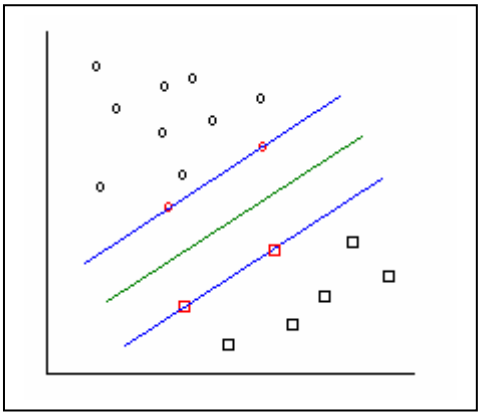

Figura 8 – Nas MVSs as margens são definidas pelos dados.

A maior distância é determinada pela minimização dos vectores normais à linha [40]:

$$d(w,b) = \min_{x_i|y_i} \frac{w.x_i + b}{|w|} - \max_{x_i|y_i} \frac{w.x_i + b}{|w|}$$

e obter-se-á $d(w,b) = \min_{x_i|y_i} \frac{1}{|w|} - \max_{x_i|y_i} \frac{-1}{|w|} = \frac{1}{|w|} - \frac{-1}{|w|} = \frac{2}{|w|}$,

cuja derivada é $d'(w) = \frac{1}{2}.|w|$.

A solução para o problema de encontrar a linha de separação que respeita as várias condições pode ser obtida pela resolução da seguinte função Lagrangeana [40], em que α_i são os multiplicadores Lagrangeanos:

$$L(w,b,\alpha) = \frac{1}{2}.|w|^2 - \sum_{i=1}^{n} \alpha_i.\{[(x_i.w) + b].y_i - 1\}$$

Esta função tem que ser minimizada em ordem a w e b, e maximizada em ordem a $\alpha_i \geq 0$, pelo que terá um ponto óptimo a que corresponderá as soluções w_0, b_0 e α_i^0, resultando numa linha de separação com as seguintes propriedades:

$$\sum_{i=1}^{n} \alpha_i^0.y_i = 0, \alpha_i^0 \geq 0, i = 1,...,n$$

$$w_0 = \sum_{i=1}^{n} \alpha_i^0.y_i.x_i, \alpha_i^0 \geq 0, i = 1,...,n$$

Segundo o teorema de Kühn-Tucker, é necessário ainda verificar a condição:

$$\alpha_i^0.\{[(x_i.w_0) + b_0].y_i - 1\} = 0, i = 1,...,n$$

que substituindo na função Lagrangeana resulta em:

$$W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i.\alpha_j.y_i.y_j.(x_i.x_j), \alpha_i \geq 0, i = 1,...,n$$

função esta restringida a $\sum_{i=1}^{n} \alpha_i . y_i = 0$. A solução é da forma de um vector expresso sob a forma de $\alpha^0=(\alpha_i^0,\ldots, \alpha_i^0)$, podendo escrever-se a função de decisão [40]:

$$f(x) = sign(\sum_{Vect.\sup orte} \alpha_i^0 . y_i . (x_i . x) - b_0)$$

sendo x_i os vectores de suporte, e $b_0 = \frac{1}{2}\left[\left(w_0 . x^*(1)\right) + \left(w_0 . x^*(-1)\right)\right]$, com x*(1) um qualquer vector de suporte pertencente à primeira classe, e x*(-1) um qualquer vector de suporte pertencente à segunda classe. Contudo, os problemas nem sempre são lineares. Assim, é necessário fazer uma transformação aos dados para que possam ser separados linearmente. As MVSs fazem uma transformação não linear aos dados para um espaço multi-dimensional onde ficará uma imagem dos dados que permita uma separação linear. Para tal transformação, as MVSs recorrem a **métodos de *Kernel***, que podem ser funções semelhantes às utilizadas pelas redes RBFs ou MLPs, ou então outras específicas ao problema em causa.

3.3.1. Algoritmo MVS

Um algoritmo que implemente uma MVS, genericamente falando, é constituído por várias etapas. A primeira é chamada de *mapping* e consiste em transformar o espaço dos vários atributos dos dados num hiper-espaço (espaço multi-dimensional). O objectivo do *mapping* é o de permitir a separação linear dos dados que terá lugar no hiper-espaço. A dimensionalidade deste espaço vai ser a suficiente para que a separação seja linear, portanto, feita com um hiperplano. O *mapping* é conseguido com recurso aos *métodos de Kernel*. A segunda etapa consiste na definição do hiperplano. A definição deste é feita com recurso a vectores. Estes, por sua vez, são construídos a partir de alguns dos atributos dos dados após a etapa de *mapping*, denominados de *features*. A selecção destes atributos é feita de modo a que o hiper-plano consiga a separação com a maior distância possível entre as classes. À escolha destes atributos dá-se o nome de *feature selection*. A última etapa consiste na apresentação dos resultados para posterior avaliação e interpretação. Na Figura 9 ilustra-se o algoritmo MVS.

Do ponto de vista teórico, a transformação levada a cabo pelos métodos de *Kernel* é de enorme potencial, permitindo que um algoritmo destes seja capaz de separar um qualquer conjunto de dados. Na prática, os métodos de *Kernel* têm limites no que diz respeito à capacidade dimensional da transformação dos dados, o que reduz o potencial de separação. Além disso, os dados normalmente sofrem de alguns defeitos como, por exemplo, o ruído, dados incompletos, entre outros, o que também diminui o potencial de separação.

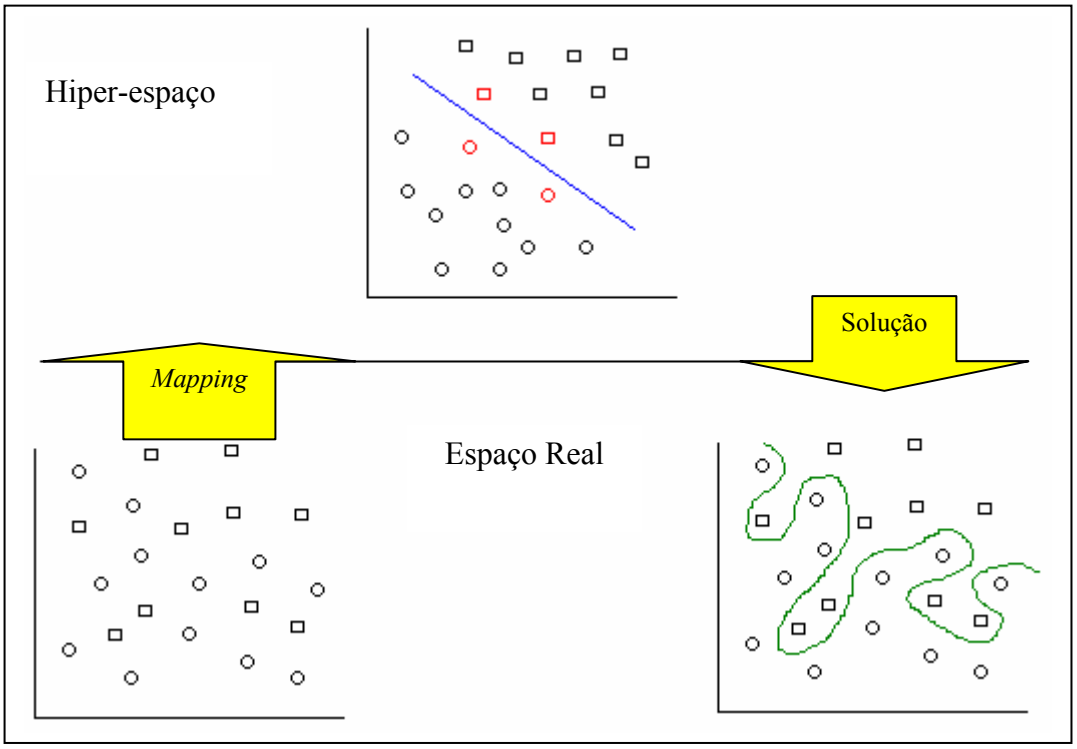

Figura 9 – As etapas de um algoritmo MVS

3.3.2. Métodos de *Kernel*

Os Métodos de *Kernel* produzem uma transformação aos dados, dando-lhes dimensionalidade tal que sejam linearmente separáveis. São muitos os métodos de *Kernel* propostos, mas os mais utilizados são [40]:

- Linear → $K(x,y) = x.y$
- Polinomial → $K(x,y) = (a.x.y + b)^c$, com $a>0$
- *Radial Basis Function* (RBF) → $K(x,y) = \exp(-a.||x-y||^2)$, com $a>0$

- *Multi-Layer Perceptron* (MLP) → K(x,y) = tanh(a.x.y + b)

3.3.3. Escolha do *Kernel*

A escolha do *Kernel* que fará o *mapping* dos dados, pode ser de importância capital para o sucesso da MVS. Por um lado, o *Kernel* tem que ter capacidade de transformar o espaço dos dados num espaço de dimensionalidade suficiente para que a separação seja linear, por outro lado, a capacidade de transformação não deve ser tal que leve ao sobre-ajustamento. Vapnik propõe o **Método de Minimização do Risco** para selecção do método, tal que [43]:

$$R(\alpha) \leq R\_emp(\alpha) + \sqrt{\left(\frac{h\left(\log\left(2l/h\right)+1\right) - \log(\eta/4)}{l} \right)}$$

Onde:

R → Risco esperado – é o erro máximo do *Kernel* em questão.
R_emp → Risco (erro) do *Kernel* na amostra em causa.
h → Dimensão VC – parâmetro que representa a capacidade do *Kernel* transformar os dados.
l → nº de itens da amostra.
η → Probabilidade do risco esperado ser ultrapassado (à escolha do utilizador).

Assim, deve-se escolher o *Kernel* que para os dados em questão e para um η escolhido pelo utilizador, se obtenha o menor valor de:

$$\sqrt{\left(\frac{h\left(\log\left(2l/h\right)+1\right) - \log(\eta/4)}{l} \right)}$$

3.3.4. Casos Não Separáveis

Há casos em que o *Kernel* não tem capacidade de transformar o espaço dos dados num espaço de dimensionalidade suficiente para que a separação seja linear, levando a que os dados possam sofrer de alguns defeitos. Assim, é permitida uma margem para alguns dados mal classificados (designada de *soft margin*). O controlo desta margem é feito

pelo utilizador com recurso a um parâmetro C. Este parâmetro tem como efeito, a limitação dos valores que os multiplicadores Lagrangeanos podem tomar, restringindo-os a $0 \leq \alpha_i^0 \leq C$ [43].

3.3.5. Regressão com MVSs

A regressão com MVSs é conseguida alterando a função do custo para que inclua um parâmetro de distância (Figura 10). Esse parâmetro (ξ) vai permitir a criação de uma margem na qual os dados serão ignorados, pelo que o parâmetro também é chamado de ξ-insensitivo. Portanto, utilizar MVSs em regressão torna necessário o controlo de dois parâmetros (C e ξ), para além dos parâmetros associados ao *Kernel*. Normalmente, estes parâmetros são determinados empiricamente, sendo usual C tomar valores múltiplos de 10, e ξ valores muito pequenos como 0,5 ou 0,05. O controlo destes parâmetros é também, fundamental para o controlo da generalização do modelo [40].

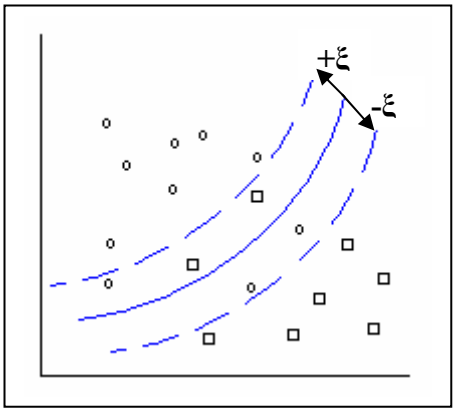

Figura 10 – Margem criada pelo parâmetro ξ-insensitivo.

3.3.6. Treino de MVSs

A função de decisão de uma MVS genérica, não linear pode ser expressa por [31], em que K é a função *Kernel*:

$$f(x) = sign(\sum_{Vect.Suporte} \alpha_i^0 . y_i . K(x_i . x) - b_0)$$

O treino de uma MVS consiste em encontrar os multiplicadores Lagrangeanos α_i, através da minimização da expressão:

$$W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1, j=1}^{n} \alpha_i . \alpha_j . y_i . y_j . K(x_i . x_j), \alpha_i \geq 0, i = 1, ..., n$$

Esta expressão está sujeita ás condições de Karush-Kuhn-Tucker (KKT):

$$\alpha_i = 0 \Rightarrow y_i . u_i \geq 1$$
$$0 < \alpha_i < C \Rightarrow y_i . u_i = 1$$
$$\alpha_i = C \Rightarrow y_i . u_i \leq 1$$

em que u_i é a saída da MVS para o exemplo de treino i.

A resolução desta expressão implica a utilização de uma matriz de dimensão igual ao número de exemplos de treino, o que em utilizações normais leva a que não haja espaço em memória de um computador para essa matriz [31].

Vapnik [43] descreve um método de resolução da expressão conhecido por "*chunking*", como o retirar da matriz os exemplos com α_i=0, já que tal não afecta a solução. Isto tem como consequência a redução da dimensão da matriz mas, mesmo assim, não é possível a resolução de problemas com número elevado de exemplos. Além disso, a determinação dos multiplicadores Lagrangeanos é feita numericamente, o que torna o processo muito lento [31]. Através de uma técnica de decomposição, é possível reduzir o problema a vários pequenos sub-problemas, permitindo solucionar casos com número elevado de exemplos.

Outro método de treino de MVSs é o ***Sequential Minimal Optimization*** **(SMO)**. O SMO também decompõe o problema em vários sub-problemas, mas resolve-os analiticamente ao invés de numericamente, o que reduz muito ao tempo de processamento [31]. Para tal, vai optimizando os multiplicadores Lagrangeanos, dois de cada vez, após o que actualiza a MVS. Para acelerar o processo usam-se, ainda, métodos heurísticos de escolha dos multiplicadores que necessitam ser optimizados (ou seja, que não respeitam as condições KKT) [31].

3.4. Avaliação de Modelos

O resultado da etapa da utilização de um algoritmo de *Data Mining* é um modelo. É comum utilizar mais do que um algoritmo sobre os mesmos dados, cada um produzindo o respectivo modelo. A ideia é escolher o modelo que melhores resultados obtém. Quer se usem vários algoritmos ou só um, põem-se sempre a questão da eficácia do modelo. Torna-se, assim, necessária a utilização de métodos de avaliação dos modelos que nos permitam aferir o grau de eficácia dos mesmos.

3.4.1. Matriz de Confusão

Utilizada em classificação, a **Matriz de Confusão** permite uma visualização inequívoca dos resultados de um modelo [23]. Os resultados são apresentados sob a forma de tabela de duas entradas: uma das entradas é constituída pelas classes desejadas, a outra pelas classes previstas pelo modelo. As células são preenchidas com o número de instâncias que correspondem ao cruzamento das entradas. Na Tabela 4 ilustra-se um exemplo de uma matriz de confusão, em que a entrada vertical são as classificações obtidas pelo modelo, e a entrada horizontal são as classificações originais dos dados. Pode-se ver que no caso da classe B, foram classificados correctamente 46 instâncias, e incorrectamente 4. Já no caso da classe A, todas as instâncias foram correctamente classificadas.

Tabela 4 – Exemplo de Matriz de Confusão.

	A	**B**	**C**
A	50	0	0
B	0	46	4
C	0	1	49

3.4.2. Regressão

Nos modelos de regressão pretende-se escolher aquele que produz valores mais próximos dos dados. A diferença entre o **valor real** (y) e o **previsto** ($\hat{y}$) é designada por **erro** ou resíduo (e_i), e pode-se calcular um erro global, ou seja, de todos os valores previstos, usando as seguintes medidas [45]:

- ***Mean Absolute Deviation*** **(MAD)*:*** $MAD = \frac{1}{N}\sum_{i}^{N} | e_i |$

- ***Sum Squared Error* (SSE):** $SSE = \sum_{i}^{N} e_i^2$
- ***Mean Squared Error* (MSE):** $MSE = \frac{SSE}{N}$
- ***Root Mean Squared Error* (RMSE):** $RMSE = \sqrt{MSE}$
- ***Root Relative Squared Error* (RRSE):** $RRSE = \frac{RMSE}{RMSE_{\bar{y}}}$

onde RMSE $_{\bar{y}}$ denota o valor de RMSE calculado para o método simples de prever o valor de *y* com o valor da sua média ($\bar{y}$). De notar que o RRSE é uma medida que é independente da escala dos valores de *y*, sendo que um valor abaixo de 100% significa que o método de previsão avaliado é melhor do que o método simples da média.

3.5. Sumário

Existem diversas técnicas de *Data Mining* cada uma com as suas potencialidades. Cada técnica distingue-se pela forma de representação do conhecimento e pelo algoritmo de procura dos seus parâmetros internos. No caso das Árvores de Decisão, entre os algoritmos mais divulgados estão o CHAID (AID com teste do chi-quadrado) e o ID que serviu de base ao famoso C4.5. Distingue-se ainda o CART como um algoritmo adequado a Árvores de Decisão e Árvores de Regressão. No caso da Indução de Regras, utilizam-se normalmente os mesmos algoritmos de Árvores de Decisão/Regressão, em que a diferença reside na forma de visualização do modelo obtido. As técnicas de Árvores de Decisão/Regressão e a Indução de Regras produzem modelos de fácil compreensão, mas revelam resultados que tendem a não ser tão bons em casos de não linearidade dos dados.

As RNAs, modelos simplificados do sistema nervoso central, são constituídas por vários neurónios organizados em camadas. Distinguem-se vários tipos de redes, quer pelas funções de activação utilizadas nos neurónios, quer pelo fluxo de sinais nas sinapses. O fluxo pode ser recorrente, ou seja, com ciclos ou unidireccional. Podem distinguir-se, ainda, pelo paradigma de aprendizagem com as redes RBF e MLPs a utilizarem o paradigma de aprendizagem supervisionada.

O tipo de RNAs mais comum é o MLP. Para este tipo de rede, o algoritmo de aprendizagem utilizado é o *Backpropagation*. Após a definição da topologia da rede, o controlo da aprendizagem é conseguido à custa de vários parâmetros, tais como a taxa de aprendizagem ou número de iterações ou épocas. O controlo adequado destes parâmetros é crucial para obter o melhor equilíbrio entre a capacidade de previsão da rede relativamente aos casos de treino (especialização), e a capacidade de fazer boas previsões com novos dados (generalização).

De uma forma geral, as RNAs não só revelam resultados melhores que outras técnicas nos casos de não linearidades dos dados, como também trabalham bem com dados incompletos, revelando boa capacidade de generalização. Mas de facto, funcionam como uma "caixa negra", sendo os modelos criados de difícil compreensão.

As MVSs apoiam-se na Teoria Estatística da Aprendizagem e distinguem-se por fazerem a separação dos dados tendo por base alguns **vectores de suporte**. À partida a separação possível é linear, pelo que para se conseguir uma separação não linear é necessário transformar os dados, aumentando-lhes a dimensão. Assim, a capacidade de separação depende muito da transformação dos dados que é feita à custa dos chamados Métodos de *kernel*. Esses métodos podem ser variados, desde formas polinomiais, RBF, ou MLPs. A possibilidade de utilização de Métodos de *Kernel* do tipo RBF ou mesmo MLPs, confere-lhes o potencial de bons resultados em casos de não linearidade.

Em geral, a aprendizagem das MVSs é feita com recurso ao algoritmo SMO. Este algoritmo divide o problema com uma elevada quantidade de dados em sub-problemas, tal como o "*chunking*". Contudo, calcula os multiplicadores Lagrangeanos de modo analítico, o que permite uma redução do esforço computacional.

Existem diversas métricas para a avaliação dos modelos criados por técnicas de *Data Mining*. Destaca-se a Matriz de Confusão para a avaliação dos modelos em classificação. Em regressão destacam-se várias métricas com ênfase na métrica RRSE, pois esta será utilizada nas experiências realizadas nesta dissertação.

Capítulo 4

Análise de Ferramentas

Como já foi dito, a Descoberta de Conhecimento em Bases de Dados (KDD), tem como objectivo desenvolver métodos e técnicas de extracção de conhecimento de alto nível a partir de informação guardada em bases de dados [14]. Para atingir este objectivo usa computadores bem como bases de dados e/ou *Data warehouses*. É, portanto, incontornável o desenvolvimento de aplicações de *software* que implementem técnicas de *Data Mining* ou acompanhem mesmo todo o processo de KDD.

Com este capítulo pretende-se fazer um levantamento e a caracterização de ferramentas de *Data Mining*, com um particular destaque para com as que implementam RNAs ou MVSs. Visto que actualmente existe um enorme número de aplicações nesta área, trata-se de uma tarefa morosa. Assim, este levantamento de caracterização será limitado às ferramentas mais utilizadas ou conhecidas.

4.1. Perspectivas de Caracterização

No desenvolvimento de aplicações de *softwar*e, sejam elas quais forem, terão necessariamente que ser levados em consideração factores de decisão como o domínio da aplicação ou o âmbito da aplicação, a linguagem de programação a ser utilizada, a plataforma de sistema operativo em que funcionará, etc. Assim, as aplicações podem ser classificadas em diversas perspectivas, consoante as suas características técnicas e não só. Em seguida, é efectuado um resumo sobre alguns estudos de classificação de ferramentas de KDD/*Data Mining*.

Santos e Azevedo [36] apontam várias possibilidades de caracterização. Começam por apontar a **linguagem de programação** a ser utilizada, e depois a **plataforma de sistema**, realçando que aplicações multi-plataforma têm um ponto forte relativamente a outras. Também se pode caracterizar uma ferramenta pela sua **escalabilidade** e **portabilidade**, bem como pelo seu **estado de desenvolvimento**, e ainda, pela possibilidade de **integração** com outras aplicações. Para além destas características

mais técnicas, são referidas ainda outras de natureza menos técnica como o **tipo de licenciamento** (*freeware*, *shareware*, *General Public Licence* – ***GNU*** ou licença comercial), e o **tipo de aplicação**, distinguindo as aplicações de carácter académico (desenvolvidas com o intuito de investigação e criação de novas soluções e de protótipos), e as aplicações comerciais (mais orientadas para o suporte empresarial e a prestação de serviços).

Goebel et al. [16], por sua, vez apresentam um esquema de caracterização em três grupos: características gerais, conectividade a bases de dados e características de *Data Mining*. O grupo **características gerais** contém factores tais como: o estado de desenvolvimento do produto; o tipo de licenciamento; a disponibilidade ou não de uma versão de demonstração (*Demo*); as arquitecturas suportadas (*stand alone*, *client/server* ou *parallel processing*); e os sistemas operativos para os quais a aplicação é disponibilizada. Na **conectividade a bases de dados** estão englobados: os formatos de dados reconhecidos pela aplicação; o tipo de conexão (*online, offline*), o número máximo de instâncias suportadas; o tipo de modelo de dados (relacional, orientado a objectos, em tabela); os tipos de atributos suportados (contínuos, discretos ou simbólicos); e os tipos de *queries*, característica esta relacionada com a interface (e.g. *Structered Query Language* - *SQL*, *Graphical User Interface* – GUI ou linguagem específica da aplicação). Finalmente, nas **características de *Data Mining***, incluem-se: as tarefas de descoberta, tais como pré-processamento, previsão, classificação, associação, segmentação, visualização e análise exploratória; a metodologia de descoberta, referindo-se com esta característica às técnicas disponibilizadas (RNAs, Árvores de Decisão/Regressão, Indução de regras, etc); e a interacção humana. Neste último factor, pretende-se medir qual o grau (maior ou menor) de necessidade de intervenção humana no processo (e.g. autónoma, guiada ou interactiva).

King et al. [22] definiram cinco categorias de características de *software* de *Data Mining*, a saber: **capacidade**, que caracteriza e classifica o que uma ferramenta pode fazer; **facilidade de aprendizagem/utilização**; **interoperabilidade,** que caracteriza a possibilidade de integração com outras aplicações; **flexibilidade** para caracterizar as possibilidades de alteração de parâmetros críticos da ferramenta ao longo do processo; e a **precisão**.

Nesta dissertação será utilizada uma caracterização semelhante à proposta por Goebel et al.[16], mas com algumas alterações. A primeira diz respeito ao primeiro grupo, **características gerais**, ao qual será acrescentado o *site* onde se pode encontrar a ferramenta. A segunda alteração é no segundo grupo, **conectividade a bases de dados**, que englobará apenas os formatos de dados reconhecidos pela aplicação. A última alteração corresponde à divisão do último grupo, **características de *Data Mining***, em dois grupos: **objectivos de *Data Mining***, que engloba diversos tipos de tarefas que a ferramenta disponibiliza (e.g. classificação ou regressão), e **técnicas de *Data Mining***, englobando as técnicas implementadas pela ferramenta.

4.2. Quais as Ferramentas de *Data Mining* Mais Utilizadas?

Santos e Azevedo [36] fazem a caracterização de uma série de ferramentas consideradas das mais utilizadas, embora neste estudo não seja referido um critério de selecção nem um suporte à sua escolha. Por outro lado, Goebel et al. [16] não referem critérios de selecção, embora apresente diversos critérios de exclusão de ferramentas, tais como:

- funciona como servidor de informação para outras ferramentas de *Data Mining*;

- embora possua alguma possibilidade de ser usado para *Data Mining* não foi desenvolvido especificamente para tal (aponta como exemplo o *Matlab*);

- designado de *Data Mining*, mas que só serve como ferramenta de visualização (tal como o *Oracle Discoverer*);

- disponibilizado por companhias de consultadoria ou de soluções específicas, não constituindo produtos de aplicação geral (não podendo ser classificados como produtos *off-the-shelf*).

King et al. [22] reduzem a selecção a catorze ferramentas por serem as que foram disponibilizadas pelos fornecedores para a sua investigação e por usarem uma das seguintes técnicas: Árvores de Decisão/Regressão, Indução de Regras, RNAs ou Redes Polinomiais.

No sítio www.kdnuggets.com[7] [50] encontram-se os resultados de um inquérito alargado a utilizadores de KDD/*Data Mining*, onde a questão principal era: "Qual a ferramenta de *Data Mining*/analítica que utilizou em 2006?". Claro que não tem o rigor de um estudo estatístico (pode até ter sido adulterado por companhias fornecedoras de *software*, e isso é reconhecido no portal), mas tendo em consideração a falta de estudos mais rigorosos e fidedignos, foi considerado que seria interessante analisar os resultados obtidos. Os resultados encontram-se resumidos na Figura 11, na qual se constata que a maioria das ferramentas fazem parte das referências anteriores (e.g. **Clementine**). No entanto, existem diversas novas aplicações (e.g. **Equbits**), não havendo, por isso, um consenso.

4.2.1. Critérios de Selecção das Ferramentas

Devido à falta de estudos com rigor científico sobre quais as ferramentas mais utilizadas, optou-se por elaborar uma lista de todas as ferramentas conhecidas, sem excepções, utilizando como referências Santos e Azevedo [36], Goebel et al.[16], King et al. [22], www.kdnuggets.com [50], e www.the-data-mine.com [52]. Ao todo, a lista contém um total de 159 ferramentas. Convém referir que este não é um número inesperado dada a importância atribuída actualmente à área do KDD/*Data Mining*.

Como uma análise exaustiva de todas estas ferramentas se encontra fora do âmbito desta dissertação, optou-se por excluir ferramentas que não tivessem ou RNAs ou MVSs. Com este critério, a lista inicial foi reduzida para 36 ferramentas. Estas ferramentas serão caracterizadas na próxima secção segundo as perspectivas atrás referidas.

[7] Trata-se de um portal que agrega informação a nível mundial relacionada com o KDD e o *Data Mining*, incluindo software, casos de estudo, notícias, sondagens, etc.

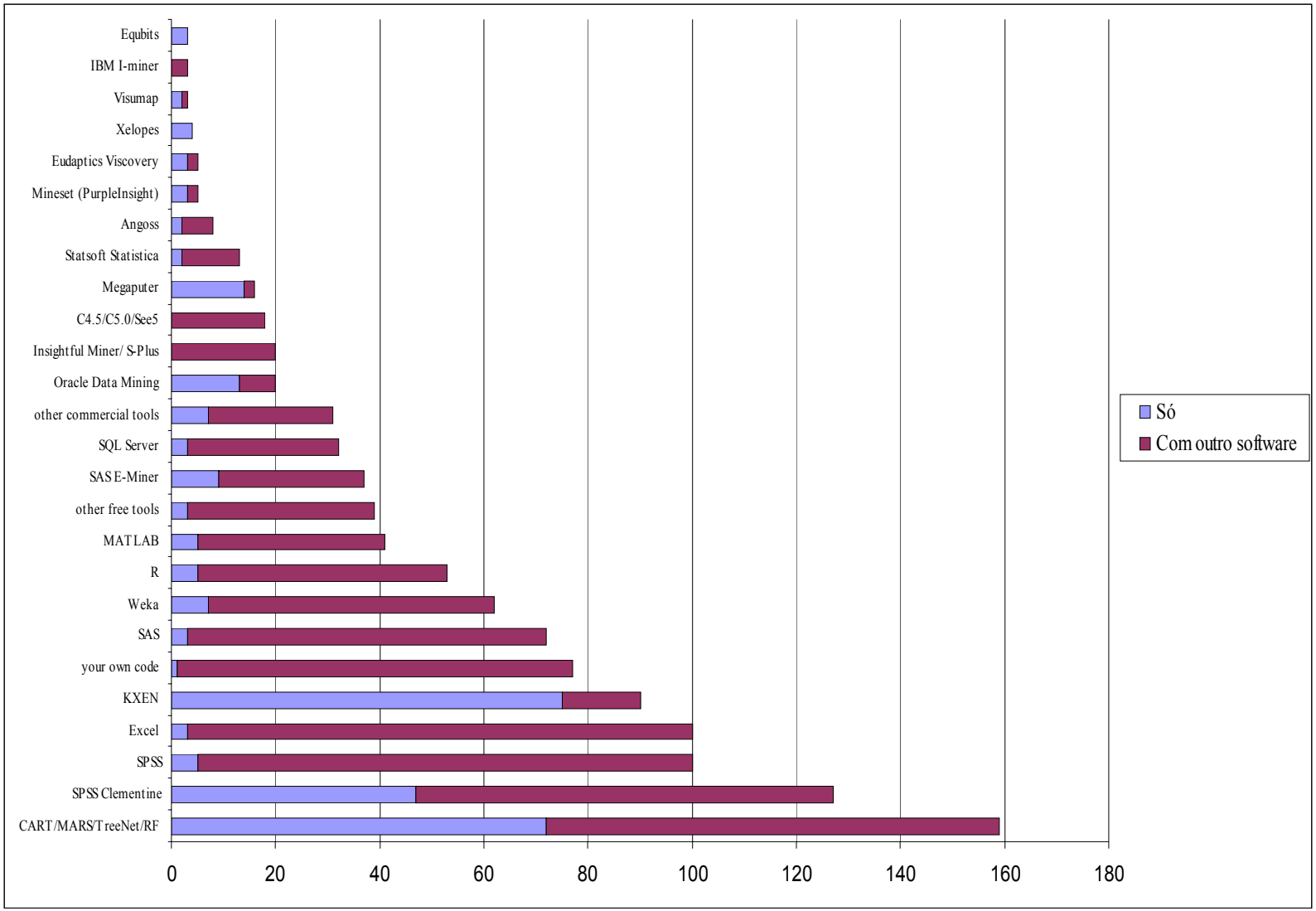

Figura 11 – "Qual a ferramenta de *Data Mining*/analítica que utilizou em 2006?" (retirado de [50]).

4.2.2. Caracterização das Ferramentas

Em seguida é feita a caracterização das ferramentas seleccionadas recorrendo a tabelas e gráficos. Os gráficos são utilizados para facilitar a visualização da informação contida nas tabelas. Nas três primeiras tabelas de caracterização de ferramentas (Tabelas 5, 6 e 7), caracterizam-se as ferramentas seleccionadas para estudo segundo as **características gerais**. A Tabela 5 apresenta o sítio de cada ferramenta e a Tabela 6 apresenta as ferramentas classificadas segundo as seguintes características:

- a **versão** - final (F) ou beta (B);
- a **licença** - comercial (C), *freeware* e *shareware* (F) ou pública (P);
- a **disponibilizade** – se é ou não disponibilizada uma versão de demonstração (*Demo*) ou a ferramenta é totalmente operacional para *download* (*Download*);
- a **aplicação** - académica (A) ou comercial (C);
- e a **arquitectura** - *Stand alone* (S), *Client/Server* (C/S) ou Processamento Paralelo (PP).

A Tabela 7 classifica as ferramentas segundo o sistema operativo suportado, podendo ser o *Windows*, *Unix* (inclui também o Linux), *MacOS* (Macintosh), ou *Outro* (qualquer outra plataforma de suporte da ferramenta). Convém referir que não serão distinguidas as versões dos sistemas operativos. A Tabela 8 mostra a **conectividade a bases de dados** das ferramentas. O tipo *ASCII* inclui vários formatos próprios das ferramentas e ficheiros de dados separados por tabulações (*tab*) ou por vírgula (também conhecido por formato CSV). Os outros tipos correspondem aos mais representativos, havendo o tipo *Outros* que inclui formatos de bases de dados menos utilizados. Na Tabela 9 caracterizam-se as ferramentas segundo as **técnicas de *Data Mining*** disponibilizados pelas ferramentas. As RNAs são classificadas em *Multilayer Perceptrons* (*MLP*), *Radial Basis Functions* (*RBF*), *Self Organizating Maps* (*SOM*) e uma classe (*NN*) que engloba outros tipos de redes neuronais e ferramentas cujo tipo de rede não é especificado. A caracterização das ferramentas engloba também as técnicas de Máquinas de Vectores de Suporte (*SVM*) e Árvores de Decisão/Regressão. Finalmente, na Tabela 10 especificam-se as ferramentas segundo os **objectivos de *Data Mining***

implementados pelas ferramentas, nomeadamente: Classificação, Regressão, Previsão, e Segmentação.

Por observação directa da Tabela 6 pode verificar-se que a maioria das ferramentas é disponibilizada na versão final e numa versão *demo*, ou então, na versão totalmente funcional para *download*. No que diz respeito ao tipo de licença, verifica-se que a maioria é do tipo comercial (Figura 12). Por observação da Figura 13 verifica-se que, embora as ferramentas de aplicação exclusiva na área académica se destaquem, as ferramentas de aplicação exclusivamente comercial e as ferramentas de aplicação mista estão em maioria. Também as ferramentas do tipo *Stand alone* se destacam por figurarem em maior número (Figura 14)

A Tabela 7 mostra, claramente, que o sistema operativo que mais ferramentas suporta é o *Windows*, sendo que cerca de metade das ferramentas suportam somente esta plataforma. Tal não é surpreendente uma vez que o *Windows* domina claramente o mercado mundial dos sistemas operativos de uso pessoal. Verifica-se pelo gráfico da Figura 15 que também o *Unix* e o *Macintosh* são sistemas suportados em exclusivo por algumas ferramentas, embora tenham pouca expressão.

Na Tabela 8 mostra-se a conectividade das ferramentas, sendo esta tabela complementada pela Figura 16, onde se apresenta um gráfico com a distribuição das ferramentas pelos vários tipos de conectividade a bases de dados. Claramente o tipo mais comum é o *ASCII*, no entanto, os tipos *ODBC*, *MY SQL* e *MS Excel* também têm expressão significativa. No gráfico da Figura 17 pode-se ver a distribuição das ferramentas pelas diversas técnicas que implementam. As RNAs são as mais representadas, mas o número de ferramentas que implementam apenas RNAs é pouco superior ao número de ferramentas que implementam apenas MVSs. No gráfico da Figura 18 está patente a distribuição dos vários tipos de RNAs. Pode-se verificar que só há ferramentas a implementar exclusivamente uma técnica na classe *NN* e *SOM*, mas como para a classe *NN* não foi possível esclarecer o tipo de RNAs implementado, fica em aberto a possibilidade de haver algumas que implementem exclusivamente redes do tipo *MLP* ou *RBF*. A Figura 19 apresenta um gráfico com a distribuição das técnicas pelas ferramentas analisadas. Constata-se que as ferramentas que implementam exclusivamente RNAs ou MVSs são quase metade. Além disso, as aplicações que

implementam RNAs, MVSs e Árvores de Decisão/Regressão (estando assim habilitadas para fazer uma comparação proposta neste trabalho), não chegam a um quarto do número total de ferramentas.

Na Figura 20 apresenta-se a distribuição de ferramentas pelos vários objectivos. Note-se que só a **classificação** tem ferramentas exclusivas.

Tabela 5 – Os sítios Internet das ferramentas analisadas.

Ferramenta	URL
Alyuda NeuroIntelligence	www.alyuda.com
BrainMaker	www.calsci.com
BSVM	www.csie.ntu.edu.tw
Clementine	www.spss.com/clementine/
DTREG	www.dtreg.com
EQUBITS Foresight(tm)	www.equbits.com
EWA Systems	www.ewasystems.com
GhostMiner	www.fqs.pl
Gist	microarray.cpmc.columbia.edu/gist/
Gornik	statconsulting.com.pl/index_en.html
Insightful Miner	www.insightful.com
Kernel Machines (Várias)	www.kernel-machines.org
Knowledge Miner	www.knowledgeminer.com
KXEN	www.kxen.com
LIBSVM	www.csie.ntu.edu.tw/~cjlin/libsvmtools/
MATLAB Neural Net Toolbox	www.mathworks.com/products/neuralnet/
MCubiX from Diagnos	www.diagnos.ca
MemBrain	www.membrain-nn.de/main_en.html
NeuralWorks Predict	www.neuralware.com
NeuroSolutions	www.nd.com
NeuroXL	www.neuroxl.com/
IPNNL Software	www-ee.uta.edu/eeweb/IP
Oracle Data Mining (ODM)	www.oracle.com
Orange	www.ailab.si/orange/
pcSVM	www.procoders.net/en/Procoders/open_source/pcSVM
R	http://www.r-project.org/
SAS Enterprise Miner	http://www.sas.com/technologies/analytics/datamining/
StarProbe	www.roselladb.com/starprobe.htm
STATISTICA Neural networks	www.statsoft.com/products/stat_nn.html
SvmFu 3	five-percent-nation.mit.edu/SvmFu/
SVM-light	svmlight.joachims.org
TANAGRA	chirouble.univ-lyon2.fr/~ricco/tanagra/index.html
HhinkAnalitics	www.thinkanalytics.com
Tiberius	www.philbrierley.com
Weka	www.cs.waikato.ac.nz/ml/weka/
XLMiner	www.xlminer.net

Tabela 6 – Caracterização das ferramentas segundo as características gerais.

Ferramenta	Final/ Beta	Licença	*Demo/ Download*	Académico/ Comercial	Arquitectura (S,C/S,PP)
Alyuda Neuro Inteligence	F	C	S	C	S
BrainMaker	F	C	N	A/C	S
BSVM	F	F	S	A	S
Clementine	F	C	N	C	S/CS
DTREG	F	C	S	A/C	S
EQUBITS Foresight (tm)	F	C	S	A/C	S
EWA Systems	F	C	N	A/C	S/CS
GhostMiner	F	C	N	A/C	S
Gist	F	F	S	A	S
Gornik	F	C	N	C	S/CS
Insightful Miner	F	C	S	A/C	S/CS
Kernel Machines	F	F	S	A	S
Knowledge Miner	F	C	S	A/C	S
KXEN	F	C	N	C	S/CS
LIBSVM	F	F	S	A	S
MATLAB NN Toolbox	F	C	S	A	S
MCubiX from Diagnos	F	C	N	C	S
MemBrain	F	F	S	A	S
NeuralWorks Predict	F	C	S	C	S
NeuroSolutions	F	C	S	A/C	S/CS
NeuroXL	F	C	N	C	S
IPNNL Software	B	F	S	A	S
Oracle Data Mining	F	C	S	C	S,CS,PP
Orange	F	F	S	A	S
pcSVM	B	P	S	A	S
R	F	P	S	A	S
SAS Enterprise Miner	F	C	S	A/C	CS
StarProbe	F	C	S	A/C	S/CS
STATISTICA NN	F	C	S	A	S/CS
SvmFu 3	B	P	S	A	S
SVM-light	F	F	S	A	S
TANAGRA	F	F	S	A	S
HhinkAnalitics	F	C	N	C	CS
Tiberius	F	C	S	A/C	S/CS
Weka	F	P	S	A	S
XLMiner	F	C	S	A/C	S

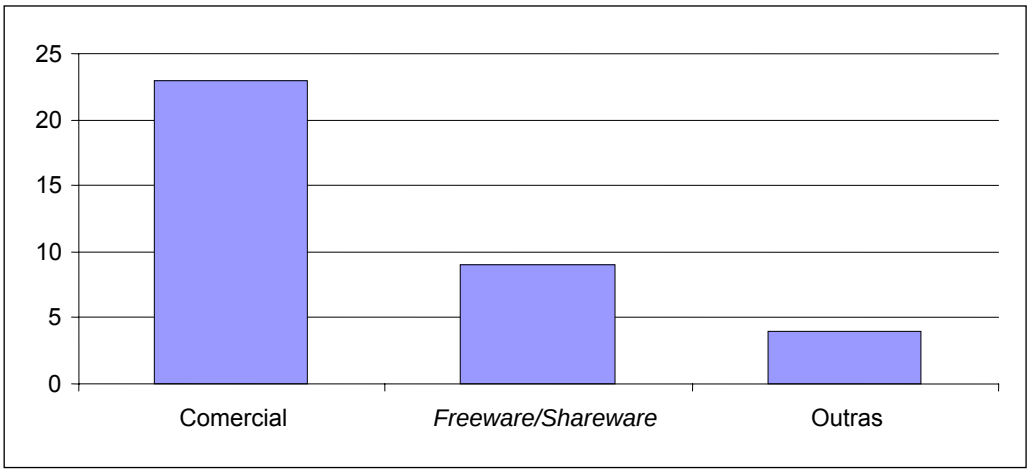

Figura 12 – Caracterização das ferramentas segundo o tipo de licença.

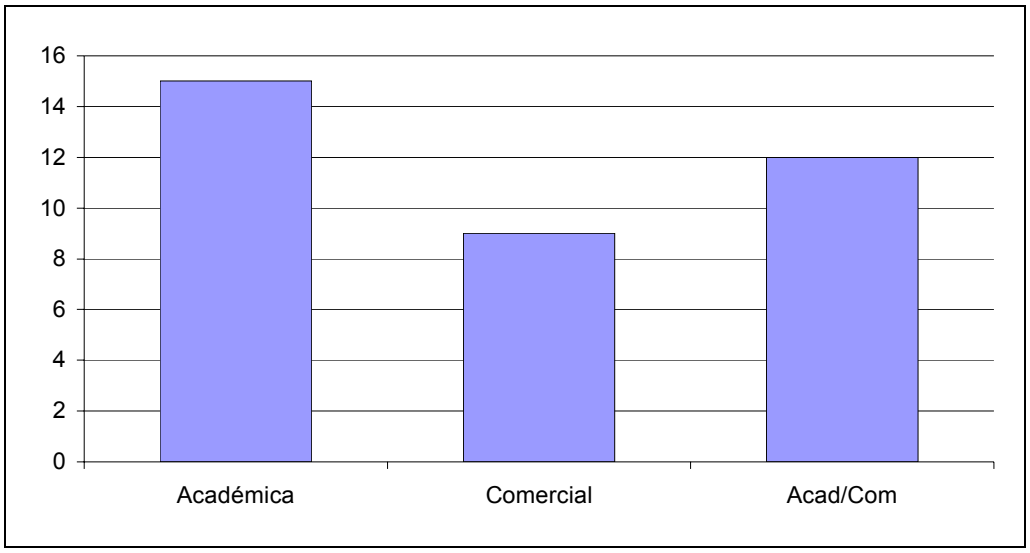

Figura 13 – Caracterização das ferramentas segundo a aplicação.

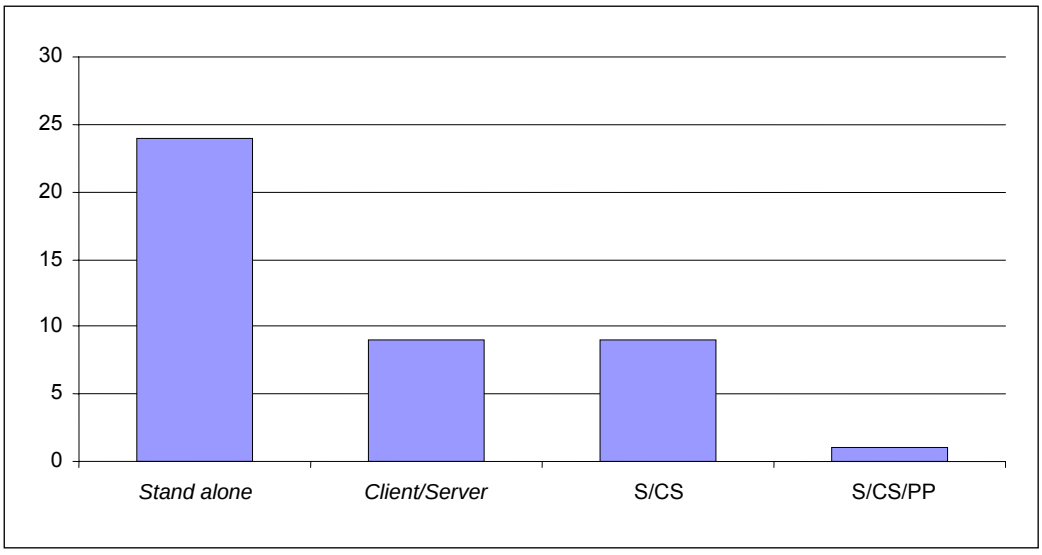

Figura 14 – Caracterização das ferramentas segundo a arquitectura.

Tabela 7 – Caracterização das ferramentas segundo o tipo de sistema operativo.

Ferramenta	Tipo de Sistema Operativo			
	Windows	Unix	Mac	Outro
Alyuda NeuroIntelligence	√			
BrainMaker	√		√	
BSVM	√			
Clementine	√	√		
DTREG	√			
EQUBITS Foresight(tm)	√			
EWA Systems				
GhostMiner	√			
Gist		√		
Gornik	√	√		
Insightful Miner	√			SOLARIS
Kernel Machines (Várias)	√			
Knowledge Miner	√		√	
KXEN	√			
LIBSVM	√	√		
MATLAB Neural Net Toolbox	√	√	√	SOLARIS
MCubiX from Diagnos	√			
MemBrain	√			
NeuralWorks Predict	√	√		
NeuroSolutions	√	√		SOLARIS, IRIX, AIX
NeuroXL				Excel
IPNNL Software	√			
Oracle Data Mining (ODM)	√	√		JAVA
Orange	√	√	√	
pcSVM		√		
R	√	√		
SAS Enterprise Miner	√	√		SOLARIS
StarProbe	√	√	√	SOLARIS
STATISTICA Neural networks				
SvmFu 3		√		SOLARIS
SVM-light	√	√		SOLARIS
TANAGRA	√			
HhinkAnalitics				
Tiberius	√			
Weka				JAVA
XLMiner	√			

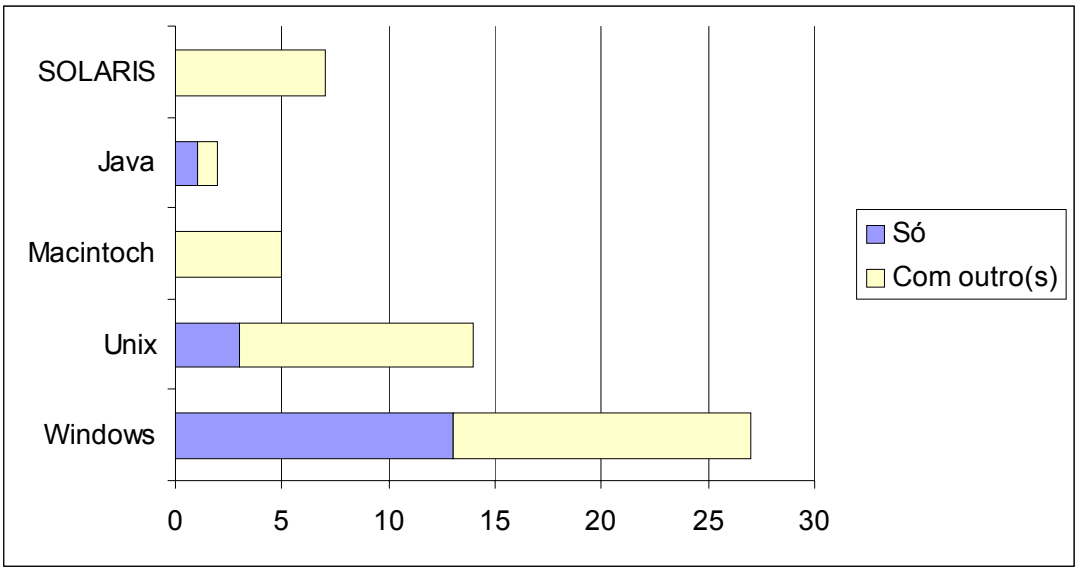

Figura 15 – Caracterização das ferramentas segundo o(s) sistema(s) operativo(s) suportado(s).

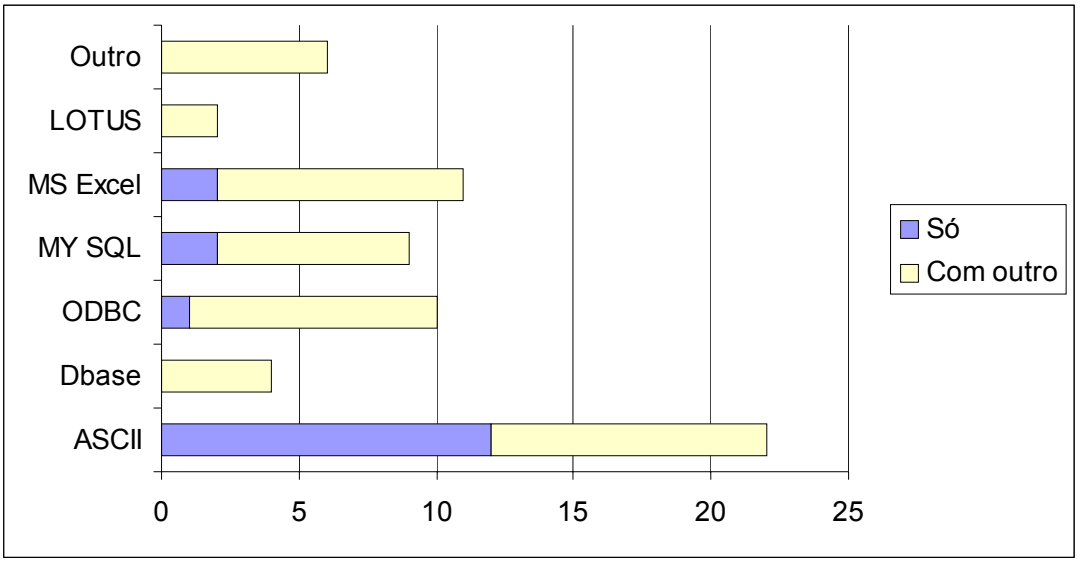

Figura 16 – Conectividade das ferramentas.

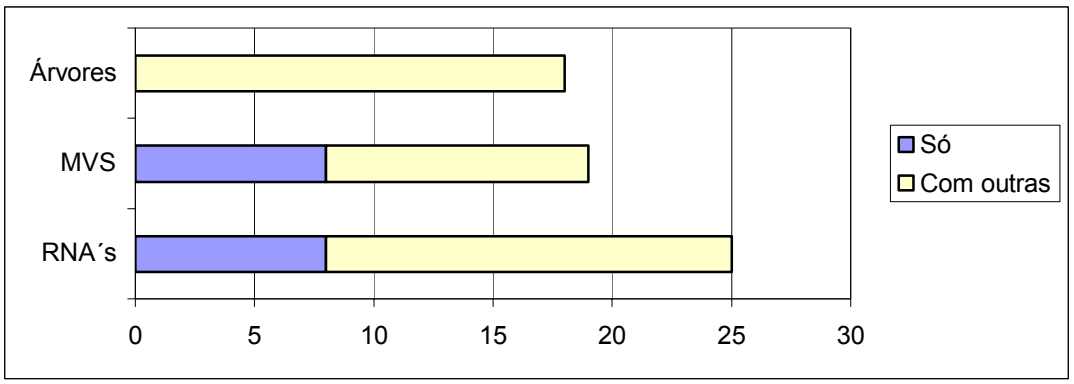

Figura 17 – Distribuição das ferramentas pelas várias técnicas que implementam

Tabela 8 – Caracterização das ferramentas segundo a conectividade a bases de dados.

Ferramenta	Tipo de Base de Dados						
	ASCII	Dbase	ODBC	MY SQL	MS Excel	LOTUS	Outro
Alyuda NeuroIntelligence	√						
BrainMaker	√	√			√	√	
BSVM	√						
Clementine	√		√		√	√	Informix, Oracle, Sybase
DTREG							
EQUBITS Foresight (tm)	√						
EWA Systems				√			
GhostMiner	√		√	√	√		
Gist	√						
Gornik	√		√	√	√		
Insightful Miner	√	√	√	√	√		SAS,SPSS, Sybase
Kernel Machines							
Knowledge Miner							
KXEN				√			
LIBSVM	√						
MATLAB NN Toolbox							
MCubiX from Diagnos	√		√	√	√		Access, freetext, imagens
MemBrain							
NeuralWorks Predict	√						
NeuroSolutions	√				√		Access
NeuroXL					√		
IPNNL Software	√						
Oracle Data Mining	√	√	√	√	√		Oracle, Sybase
Orange	√						
pcSVM							
R	√						
SAS Enterprise Miner			√				
StarProbe			√				JDBC
STATISTICA NN							
SvmFu 3	√						
SVM-light	√						
TANAGRA	√						
HhinkAnalitics		√		√			
Tiberius	√		√	√	√		
Weka	√						
XLMiner					√		

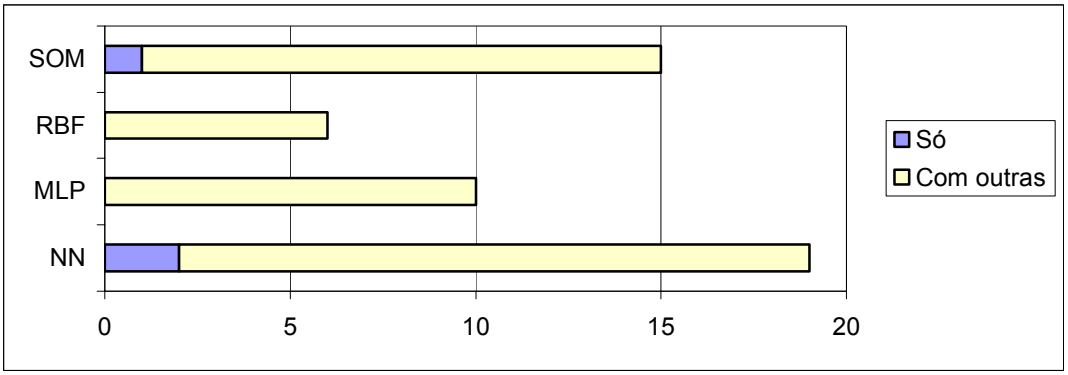

Figura 18 – Distribuição das ferramentas pelos tipos de RNAs que implementam.

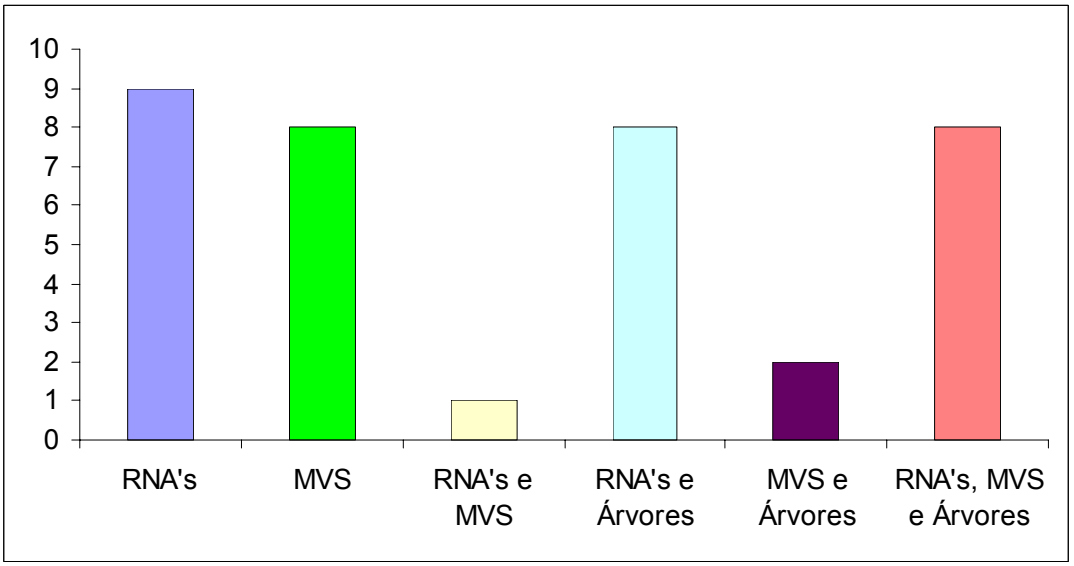

Figura 19 – Distribuição das técnicas pelas ferramentas analisadas.

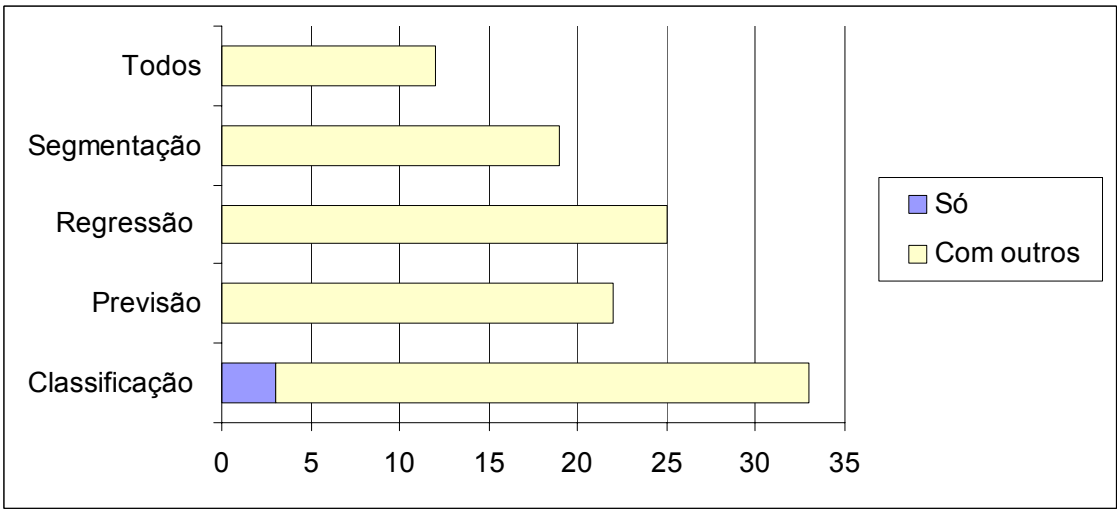

Figura 20 – Distribuição das ferramentas pelos objectivos de *Data Mining* que implementam.

Tabela 9 – Caracterização das ferramentas segundo as técnicas de *Data Mining*.

Ferramenta	RNAs				Outros	
	NN	MLP	RBF	SOM	SVM	Árvores
Alyuda NeuroIntelligence	√					
BrainMaker	√					
BSVM					√	
Clementine		√	√	√	√	√
DTREG					√	√
EQUBITS Foresight(tm)					√	
EWA Systems	√			√	√	√
GhostMiner	√				√	√
Gist					√	
Gornik		√		√	√	√
Insightful Miner		√				√
Kernel Machines	√		√		√	
Knowledge Miner				√		
KXEN					√	
LIBSVM					√	
MATLAB NN Toolbox		√	√	√		
MCubiX from Diagnos	√					√
MemBrain	√	√		√		
NeuralWorks Predict	√			√		
NeuroSolutions	√			√		
NeuroXL	√			√		
IPNNL Software		√		√		
Oracle Data Mining					√	√
Orange	√			√	√	√
pcSVM					√	
R	√	√	√	√	√	√
SAS Enterprise Miner	√			√		√
StarProbe	√					√
STATISTICA NN		√				√
SvmFu 3					√	
SVM-light					√	
TANAGRA	√	√	√	√		√
HhinkAnalitics	√					√
Tiberius	√				√	√
Weka	√	√	√	√	√	√
XLMiner	√					√

Tabela 10– Caracterização das ferramentas segundo os objectivos de *Data Mining*.

Ferramenta	Objectivos implementados			
	Classificação	Previsão	Regressão	Segmentação
Alyuda NeuroIntelligence	√	√		
BrainMaker	√	√		
BSVM	√		√	
Clementine	√	√	√	√
DTREG	√	√	√	
EQUBITS Foresight(tm)	√	√	√	
EWA Systems	√		√	√
GhostMiner	√			√
Gist	√			
Gornik	√			√
Insightful Miner	√	√	√	√
Kernel Machines (Várias)	√		√	
Knowledge Miner	√	√		
KXEN	√	√	√	√
LIBSVM	√		√	
MATLAB Neural Net Toolbox	√	√	√	
MCubiX from Diagnos	√	√	√	
MemBrain				
NeuralWorks Predict		√		√
NeuroSolutions	√	√	√	√
NeuroXL	√	√	√	√
IPNNL Software	√		√	
Oracle Data Mining (ODM)	√	√	√	√
Orange	√	√	√	√
pcSVM	√			
R	√	√	√	√
SAS Enterprise Miner	√	√	√	√
StarProbe	√	√	√	√
STATISTICA Neural networks	√	√		√
SvmFu 3	√			
SVM-light	√		√	
TANAGRA			√	√
HhinkAnalitics	√		√	√
Tiberius	√	√	√	
Weka	√	√	√	√
XLMiner	√	√	√	√

4.3. Sumário

Dada a enorme variedade de ferramentas de *Data Mining*, procurou-se saber quais as mais utilizadas. O sítio www.kdnuggets.com [50] procura responder a esta questão, mas não o faz com rigor estatístico. Nenhuma referência foi encontrada sobre critérios de selecção que permitissem responder a esta questão. Por conseguinte, foi realizado um

levantamento inicial sem critérios que produziu um elevado número de ferramentas (superior a 150). Depois, aplicou-se o critério de que as ferramentas teriam que implementar ou RNAs ou MVSs, técnicas alvo de experiências nesta dissertação, reduzindo o rol de ferramentas para 36.

A caracterização das ferramentas que se realizou teve como critérios os seguintes: **características gerais, conectividade a bases de dados, técnicas de *Data Mining,*** e **objectivos de *Data Mining.*** Os gráficos e tabelas apresentados permitem resumir a caracterização das ferramentas pelas suas características mais representativas.

Quanto às **características gerais**, as ferramentas de *Data Mining* caracterizam-se por estarem disponíveis para o sistema operativo *Windows* na versão final e numa versão *demo*, ou então, na versão totalmente funcional para *download*, versão essa, do tipo *Stand alone*. Caracterizam-se também, por terem licença comercial e, essencialmente, aplicação comercial. Quanto à **conectividade a bases de dados**, as ferramentas caracterizam-se por usarem principalmente dados em formato *ASCII*. No que diz respeito às **técnicas de *Data Mining*** implementadas, caracterizam-se por disponibilizarem maioritariamente RNAs e, embora as MVSs e as Árvores de Decisão/Regressão também estejam bem representadas, escasseiam ferramentas que implementem as três técnicas (ver Tabela 11). Finalmente, quanto aos **objectivos de *Data Mining***, as ferramentas caracterizam-se por implementarem vários, com ligeiro destaque para a maior representatividade do objectivo Classificação.

Capítulo 5

Experiências

Neste capítulo serão utilizadas as técnicas de MVSs, RNAs e Árvores de Decisão/Regressão, na extracção de conhecimento em dados do mundo real. Assim, serão efectuadas experiências em tarefas de classificação e de regressão com o objectivo de fazer uma comparação de resultados (*benchmarking*).

5.1. Ferramenta a Utilizar

Visto haver uma enorme oferta de ferramentas de *Data Mining*, naturalmente tem que se proceder a uma selecção de modo a tornar viável uma comparação de técnicas. Os critérios de escolha da ferramenta passam pelo facto da mesma ter que implementar RNAs, MVSs e Árvores aplicáveis à classificação e à regressão. Por outro lado, a ferramenta deveria ter uma interface simples, passível de ser experimentada por utilizadores não especializados.

Tabela 11 – Ferramentas que implementam RNAs, MVSs e Árvores de Decisão/Regressão.

Ferramenta	RNAs				Outras	
	NN	MLP	RBF	SOM	MVSs	Árvores
Clementine		√	√	√	√	√
Orange	√			√	√	√
R	√	√	√	√	√	√
Tiberius	√				√	√
Weka	√	√	√	√	√	√

Na Tabela 11, pode-se verificar que não são muitas as ferramentas passíveis de serem utilizadas nas experiências pretendidas e algumas destas possuem, ainda, particularidades que as tornam impróprias para as experiências. No caso da ferramenta Clementine, por ser comercial e não disponibilizar versão de demonstração, esta não poderá ser utilizada. Em situação idêntica encontra-se o Tiberius que, embora disponibilize versão de demonstração, é demasiado limitada para os propósitos das experiências que se pretendem realizar. Por sua vez, a ferramenta Orange possui uma interface

gráfica deveras interessante (Figuras 21 e 22). No entanto, infelizmente ainda não possui MVSs adequadas a casos de regressão. Assim, restam as ferramentas R e Weka.

Convém referir que a ferramenta R tem uma interface via linha de comandos, o que não é propriamente adequada à sua utilização por utilizadores não especializados. Contudo, esta ferramenta será escolhida, dado que existe um reduzido leque de ferramentas disponíveis para a comparação das técnicas segundo os critérios já referidos.

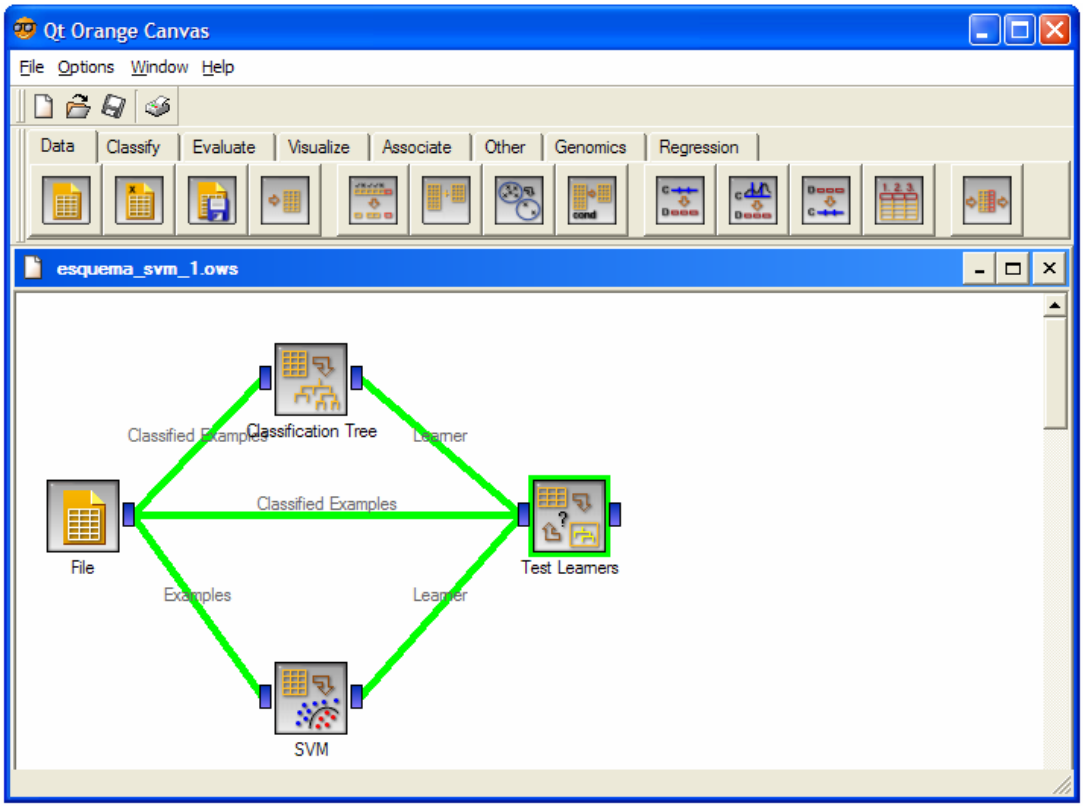

Figura 21 – Comparação de MVSs e Árvores de Decisão/Regressão no caso Bupa (Orange).

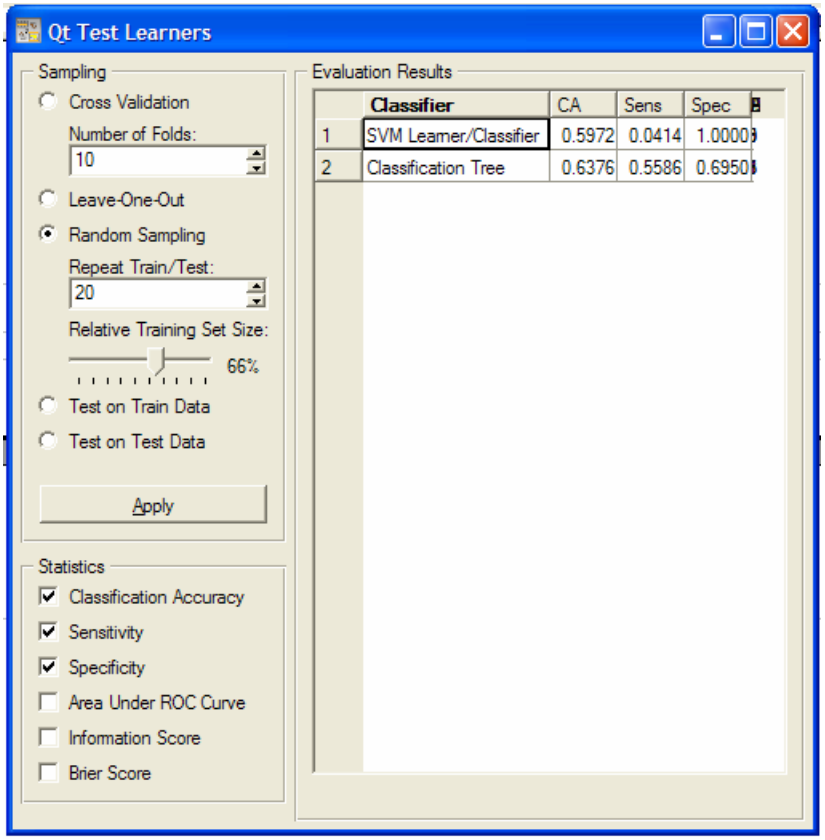

Figura 22 – Interface de teste do desempenho de técnicas de *Data Mining* (Orange).

A ferramenta Weka implementa as técnicas das RNAs, MVSs e Árvores de Decisão/Regressão, podendo estas ser aplicadas tanto à classificação como à regressão. Como interface utiliza um *GUI* (*Graphic User Interface*), tendo também uma licença não comercial (*freeware*). Alem disso, observando a Tabela 6 constata-se que das ferramentas não comerciais, o Weka é (alegadamente) a ferramenta mais utilizada.

5.2. Dados Utilizados

Os conjuntos de dados foram retirados do repositório UCI [27]. Neste repositório público, encontram-se cerca de uma centena de dados disponibilizados por fontes governamentais e universidades (entre outras), para serem utilizados livremente em investigação. De notar que este repositório tem sido deveras utilizado, a nível mundial, pela comunidade da Aprendizagem Automática/*Data Mining* para testar algoritmos [38]. Além disso, estes dados encontram-se praticamente prontos para o processo de *Data Mining*, isto é, necessitam de pouco trabalho de pré-processamento.

Na escolha dos 14 conjuntos de dados (*datasets*) procurou-se escolher somente problemas que se adequassem à classificação ou à regressão. Também existiu o cuidado de escolher dados com características variadas, ou seja, com diferentes dimensões (entre menos de 200 instâncias até mais de 2000 exemplos), passando por valores em falta e atributos variados. Nos parágrafos seguintes estão descritos em pormenor os problemas que foram utilizados nas experiências. Os primeiros sete destinam-se à classificação, sendo que os restantes sete à regressão.

- *Agaricus-lepiota* – Doado por Jeff Schlimmer em 1987, contém descrições de cogumelos pertencentes à base de dados de *Audubon Society Field Guide to North American Mushrooms* (1981). Com 8124 instâncias, 22 atributos nominais[8] que descrevem os cogumelos, e um atributo de saída que classifica cada um dos cogumelos em "comestível" (51,8%) ou "venenoso" (48,2%). Faltam-lhe 2480 valores, todos pertencentes ao atributo número 11.

- *Balance-scale* – Doado por Tim Hume em 1994, possui dados gerados que se referem à posição do ponteiro de uma balança: à esquerda (46,8%), à direita (46,8%) ou equilibrado (7,84%). Tem 625 instâncias, quatro atributos numéricos e um nominal para classificação da posição do ponteiro. Não existem valores omissos.

- *Bupa* - Richard S. Forsyth doou este *dataset* em 1990, sendo criado pelo *BUPA Medical Research Ltd*. Os dados consistem em análises ao sangue em busca de indicadores relacionados com problemas de fígado derivados de abuso no consumo de álcool. Tem 345 instâncias e sete atributos, sendo seis numéricos e um nominal para classificação em duas classes (a primeira com uma prevalência de 42,0% e a outra com os restantes 58,0%). Não existem valores em falta.

- *House-votes-84* - Jeff Schlimmer em 1987 doou este conjunto com dados retirados do *United States Congressional Voting Records Database 1984*. Inclui dados dos votos para o *U.S. House of Representatives Congressmen*.

[8] Atributo discreto não ordenado com mais de 2 classes.

Possui 435 instâncias, 16 atributos binários (sim ou não) e uma classe de saída dividida em "Republicano" (38,6%) ou "Democrata" (61,4%). Tem diversos valores em falta distribuídos ao longo de todos atributos.

- *Ionosphere* – De *Space Physics Group, Applied Physics Laboratory, Johns Hopkins University*, foi doado por Vincent Sigillito em 1989. Contém dados relacionados com a reflexão de emissões de radar na Ionosfera e que se pretendem classificar mediante o facto serem reflectidas ou não (incluindo parcialmente reflectidas). Tem 351 instâncias, 34 atributos contínuos (reais) e um de classificação em "boa reflexão" (65,1%) e "má reflexão" (35,9%). Não faltam valores em nenhuma instância.

- *Pima-indians* – Pertencente ao *National Institute of Diabetes and Digestive and Kidney Diseases*, foi doado em 1990 por Vincent Sigillito. É constituída por vários dados sobre mulheres com pelo menos 21 anos, descendentes dos índios Pima. O objectivo é determinar se têm ou não diabetes. Tem 768 instâncias, oito atributos numéricos e um nominal de classificação positiva (34,9%) ou negativa (65,1%).

- *Post-operative* – Dados sobre pacientes de pós-operatório da *School of Nursing*, *University of Kansas*, doados por Jerzy W. Grzymala-Busse em 1993. O objectivo é decidir para que área deve o paciente ser enviado com base em vários indicadores obtidos por análise, sobre o seu estado. Tem 90 instâncias, cada com nove atributos nominais, incluindo o de classificação: Unidade de Cuidados Intensivos (2,2%), Internamento Geral (26,7%), e Alta (71,1%). Faltam três valores no oitavo atributo.

Os problemas utilizados para classificação estão resumidos na Tabela 12. Os atributos de entrada são descritos como numéricos (num), binários (bin) e nominais (nom). Na tabela consta ainda, o número de instâncias (Nº Inst.), e o método de validação utilizado (10-*fold Cross-validation* - CV[9], ou *holdout Percentage Split* - PS[10]).

[9] Validação cruzada com 10 desdobramentos. Este método é mais lento do que a divisão em casos de treino/teste, pelo que será utilizado somente nos conjuntos de dados de menor dimensão.

[10] Separação simples em casos de treino (66%) e teste (33%), com amostragem aleatória.

Tabela 12 – Sumário dos conjuntos de dados utilizados em classificação.

Conjunto de Dados	Descrição	Atributos			Nº Inst.	Valores em Falta	Método
		Num	Bin	Nom			
Agaricus-lepiota	Toxicidade de cogumelos.	0	0	23	8124	Sim	PS
Balance-scale	Posição do ponteiro de uma balança.	4	0	1	625	Não	PS
Bupa	Perturbações do fígado.	6	0	1	345	Não	CV
House-votes-84	Votos para *U.S. House of Representatives Congressmen.*	0	16	1	435	Sim	CV
Ionosphere	Reflexões de radar na ionosfera.	34	0	1	351	Não	CV
Pima-indians	Mulheres que poderão ter diabetes.	8	0	1	768	Não	PS
Post-operative	Pacientes em pós-operatório.	0	0	9	90	Sim	CV

De seguida, são descritas as tarefas de regressão escolhidas:

- *Abalone* – Dados pertencentes ao *Marine Research Laboratories*, Taroona, *Department of Primary Industry and Fisheries*, Tasmânia, Austrália. Foram doados por Sam Waugh, e contém medições físicas de uma espécie de moluscos. O objectivo é prever a idade dos moluscos (atributo numérico) com bases em seis atributos numéricos e um nominal. Possui 4177 instâncias, e não faltam valores em nenhum dos atributos.

- *Auto-mpg* – Estes dados são oriundos da *Carnegie Mellon University*. Referem-se a características de automóveis. O objectivo é prever o consumo de combustível desses automóveis em circuito citadino. Tem 398 instâncias, cada com oito atributos numéricos, e um nominal. Faltam seis valores a um dos atributos numéricos.

- *Autos* – Também sobre características de automóveis, foram obtidos por várias entidades americanas relacionadas com o sector automóvel. Foram doados por Jeffrey C. Schlimmer em 1987. Pretende-se com estes dados,

prever o valor de mercado do automóvel. Tem 205 instâncias, 26 atributos sendo 16 numéricos e 15 nominais. Faltam valores em alguns atributos.

- *Breast* - Nick Street doou em 1995 estes dados pertencentes à *University of Wisconsin*. São dados sobre pacientes com cancro do peito que pode ser de dois tipos, recorrente ou não recorrente. Pretende-se conhecer o tempo de recorrência ou, no caso de ser não recorrente, o tempo livre de doença. Possui 198 instâncias, um atributo nominal e 34 numéricos, sendo um deles um número de identificação do paciente pelo que será excluído. Há quatro valores em falta num atributo numérico.

- *CPU* – Dados criados por Phillip Ein-Dor e Jacob Feldmesser, *Faculty of Management*; *Tel Aviv University*, Israel, e doados por David W. Aha em 1987. Diz respeito às características da arquitectura de computadores, pretendendo-se fazer a previsão do desempenho de cada um dos computadores. Tem dois atributos nominais e sete numéricos. Não faltam valores nas 209 instâncias.

- *Housing* – Também obtido pela *Carnegie Mellon University*, possui 506 instâncias de dados sobre diversos indicadores que os criadores supõem influenciar o preço de habitações. Pretende-se com estes dados prever o preço das habitações com base em 12 atributos numéricos e um binário. Não existem valores em falta.

- *Servo* – Criado por Karl Ulrich (*MIT*) em 1986, e doado por Ross Quinlan em 1993, contém dados sobre uma simulação de um servomecanismo robótico. É composto por 167 instâncias, cada com dois atributos nominais e dois numéricos, sendo que a variável de saída diz respeito ao tempo de resposta do sistema. Não existem valores em falta.

A Tabela 13 resume os problemas de regressão, separando-os também segundo o método de validação utilizado para a estimação do desempenho do algoritmo.

Tabela 13 – Sumário dos conjuntos de dados utilizados em regressão.

Conjunto de Dados	Descrição	Atributos			Nº Inst.	Valores em Falta	Método de Validação
		Num	Bin	Nom			
Abalone	Idade de moluscos.	7	0	1	4177	Não	PS
Auto-mpg	Combustível consumido por automóveis.	8	0	1	398	Sim	CV
Autos	Preço de automóveis.	16	0	15	205	Sim	CV
Brest	Tempo de recorrência de cancro do peito.	33	0	1	198	Sim	CV
CPU	Desempenho de computadores.	7	0	2	209	Não	CV
Housing	Preço de casas nos subúrbios de Bóston.	12	1	0	506	Não	PS
Servo	Tempo de resposta de servomecanismos.	2	0	2	167	Não	CV

5.3. Descrição da Ferramenta Weka

Não sendo objectivo desta dissertação constituir um manual de utilização de ferramentas de *Data Mining* é, no entanto, oportuno fazer uma pequena descrição das ferramentas que vão ser utilizadas.

Criado pela Universidade de Waikato na Nova Zelândia, o Weka foi desenvolvido na linguagem de programação Java (orientada aos objectos) e implementa uma grande variedade de técnicas [45]. Disponibiliza também, diversos algoritmos de pré-processamento de dados, bem como de análise resultados. O leque de técnicas que implementa permite a utilização da ferramenta em problemas de classificação, regressão e segmentação.

O menu inicial (Figura 23) confronta o utilizador com a escolha de uma interface, de entre quatro possíveis, cada uma com as suas características específicas:

- *Simple CLI* (*Command Line Interface*) - proporciona uma interface de linha de comandos onde se podem executar directamente comandos do Weka.

Embora disponibilize todas as funcionalidades, requer um elevado grau de conhecimento dos comandos que podem ser utilizados.

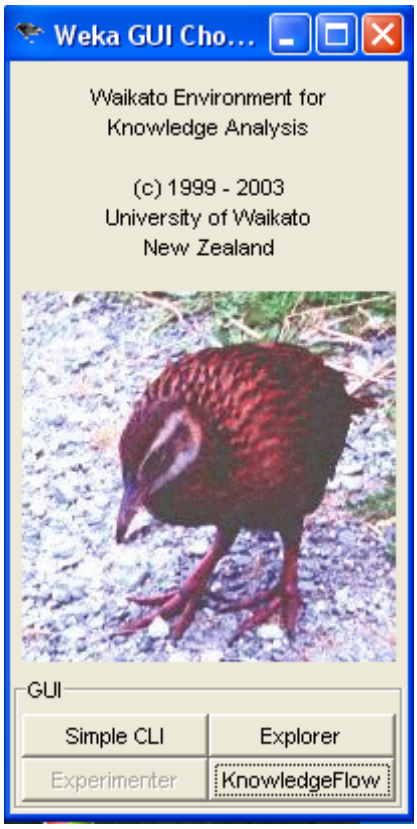

Figura 23 – A janela inicial do Weka (GUI *Chooser*).

- *Explorer* - proporciona um ambiente gráfico de manipulação de dados pela utilização de diversos algoritmos. É a interface mais fácil de utilizar, guiando o utilizador através de menus e formulários, impedindo-o de fazer escolhas não aplicáveis, e apresentando *pop ups* de informação sobre o preenchimento dos vários campos.

- *Knowledge Flow* - permite o desenvolvimento de projectos de *Data Mining* num ambiente gráfico com fluxos de informação (Figura 24). Por outro lado, entre as várias vantagens que possui, é de realçar o *layout* intuitivo, e o facto de permitir o processamento de dados em *batch* ou incrementalmente, o que lhe confere a possibilidade de aplicação a conjuntos de dados de elevada dimensão. Permite, também, o processamento paralelo, em que cada fluxo de dados distinto é processado no seu *thread*.

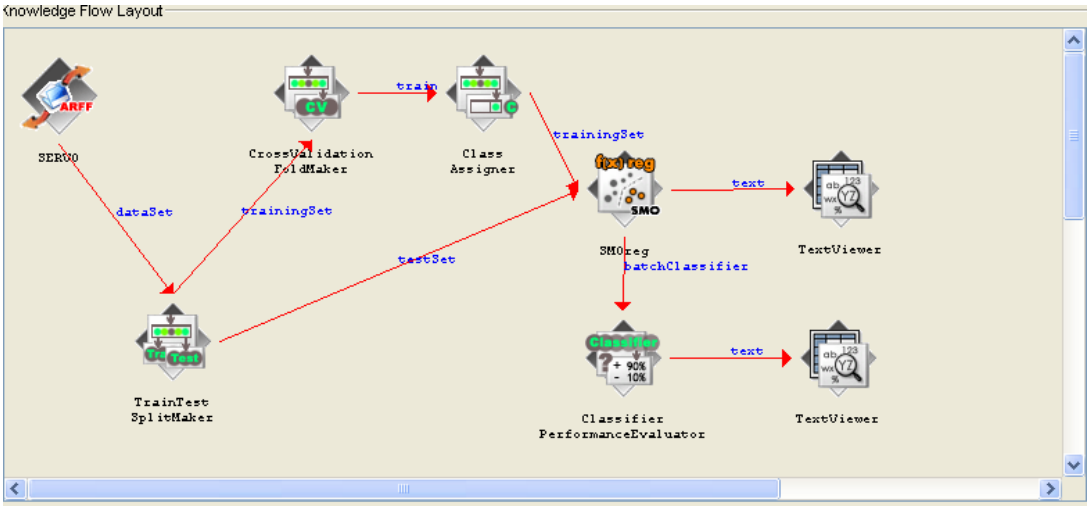

Figura 24 – Exemplo do ambiente *Knowledge Flow* no Weka.

- *Experimenter* - também em ambiente gráfico, permite testar técnicas diferentes em classificação ou regressão, por forma a compará-las. Embora tal também seja possível no *Explorer* e no *Knowledge Flow*, no *Experimenter* é possível escolher diversos conjuntos de dados a serem utilizados numa só experiência, várias técnicas a serem experimentadas, o número de repetições (*runs*) do teste, entre outras escolhas. Depois a experiência é executada sem ser necessária a supervisão do utilizador. Os resultados são guardados em ficheiro para posterior análise. É possível fazer a experiência com computação distribuída através de RMI (*Remote Method Invocation*). Esta é a interface ideal para experiências pelo que será utilizada para levar a cabo as experiências desta dissertação.

A ferramenta Weka, reconhece ficheiros em formato *arff*, próprio do Weka, obrigando o utilizador a formatar os dados de modo a serem reconhecidos como ficheiros desse tipo. Para formatar os dados como *arff*, pode utilizar-se um simples editor de texto como o *MS Word*. Abre-se o ficheiro de dados em causa, e nele introduz-se um cabeçalho que descreve os atributos, como no seguinte exemplo:

```
@RELATION SERVO
@ATTRIBUTE   M   {A,B,C,D,E}
@ATTRIBUTE   S   {A,B,C,D,E}
@ATTRIBUTE   PG REAL
```

@ATTRIBUTE VG REAL
@ATTRIBUTE C REAL

Os dados surgem após a palavra-chave "@DATA". A importação termina gravando o ficheiro com a extensão *arff*. Alternativamente, pode-se utilizar um conversor disponibilizado pelo Weka que permite abrir ficheiros cujos dados estejam separados por vírgula (.csv) ou por *tab*.

O Weka disponibiliza uma vasta variedade de algoritmos de *Data Mining*, entre os quais algoritmos de RNAs, MVSs e Árvores de Decisão/Regressão. O algoritmo de RNAs utilizado nestas experiências implementa uma rede do tipo *MLP*, chamando-se por isso, *MLP*. Pode ser utilizado para classificação ou para regressão. Os algoritmos que implementam MVSs utilizam o método *SMO* (ver 3.3.6), pelo que se chamam *SMO* e *SMOreg* (conforme a sua utilização em tarefas de classificação ou regressão). O segundo possui mais alguns parâmetros que o primeiro (por exemplo o ξ) pelo facto de ser dedicado à regressão. No que diz respeito a Árvores de Decisão/Regressão, é utilizado o *J48* para classificação, que mais não é do que uma implementação do famoso algoritmo *C4.5* (criado por J. Quinlan) para o Weka. Para regressão será utilizado o *REPtree*, que constrói Árvores de Decisão ou de Regressão, fazendo a poda com recurso à técnica de *Reduced Error Pruning* (*REP*).

5.3.1. Classificação com RNAs e MVSs no Weka

Como já se viu, o *Experimenter* é a interface mais adequada a experiências já que é possível escolher várias tarefas e técnicas a serem testadas numa única experiência. Depois, a experiência é executada sem ser necessária intervenção do utilizador, tendo posteriormente acesso aos resultados guardados em ficheiro. É ainda possível fazer a experiência com computação distribuída através de *RMI* (*Remote Method Invocation*), o que pode acelerar consideravelmente as experiências. A possibilidade de fazer experiências com computação distribuída não é de menor importância, pois um elevado número de conjuntos de dados, conjugado com as várias repetições (*runs*) e técnicas, requer um elevado poder computacional.

A interface *Experimenter* fornece ao utilizador três painéis: *Setup, Run* e *Analyse*. A experiência tem início no painel *Setup*, onde será configurada, optando por um de dois modos: simples (*Simple*) ou avançado (*Advanced*). Começando pelo simples, configura-se uma nova experiência (*New*) definindo o ficheiro de destino dos resultados para posterior análise em *Results Destination*. Em *Experiment Type*, escolhe-se entre *cross-validation* ou *train/test split*, pode-se escolher, também, o método de classificação/regressão. Na primeira experiência será escolhido o *cross-validation* com o valor por defeito de 10-*fold*, ou seja, são utilizados 10 desdobramentos. Em *Data sets* adicionam-se os conjuntos pretendidos. Neste caso serão o *Bupa, House-votes-84, Ionospher,* e *Post-operative.*

Em *Iteration Control* define-se o número de vezes que cada técnica será testada, sendo possível mudar a ordem da iteração entre *Data sets first* ou *Algorithms first*. Neste trabalho optou-se por 20 repetições (*runs*) e *Algorithms first*. Finalmente escolheram-se as técnicas de *Data Mining* (RNAs, MVSs e Árvores de Decisão/Regressão), tendo-se optado, nesta experiência, por manter em todas as definições (parâmetros do algoritmo) sugeridas inicialmente pelo Weka.

O aspecto do *Experimenter,* configurado tal como foi descrito, pode ser observado na Figura 25. As Tabelas 14, 15 e 16 apresentam os nomes dos parâmetros associados a cada técnica, a descrição dos mesmos e valor utilizado por omissão.

Tabela 14 – Parametrização do *MLP*.

Parâmetro	Descrição	Valor por Omissão
GUI	Permite visualizar uma interface GUI para configuração da topologia da rede.	FALSO
autoBuild	Constrói as camadas intermédias da rede.	VERDADEIRO
Debug	Apresentação de informação adicional.	FALSO
decay	Diminui a taxa de aprendizagem: a taxa de aprendizagem de cada iteração é obtida dividindo-a pelo número da iteração.	FALSO
hiddenLayers	Define as camadas intermédias.	(Atributos+classes)/2
LearningRate	Taxa de aprendizagem.	0,3
momentum	Taxa de aprendizagem.	0,2
nominalToBinaryFilter	Converte os atributos nominais em binários.	VERDADEIRO
normalizeAttributes	Normaliza os atributos numéricos.	VERDADEIRO
normalizeNumericClass	Normaliza o atribuído a prever caso seja numérico.	VERDADEIRO
randomSeed	Semente *(seed)* para a geração aleatória dos pesos iniciais das sinapses.	0
Reset	Permite que se reinicie a aprendizagem com uma taxa de aprendizagem menor caso o algoritmo esteja a divergir.	VERDADEIRO
TrainingTime	Número de iterações de treino.	500
validationSetSize	Percentagem dos dados a serem utilizados para validação.	0
validationThreshold	Número de vezes que o erro pode piorar nos dados de validação até terminar o treino.	20

Tabela 15 – Parametrização do *SMO*.

Parâmetro	Descrição	Valor por Omissão
buildLogisticModels	Permite modelação logística.	FALSO
c	Parâmetro C de complexidade.	1
cacheSize	Dimensão da *cache.*	250007
debug	Apresentação de informação adicional.	FALSO
exponent	Expoente do *kernel* polinomial.	1.0E-12
featureSpaceNormalization	Normalização do *feature space.*	FALSO
FilterType	Determina a transformação dos dados.	*Normalize Training Data*
gamma	Valor do parâmetro do *kernel* RBF.	0,01
lowerOrderTerms	Permite polinómios de ordem inferior.	FALSO
numFolds	Número de *folds* para CV em modelos logísticos.	Usa os dados de treino
randomSeed	Semente *(seed)* para geração aleatória de elementos durante o CV.	1
toleranceParameter	Tolerância.	0,001
UseRBF	Escolha do *kernel* RBF.	FALSO

Tabela 16 – Parametrização do *J48*.

Parâmetro	Descrição	Valor por Omissão
binarySplits	Divisão binária em atributos nominais.	FALSO
ConfidenceFactor	Factor de confiança utilizado na poda.	0,25
debug	Apresentação de informação adicional.	FALSO
MinNumObj	Número mínimo de instâncias por folha.	2
numFolds	Determina os dados utilizados para a poda.	3
reducedErrorPruning	Permite optar por *reduced error pruning* ou poda do C.4.5.	FALSO
SaveInstanceData	Opção para guardar informação dos dados.	FALSO
Seed	Semente (*seed*) para gerar aleatoriamente dos índices quando se usa *reduced errror pruning.*	1
subtreeRaising	Permite a *subtree raisin* na poda.	VERDADEIRO
Unpruned	Impede a poda.	FALSO
UseLaplace	Método Laplace na contagem das folhas.	FALSO

Seleccionando o painel *Run* é possível iniciar a experiência (*Start*). Enquanto a experiência decorre é apresentado ao utilizador um relatório do decorrer da mesma. É também possível interromper a experiência a qualquer momento (*Stop*). Quando a experiência terminar os resultados são gravados no ficheiro escolhido e este painel informa o utilizador que a experiência terminou com sucesso. Após o término da experiência é possível fazer a análise dos resultados usando o painel *Analyse*. Em *Experiment* é possível analisar os resultados da experiência que acaba de ser executada.

Alternativamente, pode-se especificar um ficheiro com os resultados. Configura-se o teste (*Configure test*) começando definir o que se pretende nas linhas (*Row keys fields*) e nas colunas (*Column keys fields*). Neste caso, nas linhas aparecerão os conjuntos de dados testados, nas colunas a percentagem de classificações correctas por técnica, segundo a significância definida (*Significance*). Em seguida executa-se a análise em *Perform test*, sendo que os resultados podem ser vistos na Figura 26. O melhor resultado é distinguido por "v", e o pior por "*". Na base da tabela (2ª e 3ª colunas), apresenta-se uma contagem de vezes em que a técnica é melhor, igual ou pior que a técnica da 1ª coluna.

A segunda experiência de classificação utilizou os restantes conjuntos de dados com o método de validação *train/test split*, em que parte dos dados é utilizada para treino (no *Experimenter* o valor por defeito é de 66%), e a outra parte é utilizada para teste. As instâncias para cada uma das partes são escolhidas aleatoriamente. A configuração da experiência pode ser observada na Figura 27.

Os resultados das experiências de classificação estão resumidos na Tabela 17. Nela apresenta-se por técnica, a percentagem de classificações correctas (%C) nos conjuntos de teste, o respectivo desvio padrão (DP), e ainda os tempos de treino e de teste (em segundos). Os melhores resultados encontram-se realçados a negrito.

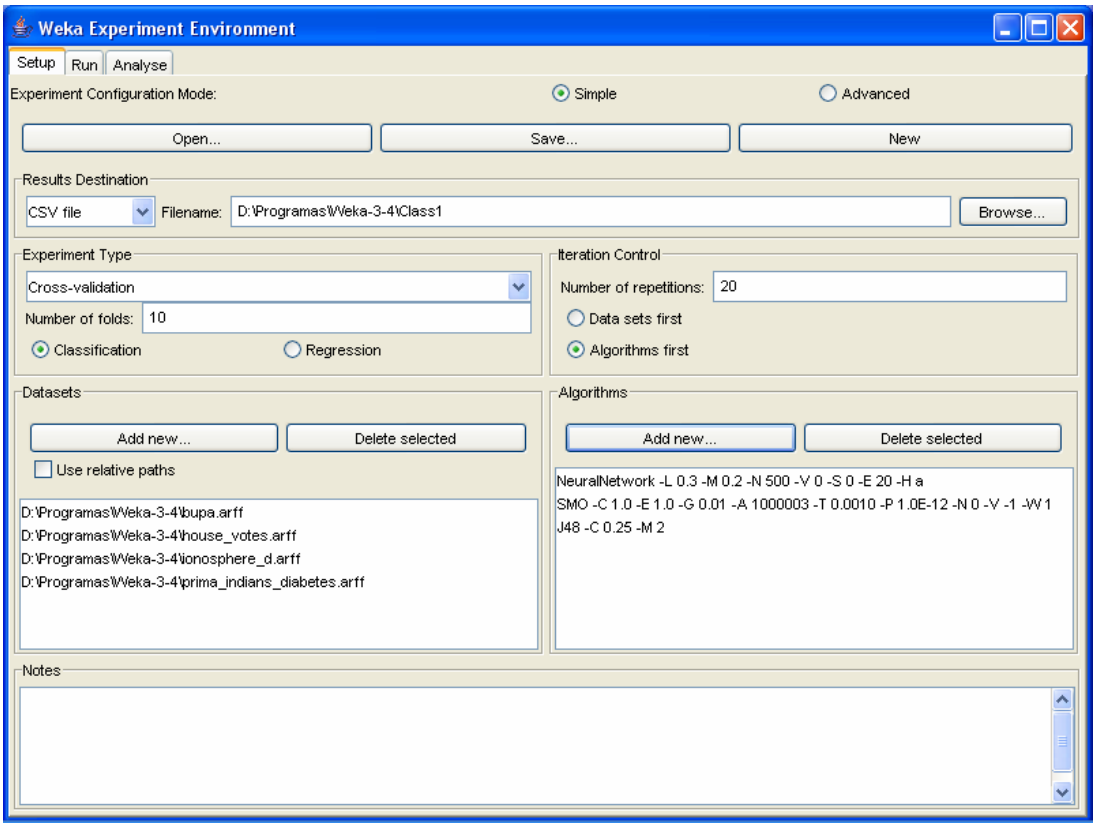

Figura 25 – *Experimenter* configurado para classificação com 10-*fold cross-validation*.

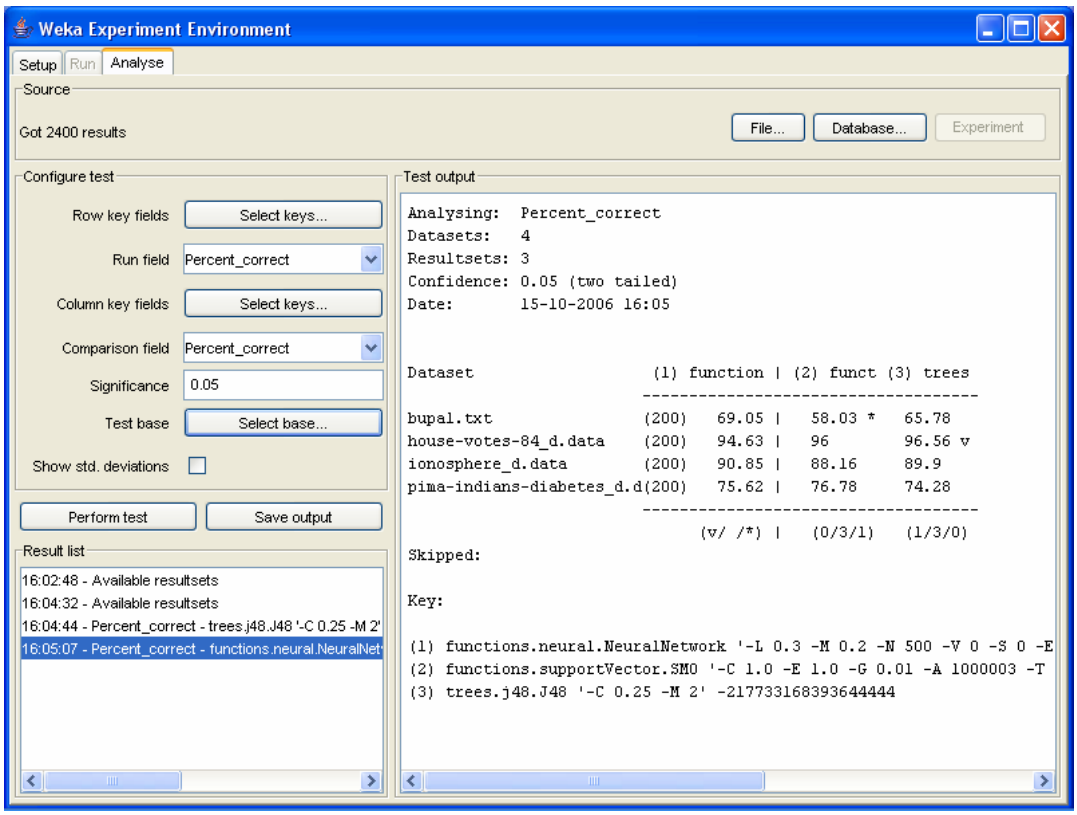

Figura 26 – Resultado da experiência de classificação com 10-*fold cross-validation*.

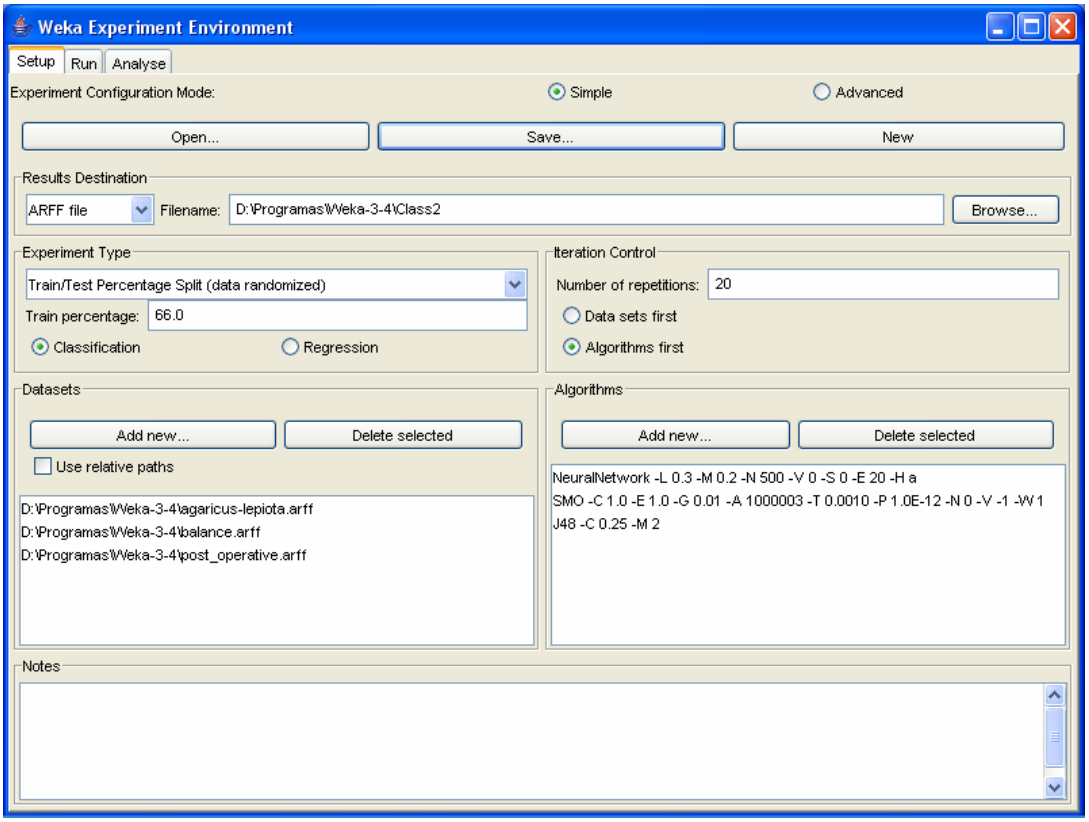

Figura 27 – *Experimenter* configurado para classificação com *train/test split* de 66%.

Tabela 17 – Resumo dos resultados obtidos em classificação.

Conjuntos de Dados	MLP			SMO			J48		
	%C	DP	Treino e Teste	%C	DP	Treino e Teste	%C	DP	Treino e Teste
Agaricus	58,7	1,1	3432,5	**63,9**	0,6	119,7	60,9	0,8	**0,45**
Balance-scale	**90,8**	1,5	1,9	87,5	1,0	0,43	77,8	2,3	**0,02**
Bupa	**69,1**	7,8	1,7	58,0	1,3	0,16	68,8	7,3	**0,02**
House-votes-84	94,6	3,3	7,6	96,0	2,8	0,18	**96,6**	2,7	**0,01**
Ionosphere	**90,9**	4,4	21,3	88,2	5,1	0,22	89,9	4,4	**0,14**
Pima-indians	75,6	5,0	5,2	**76,8**	4,4	0,24	74,3	4,9	**0,05**
Post-operative	54,0	6,8	1,7	67,7	4,1	0,85	**68,7**	3,1	**0,00**
Média	76,2	4,3	496,0	**76,9**	2,8	17,4	76,7	3,7	**0,10**

5.3.2. Regressão com RNAs e MVSs no Weka

A interface *Experimenter* permite que a experiência seja configurada optando por um de dois modos: o simples (*Simple*) e o avançado (*Advanced*). Este último modo é mais difícil de utilizar, pelo que é desaconselhado a utilizadores não especializados. Talvez a maior vantagem na utilização deste modo é a possibilidade de configurar experiências distribuídas. Os resultados destas experiências devem ser enviados para uma base de dados centralizada. Para participar na experiência, cada anfitrião (*host*) tem que ter a

plataforma *Java* instalada, ter acesso aos *datasets* que serão utilizados e estar a executar o servidor de experiências *weka.experi-ment.RemoteEngine*. Se os resultados forem enviados para uma base de dados central, o anfitrião deverá ter também os *drivers* JDBC instalados. A título de exemplo, apresenta-se na Figura 28 o modo avançado, já que não será realizada nenhuma experiência distribuída.

Também em regressão, o conjunto de tarefas foi dividido para duas experiências com métodos de validação distintos. Assim, os conjuntos de dados *Auto-mpg, Autos, Brest, CPU* e *Servo*, foram validados com um 10-*fold cross-validation*, enquanto que os problemas *Abalone* e *Housing* foram validados com um *Train/Test Percentage Split* de 66%. Em todos os problemas foram executadas 20 simulações (*Runs)*. Os parâmetros serão mantidos para a técnica *MLP*. Por sua vez, a configuração base das técnicas *SMOreg* e *REPtree* é descrita nas Tabelas 18 e 19.

Na Tabela 20 resumem-se os resultados obtidos, apresentando-se por técnica o *Root Relative Squared Error* (*RRSE*), bem como o respectivo desvio padrão (DP) e, ainda, os tempos de treino e de teste (em segundos). Os melhores resultados encontram-se realçados a negrito.

Tabela 18 – Parametrização do *SMOreg*.

Parâmetro	Descrição	Valor por Omissão
c	Parâmetro C de complexidade.	1
cacheSize	Dimensão da *cache*.	250007
debug	Apresentação de informação adicional.	FALSO
epsilon	Tolerância dos desvios.	0,001
exponent	Expoente do *kernel* polinomial.	1.0E-12
featureSpaceNormalization	Normalização *do feature space*.	FALSO
FilterType	Determina como os dados serão transformados.	*Normalize Training Data*
gamma	Valor do parâmetro do *kernel* RBF.	0,01
lowerOrderTerms	Permite polinómios de ordem inferior.	FALSO
toleranceParameter	Tolerância.	0,001
UseRBF	Escolha do *kernel* RBF.	FALSO

Tabela 19 – Parametrização do *REPtree*.

Parâmetro	Descrição	Valor por Omissão
debug	Apresentação de informação adicional.	FALSO
MaxDepth	Máxima dimensão da árvore.	Sem limite
minNum	Mínimo peso total por folha.	2
MinVarianceProp	Proporção mínima da variância num nó para que seja efectuada a divisão no caso de Árvores de Regressão.	0,001
NoPruning	Impede a poda.	FALSO
NumFolds	Determina a quantidade de dados utilizados para a poda.	3
seed	Semente (*seed*) para a geração aleatória dos índices quando se usa *reduced error pruning*.	1

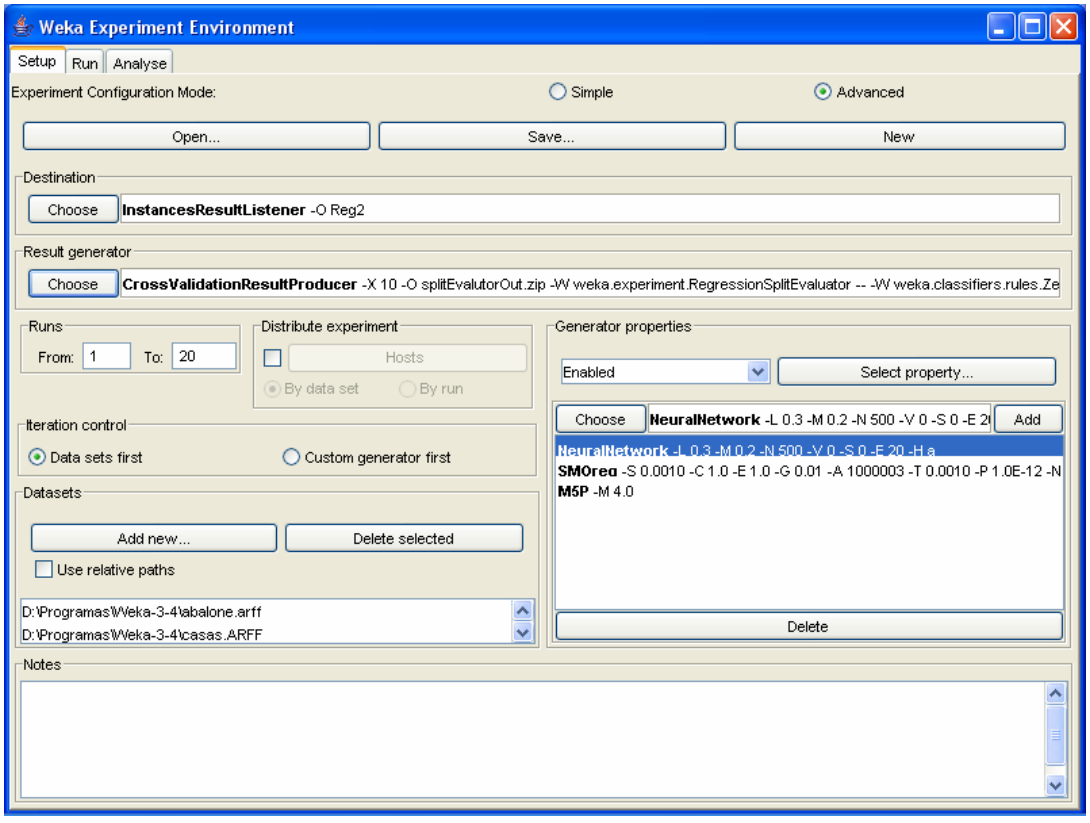

Figura 28 – Aspecto do modo avançado do *Experimenter*.

Tabela 20 – Resumo dos resultados obtidos em regressão.

Conjuntos de Dados	MLP			SMOreg			REPtree		
	RRSE	DP	Treino e Teste	RRSE	DP	Treino e Teste	RRSE	DP	Treino e Teste
Abalone	71,0	7,1	20,4	**69,4**	1,6	52,7	72,5	72,5	**0,11**
Auto-mpg	43,4	9,4	1,8	44,2	5,4	0,4	**43,0**	42,3	**0,01**
Autos	33,6	10,7	48,6	**32,9**	8,9	1,6	48,9	48,9	**0,01**
Brest	175,6	38,7	10,6	**90,0**	14,9	0,2	101,7	101,7	**0,02**
CPU	141,7	99,0	608,6	**28,1**	6,1	0,4	100,0	100,0	**0,00**
Housing	**50,3**	6,4	20,4	55,9	6,1	52,7	54,4	54,4	**0,01**
Servo	**39,9**	18,6	1,5	80,8	8,9	0,2	51,5	51,5	**0,00**
Média	79,4	27,1	101,6	**57,3**	7,4	15,55	67,4	67,4	**0,02**

5.4. Descrição da Ferramenta R

Apresentando uma interface de linha de comandos, o R oferece um conjunto de ferramentas de manipulação e análise estatística de dados. No que diz respeito ao *Data Mining*, disponibiliza diversas técnicas entre os quais RNAs, MVSs e Árvores de Decisão/Regressão. Trata-se de um projecto *open source* que pode ser encontrado em http://www.r-project.org/ [51].

Foi desenvolvido como uma derivação da linguagem estatística *S*, inicialmente para ambientes *Unix/Linux*, embora actualmente esteja disponibilizado em diversas plataformas, tais como o *Windows* ou *Machintoch*. Tem por paradigma a programação orientada a objectos, permitindo que novas características sejam acrescentadas através de pacotes (*packages*). A interface é muitas vezes vista como um factor que acrescenta dificuldade na aprendizagem e utilização do R, mas o ambiente de programação tem como vantagem permitir um grande controlo das várias ferramentas disponibilizadas. Além disso, existem alguns pacotes, em fase de desenvolvimento, que implementam interfaces do tipo GUI, tal como o *Rattle*[11] [51].

O R possui também uma grande facilidade de importação de dados, reconhecendo vários formatos (separados por vírgulas ou espaços) bem como um formato próprio. Permite ainda, diversas formas de visualização de dados e resultados, seja em tabelas ou em gráficos.

[11] Todavia, convém referir que o *Rattle* ainda está numa fase beta em termos do seu desenvolvimento.

5.4.1. Utilização de RNAs e MVSs no R

A utilização de RNAs, MVSs ou Árvores de Decisão/Regressão obriga à instalação dos *packages*, respectivamente, *nnet*, *e1071* e *tree*. No ambiente *Windows,* no menu *Packages* selecciona-se *Install package(s)*, e abrir-se-á uma janela que permite a escolha do servidor através do qual se selecciona e descarrega o ficheiro que contém o *package* desejado (Figura 29). Após a instalação, é necessário o carregamento do *package*, através do mesmo menu, desta vez seleccionando *Load package*.

Cada técnica possui um conjunto de parâmetros que podem ser alterados tendo em vista o controlo da criação dos modelos. Tal como no Weka, optou-se pela utilização dos valores por omissão. No entanto, o *package* de RNAs exige que o parâmetro *size* (número de neurónios intermédios do MLP) seja definido pelo utilizador. Idealmente, este parâmetro deveria ser escolhido através de uma selecção de modelos, onde se testariam diversos valores, escolhendo-se a rede que apresentasse o melhor valor num conjunto de validação. Contudo, este procedimento exige elevados recursos computacionais e, provavelmente, não é de aplicação comum pelo utilizador não especializado. Assim, dentro do espírito deste estudo e a título meramente demonstrativo, serão escolhidos 4 neurónios intermédios. Trata-se de um valor que permite criar RNAs de reduzida dimensão e que, por isso, são menos propícias ao fenómeno de sobreajustamento: i.e., perda de capacidade de generalização. Após algumas experiências preliminares, nos parâmetros *range* e *decay* foram utilizados os valores de 0,2 e 5E-4. As Tabelas 21, 22 e 23 apresentam por técnica, os parâmetros, respectiva descrição e valor por omissão do R, tanto para a classificação como para a regressão.

A utilização das técnicas é feita à custa de linha de comandos, tal como já foi dito. Assim, torna-se necessário criar código que faça a leitura dos dados, implemente a forma de validação (*k-fold cross-validation* ou *percentage split*), crie o modelo utilizando uma das técnicas, e faça o teste do modelo e cálculo de uma métrica sobre os resultados do teste. Nesta dissertação, foi desenvolvido um código R (ver Anexo A), que implementa a validação *10-fold ou percentage split* de 66%, para as diversas técnicas estudadas. De referir que cada experiência foi repetida 20 vezes (*runs*) pelo que no final é calculada a precisão média e o respectivo desvio padrão.

Tabela 21 – Parametrização da função *nnet* (técnica MLP).

Parâmetro	Descrição	Valor por Omissão
Formula	Descrição simbólica do modelo.	Opcional
x	Vector ou matriz de dados.	Opcional
Y	Vector de respostas.	Opcional
weights	Pesos de cada exemplo.	1
size	Número de unidades na camada intermédia.	
data	Dados para treino.	Opcional
subset	Vector de índices dos valores a serem utilizados no treino.	Opcional
na.action	Parâmetro a especificar acção a tomar caso encontrem "*NAs*" (Valores omissos).	na.omit
contrasts	Lista de contrastes.	Opcional
wts	Vectores com os pesos iniciais.	Aleatórios
mask	Vector a indicar quais os parâmetros a serem optimizados.	Todos
linout	Permite escolher saídas lineares.	Logistica
entropy	Permite escolher modelação através de entropia.	Least-squares
softmax	Permite escolher uma modelação probabilística quando existem mais do que 2 classes de saída.	FALSO
censored	Variante *do softmax.*	FALSO
skip	Permite sinapses entre a entrada e a saída.	FALSO
rang	Define o intervalo de valores dos pesos iniciais ([-rang,rang]).	0,7
decay	Parâmetro de redução da complexidade, quanto maior, menor serão os valores dos pesos da MLP.	0
maxit	Número máximo de iterações.	100
trace	Permite seguir o algoritmo de treino.	VERDADEIRO
MaxNWts	Número máximo de pesos.	1000
abstol	Pára se o critério de *fit* do modelo cai abaixo de abstol.	1.0e-4
reltol	Pára caso o algoritmo não consiga reduzir o critério de *fit* pelo menos 1-retol.	1.0e-8

Tabela 22 – Parametrização da função *svm* (técnica MVS).

Parâmetro	Descrição	Valor por Omissão
formula	Descrição simbólica do modelo.	Opcional
data	Dados para treino.	Opcional
x	Vector ou matriz de dados.	Opcional
y	Vector de respostas.	Null
scale	Vector lógico que determina quais as variáveis que sofrerão alteração de escala.	Média=0 e Variância=1
type	Permite escolher o tipo de MVS.	C-*classification* e eps-*regression*
kernel	Configuração dos parâmetros do *kernel.*	"radial"
degree	Parâmetro necessário para o *kernel* polinomial.	3
gamma	Parâmetro necessário para todos os *kernels* excepto linear.	1/Dimensão dos dados
coef0	Parâmetro necessário para *kernels* do tipo polinomial ou *sigmoid.*	0
cost	Custo C.	1
nu	Parâmetro necessário para nu-classification, nu-regression e one-classification.	0,5
class.weights	Pesos das classes.	Null
cachesize	Dimensão da *cache* (em MB).	40
tolerance	Tolerância do critério de paragem.	0,001
épsilon	Parâmetro da função insensitiva.	0,1
Shrinking	Heurística de *shrinking.*	VERDADEIRO
cross	Implementa *k-fold CV*.	0
probability	Parâmetro que permite previsões probabilísticas.	FALSO
subset	Vector de índices dos valores a serem utilizados no treino.	Opcional
na.action	Parâmetro a especificar acção a tomar caso encontrem "*NAs*" (Valores omissos).	na.omit

Tabela 23 – Parametrização da função *tree* (técnica de Árvores de Decisão/Regressão).

Parâmetro	Descrição	Valor por Omissão
formula	Descrição simbólica do modelo.	Opcional
data	Dados para treino.	Opcional
weights	Vector de pesos de cada atributo.	Opcional
subset	Vector de índices dos valores utilizados no treino.	Opcional
na.action	Especificação da acção a tomar caso encontrem "*NAs*".	na.omit
control	Lista a ser usada com a função tree.control.	Opcional
method	Método utilizado na criação da árvore: "recursive.partition" ou "model.frame".	"recursive.partition",
split	Critério de divisão ("deviance", "gini").	c("deviance", "gini"),
model	Retorna o modelo criado.	FALSO
x	Retorna a matriz do modelo.	FALSO
y	Retorna as respostas do modelo.	VERDADEIRO
wts	Retorna os pesos do modelo.	VERDADEIRO

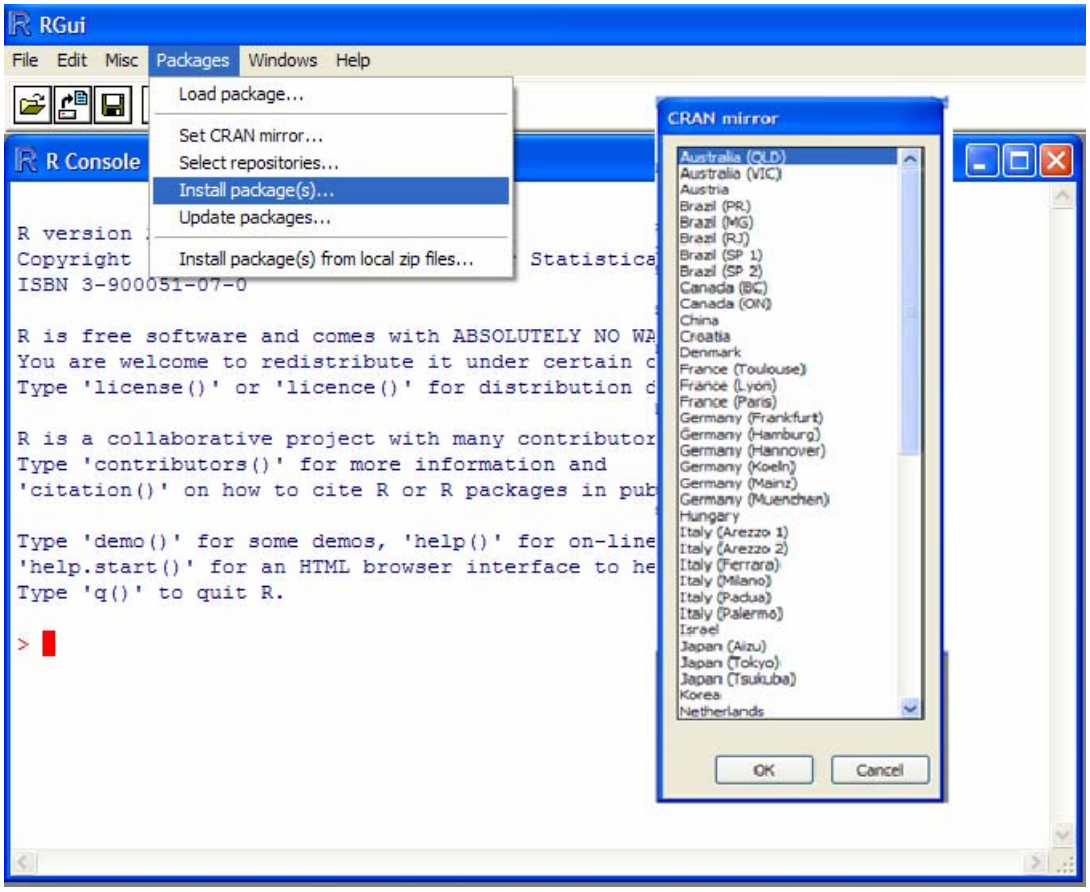

Figura 29 – Aspecto da consola do R no ambiente Windows.

5.4.1.1. Classificação com RNAs e MVSs no R

A Tabela 24 resume os resultados obtidos em classificação com as técnicas de *Data Mining* na ferramenta R.

Tabela 24 – Resumo dos resultados obtidos em classificação com o R.

Conjuntos de Dados	RNAs			MVSs			Árvores		
	%C	DP	Treino e Teste	%C	DP	Treino e Teste	%C	DP	Treino e Teste
Agaricus	99,9	0,0	7,32	**99,9**	0,0	5,55	99,9	0,1	**0,45**
Balance-scale	**90,4**	1,2	2,37	90,3	0,0	1,05	68,7	0,3	**0,02**
Bupa	65,6	2,5	1,05	**66,5**	0,0	0,75	59,8	0,3	**0,02**
House-votes-84	95,1	0,5	3,29	**95,6**	0,0	1,11	95,1	0,0	**0,01**
Ionosphere	85,9	1,8	3,33	**93,1**	0,0	1,09	84,4	0,3	**0,14**
Pima-indians	67,6	5,7	0,24	**76,3**	2,2	0,19	73,4	2,7	**0,05**
Post-operative	60,4	3,4	1,05	**72,5**	0,0	0,37	56,8	1,7	**0,00**
Média	80,7	2,2	2,66	**84,9**	0,3	1,44	76,9	0,7	**0,10**

5.4.1.2. Regressão com RNAs e MVSs no R

Após a aplicação das técnicas de regressão da ferramenta R aos diversos conjuntos de dados, obtiveram-se os resultados que se apresentam resumidos na Tabela 25.

Tabela 25 – Resumo dos resultados obtidos em regressão com o R.

Conjuntos de Dados	RNAs			MVSs			Árvores		
	RRSE	DP	Treino e Teste	RRSE	DP	Treino e Teste	RRSE	DP	Treino e Teste
Abalone	**66,9**	2,0	0,7	71,1	1,0	2,50	76,2	1,7	**0,15**
Auto-mpg	44,4	3,2	1,6	**41,5**	0,0	1,82	56,3	0,0	**0,26**
Autos	100,4	1,0	37,8	**100,0**	0,0	1,93	106,7	0,0	**1,05**
Brest	102,0	2,9	0,8	**95,4**	0,0	1,05	115,0	0,0	**0,46**
CPU	87,8	11,9	1,5	**47,4**	0,0	1,66	102,4	0,0	**0,23**
Housing	86,9	9,9	0,1	**49,9**	4,6	0,09	56,1	4,7	**0,05**
Servo	**46,0**	9,3	0,9	66,5	0,0	0,44	50,0	0,0	**0,17**
Média	76,3	5,8	6,2	**67,4**	**0,8**	1,36	80,4	0,9	**0,34**

5.5. Análise dos Resultados

As Tabelas 26 e 27 resumem os resultados obtidos (média e respectivo desvio padrão) da aplicação das técnicas de *Data Mining* em classificação e regressão. No primeiro caso, foi utilizado como métrica de erro a percentagem de instâncias correctamente classificadas. Portanto, quanto maior for o valor do avaliador melhor é a avaliação do

modelo. Quanto à regressão, foi utilizada a medida *RRSE*. Como esta métrica é relativa ao erro médio, quanto menor for o seu valor, melhor será a técnica. Acrescenta-se também a Tabela 28, que apresenta por técnica, o número de vezes que obteve o melhor resultado, o número de vezes que obteve o segundo melhor resultado e o número de vezes que ficou em último lugar. Para além disso, será utilizada uma análise comparativa adicional, onde se traduz em pontuação os valores obtidos na Tabela 28, ou seja, admite-se que o 1º lugar corresponde a três pontos, o 2º a dois e o 3º a um ponto. A Tabela 29 apresenta o correspondente *ranking*, onde se pode observar que as MVSs obtêm o primeiro lugar, seguidas das RNAs e finalmente pelas Árvores de Decisão/Regressão.

Além destes resultados, as técnicas também podem ser analisadas segundo os tempos de treino e teste. Segundo estes critérios há diversos empates, provavelmente por as ferramentas estudadas não possuírem grande precisão temporal. De qualquer modo é possível visualizar o resultado geral do desempenho das técnicas experimentadas.
Nas Figuras 30 e 31 apresentam-se gráficos que relacionam os valores médios dos critérios de avaliação (tempo de treino e teste, e percentagem de classificações correctas - %C) utilizados para avaliar as técnicas aplicadas a classificação. Verificam-se grandes diferenças nos tempos de treino e de teste, destacando-se as Árvores de Decisão pelos baixos valores de tempo de treino e teste, e as RNAs pelos elevados valores. Também nas Figuras 32 e 33 são apresentados em gráfico os valores médios dos critérios de avaliação (tempo de treino e de teste também e *RRSE*) utilizados para avaliar as técnicas aplicadas a regressão. Pode-se verificar que as Árvores mantêm os melhores resultados ao nível de tempos de teste e de treino, enquanto que as RNAs, os piores resultados.

Tabela 26 – Resumo dos resultados obtidos em classificação.

Conjuntos de Dados	RNAs		MVSs		Árvores	
	R	Weka	R	Weka	R	Weka
Agaricus	99,9	58,7	**99,9**	**63,9**	99,9	60,9
Balance-scale	**90,4**	**90,8**	90,3	87,5	68,7	77,8
Bupa	65,6	**69,1**	**66,5**	58,0	59,8	68,8
House-votes-84	95,1	94,6	**95,6**	96,0	95,1	**96,6**
Ionosphere	85,9	**90,9**	**93,1**	88,2	84,4	89,9
Pima-indians	67,6	75,6	**76,3**	**76,8**	73,4	74,3
Post-operative	60,4	54,0	**72,5**	67,7	56,8	**68,7**
Média	80,7	76,2	**84,9**	**76,9**	76,9	76,7

Tabela 27 – Resumo dos resultados obtidos em regressão.

Conjuntos de Dados	RNAs		MVSs		Árvores	
	R	Weka	R	Weka	R	Weka
Abalone	**66,9**	71,0	71,1	**69,4**	76,2	72,5
Auto-mpg	44,4	43,4	**41,5**	44,2	56,3	**43,0**
Autos	100,4	33,6	**100,0**	**32,9**	106,7	48,9
Brest	102,0	175,6	**95,4**	**90,0**	115,0	101,7
CPU	87,8	141,7	**47,4**	**28,1**	102,4	100,0
Housing	86,9	**50,3**	**49,9**	55,9	56,1	54,4
Servo	**46,0**	**39,9**	66,5	80,8	50,0	51,5
Média	76,3	79,4	**67,4**	**57,3**	80,4	67,4

Tabela 28 – *Ranking* das técnicas.

Técnica	Classificação			Regressão		
	1º	2º	3º	1º	2º	3º
RNAs	4	5	5	4	7	3
MVSs	8	4	2	9	1	4
Árvores	2	5	7	1	6	7

Tabela 29 – *Ranking* por pontuação.

Técnica	Classificação	Regressão
RNAs	27	29
MVSs	**34**	**33**
Árvores	23	22

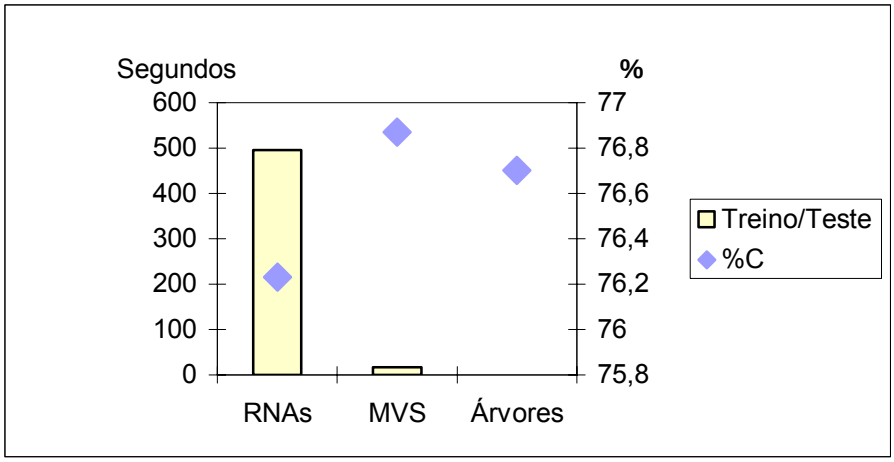

Figura 30 – Percentagem de classificações correctas e tempos de treino/teste (Weka).

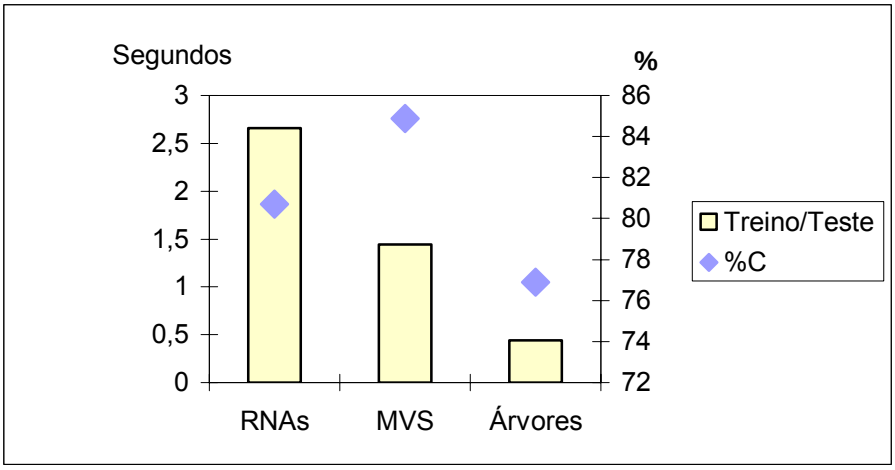

Figura 31 – Percentagem de classificações correctas e tempos de treino/ teste (R).

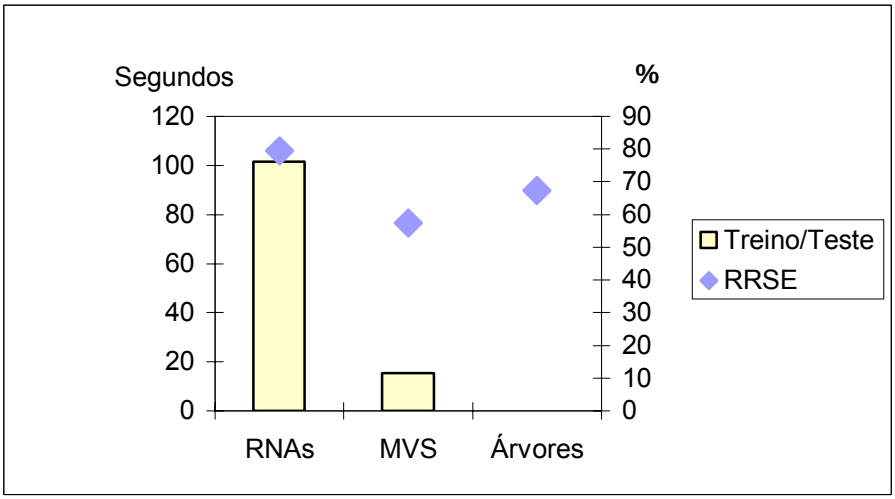

Figura 32 – RRSE e tempos de treino/ teste em regressão (Weka).

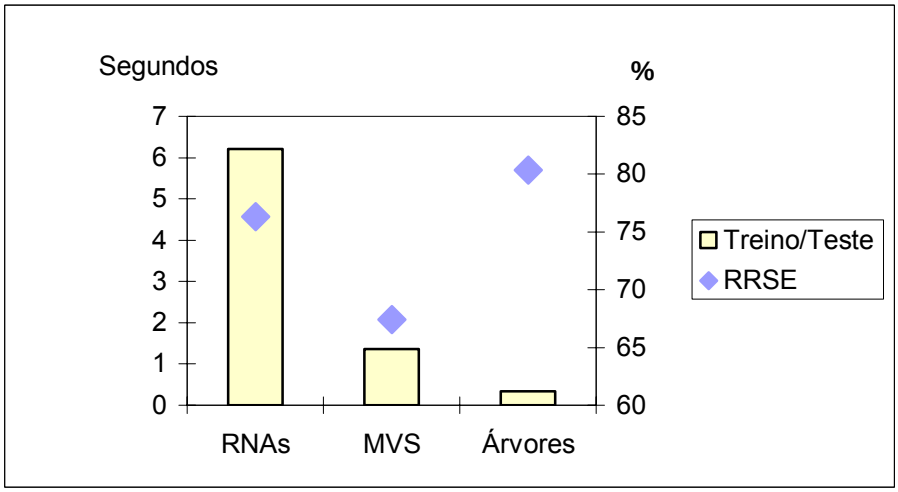

Figura 33 – RRSE e tempos de treino/ teste em regressão (R).

Na Figura 34 e 35 apresentam-se as relações entre os tempos de treino e teste e o número de instâncias nos casos de classificação, respectivamente para o Weka e o R. De igual modo, nas Figuras 36 e 37 apresentam-se as mesmas relações mas nos casos de regressão. Nos casos de classificação verifica-se uma tendência de diminuição dos tempos de treino e teste com a diminuição do número de instâncias. Nos casos de regressão, parece não haver relação entre os tempos de treino e teste e o número de instâncias.

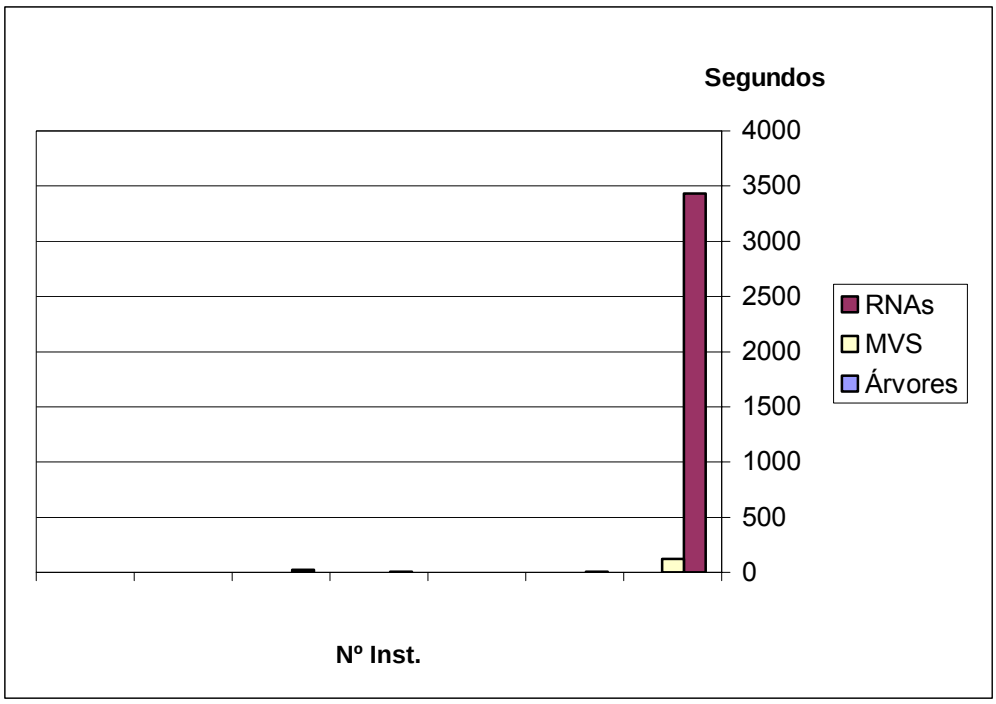

Figura 34 – Tempos de treino/teste e número de instâncias em classificação (Weka).

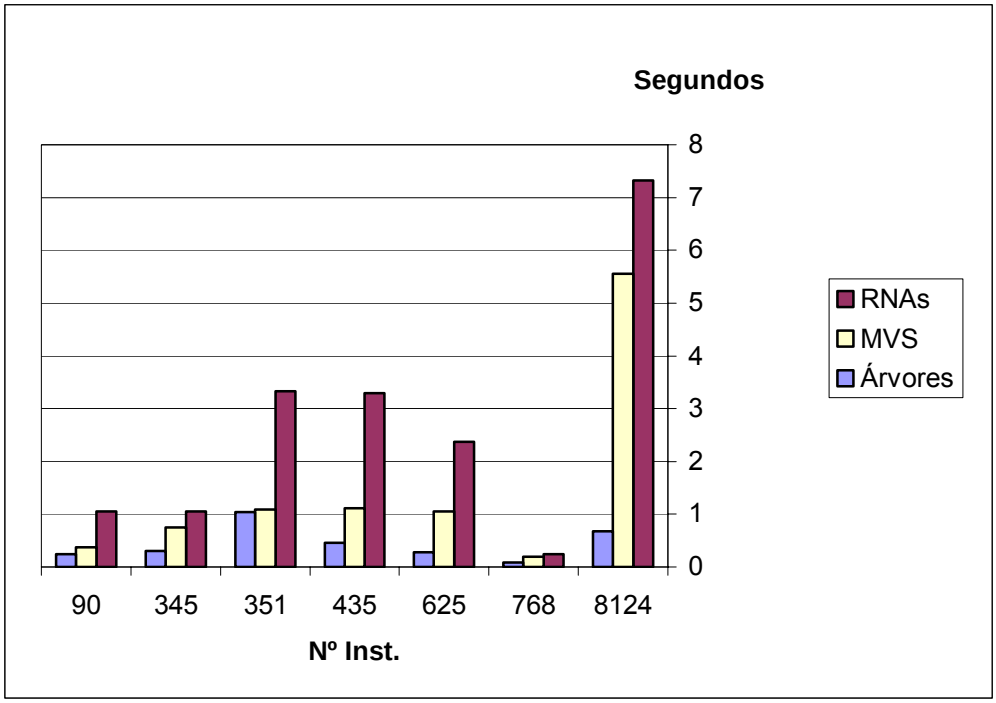

Figura 35 – Tempos de treino/ teste e número de instâncias em classificação (R).

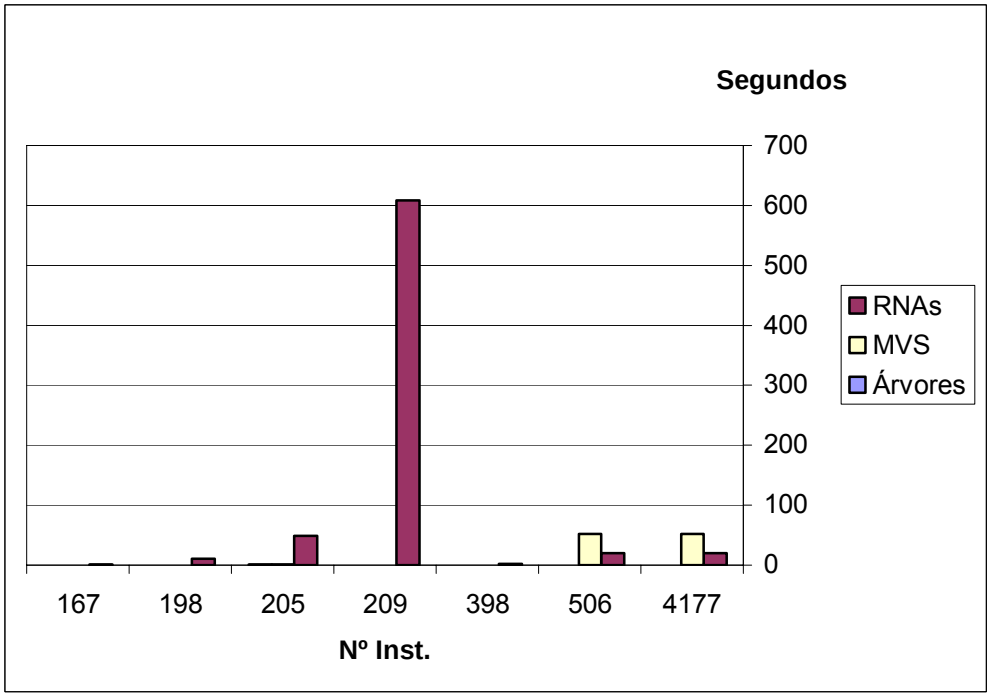

Figura 36 – Tempos de treino/ teste e número de instâncias em regressão (Weka).

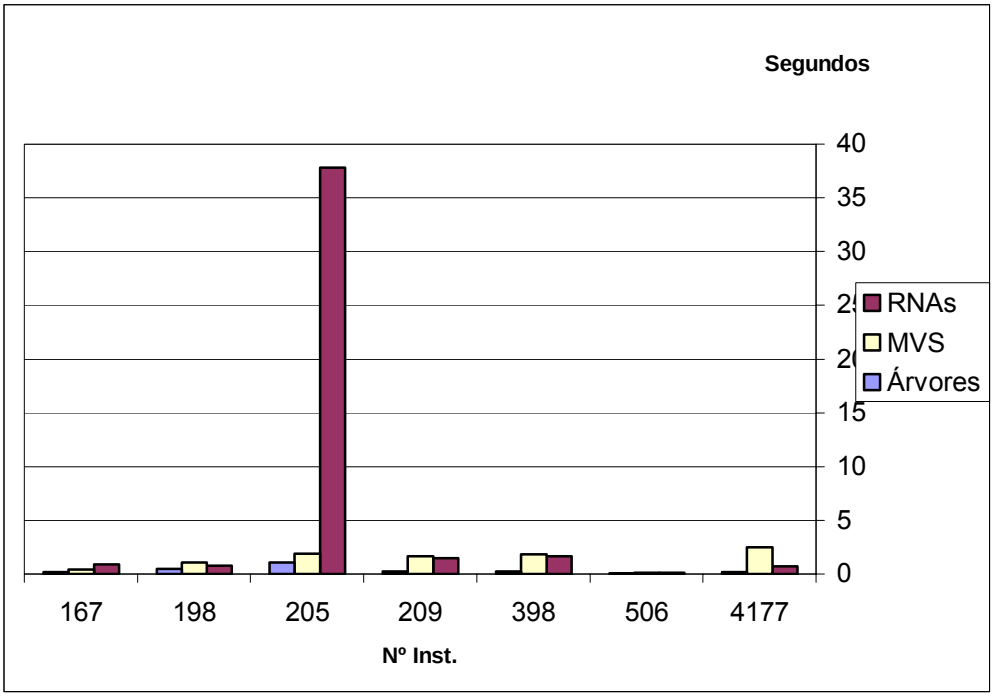

Figura 37 – Tempos de treino/ teste e número de instâncias em regressão (R).

5.6. Sumário

Procedeu-se à realização de experiências utilizando duas ferramentas de utilização livre muito divulgadas: o Weka e o R. O Weka disponibiliza vários tipos de interfaces, desde a simples linha de comando até interfaces gráficas atractivas e de fácil utilização. Ao invés, o R obriga à criação de código para a sua utilização (ver Anexo A), embora estejam a ser desenvolvidas funcionalidades gráficas que visam colmatar essa limitação (i.e. *Rattle* [51]). A configuração das ferramentas para as experiências realizadas tomou os valores por omissão, caso que reflecte a configuração mais provável do utilizador não especializado.

Os dados utilizados nas experiências foram retirados do repositório público UCI, de utilização frequente pela comunidade da Aprendizagem Automática/*Data Mining,* para testar algoritmos de extracção de conhecimento. Os conjuntos de dados escolhidos têm características muito variadas, tais como tamanhos diferentes, vários tipos de dados (numéricos, binários e nominais) e valores em falta. Perfazendo um total de 14 conjuntos, 7 foram escolhidos por serem adequados à tarefa de classificação, sendo os restantes 7 utilizados em regressão.

As experiências foram repetidas 20 vezes (*runs*), sendo desta forma possível obter variáveis estatísticas fiáveis (valores médios e desvio padrão). Os resultados obtidos foram compilados em tabelas por ferramenta e por objectivo de aplicação (i.e. classificação e regressão), sendo realçados os melhores valores. Para além disso, fez-se uma classificação das técnicas mediante os resultados obtidos, onde o melhor resultado foi pontuado com 3 pontos, o segundo melhor resultado com 2 e o terceiro lugar com 1 ponto. Feita a soma dos pontos, averiguou-se a posição relativa de cada técnica, criando-se um *ranking* para a classificação e outro para a regressão.

Os resultados das experiências apontam as MVSs como as que conseguem em geral os melhores resultados, sendo seguidas pelas RNAs. No entanto, as Árvores de Decisão/Regressão são a melhor opção num ou noutro caso, distinguindo-se ainda, pelo reduzido esforço computacional.

Capítulo 6

Conclusão

Neste capítulo tecem-se as conclusões que esta dissertação permite obter. Este capítulo tem início com a síntese da dissertação, seguindo-se uma discussão sobre a mesma. Finalmente, apontam-se algumas possibilidades de trabalho futuro. A discussão passa ainda, por uma referência às limitações deste trabalho e uma enumeração das contribuições que esta dissertação proporciona.

6.1. Síntese

Hoje em dia, as Tecnologias de Informação (TI) permitem a recolha e armazenamento de volumes de informação cada vez maiores. Esta enorme quantidade de dados dificulta a selecção de informação e extracção de conhecimento. Assim, é difícil distinguir qual a informação útil**,** sendo que os seres humanos são limitados e as técnicas estatísticas convencionais falham quando existe uma elevada complexidade e/ou dimensionalidade de dados.

O processo de Descoberta de Conhecimento em Bases de Dados, designado pela sigla *KDD*, implementa a extracção (semi-)automática de conhecimento de elevado nível a partir de dados em bruto. Caracteriza-se por várias etapas das quais se destaca o *Data Mining*. É nesta etapa que são utilizados algoritmos de extracção de conhecimento, designados comummente por técnicas de *Data Mining*. Actualmente, a aplicação do processo de KDD está generalizada nas mais diversas áreas, incluindo sistemas de *Business Intelligence* e de comércio electrónico, entre outros.

Nesta dissertação foram abordados dois objectivos de *Data Mining* deveras relevantes: a classificação e a regressão. Ambos utilizam uma aprendizagem supervisionada, sendo que se distinguem pelo tipo de saída considerada (i.e. discreta para o primeiro caso e contínua para o segundo). Dado o interesse neste tipo de tarefas, foram desenvolvidas diversas técnicas, cada uma com as suas vantagens e desvantagens. As Redes Neuronais Artificiais (RNAs) e as Máquinas de Vectores de Suporte (MVSs) distinguem-se por produzirem modelos que tendem a obter melhores resultados quando os dados

apresentam relações não lineares, embora os modelos obtidos sejam de difícil compreensão pelos seres humanos. Por sua vez, as Árvores de Decisão/Regressão, de utilização muito generalizada na área do *Data Mining*, criam modelos com uma estrutura de árvore, à base de regras do tipo "SE … ENTÃO …", sendo por isso de compreensão mais acessível. Por conseguinte, torna-se pertinente esclarecer as quais as capacidades das RNAs e das MVSs, comparando-as com as Árvores de Decisão. O apuramento das capacidades das RNAs e das MVSs quando aplicadas ao *Data Mining*, pode ser realizado aplicando-as a conjuntos de dados e comparando os seus resultados com os resultantes da aplicação de Árvores de Decisão/Regressão aos mesmos dados.

Por outro lado, existem diversas ferramentas de *software* que geralmente incluem uma grande variedade de técnicas. Para o utilizador comum, não especializado, importa saber como escolher não só o algoritmo de *Data Mining* mas também a ferramenta que implementa esse mesmo algoritmo. Assim, é importante saber quais as aplicações disponíveis. Neste trabalho foram analisadas três dezenas de ferramentas. Após uma verificação das suas características principais, foram definidos dois critérios de selecção para a fase experimental: i) deveriam implementar pelo menos RNAs, MVSs e Árvores e ii) deveriam ser de uso gratuito. Com base nestes critérios, foi escolhida a ferramenta Weka e o ambiente de programação estatístico R. Ambas as aplicações são bastante utilizadas (ver Capítulo 4). No entanto, o Weka é de utilização mais amigável que o R, embora tal facto possa ser alterado em breve com o desenvolvimento de novos *packages* gráficos, tal como o *Ratlle* (ainda numa fase beta de desenvolvimento).

Em geral, cada algoritmo de *Data Mining* contém diversos parâmetros que afectam a qualidade do modelo obtido. Durante a fase experimental, optou-se por utilizar os valores por omissão fornecidos pelas ferramentas, pois tal reflecte a configuração que um utilizador não especializado provavelmente irá utilizar. Por sua vez, os dados foram seleccionados tentando reflectir cenários distintos, para evitar que haja a possibilidade das técnicas se evidenciarem num ou noutro caso em particular. Assim, foi escolhido o repositório público UCI, tendo sido utilizados 7 problemas de classificação e 7 tarefas de regressão. Para cada conjunto de dados e técnica, foram aplicadas 20 execuções, com vista à obtenção de métricas fiáveis. Esses resultados foram compilados em tabelas e resumidos em gráficos para melhor visualização e mais fácil compreensão dos mesmos. Cada técnica foi analisada de acordo com dois factores: i) o seu desempenho em

previsão; e ii) o esforço computacional exigido (medido em termos de tempo). No que diz respeito ao primeiro factor, os melhores resultados foram obtidos pelas MVSs, seguidos das RNAs. Por sua vez, as árvores de decisão têm uma menor exigência em termos computacionais.

6.2. Discussão

Neste trabalho, pretendeu-se avaliar qual a qualidade, em termos de previsão, obtida por duas técnicas não lineares, as RNAs e MVSs, comparando-as com Árvores de Decisão/Regressão. Estas técnicas contêm diversos parâmetros, bem como variações do algoritmo de procura do modelo óptimo, que afectam o seu desempenho final, sendo que existem diversas implementações conforme o tipo de ferramenta que se utiliza. Por exemplo, a aplicação Weka contém vários algoritmos que implementam RNAs ou Árvores de Decisão/Regressão. Ora, o utilizador comum (não especializado), terá dificuldades em tomar escolhas, tendendo a aceitar os parâmetros/algoritmos sugeridos por estas ferramentas. Contudo, existem largas dezenas de ferramentas de *Data Mining*, sendo que cada uma apresenta um conjunto distinto de técnicas. Dado que uma exploração exaustiva de todo o *software* de *Data Mining* se encontra fora do âmbito desta dissertação, optou-se somente por, numa primeira fase, efectuar uma análise geral às ferramentas que disponibilizam RNAs ou MVSs, constatando-se que actualmente existem pelo menos 36 ferramentas com estas características. Este elevado número é um bom indicador de que há um elevado interesse no uso de RNAs e MVSs em aplicações de *Data Mining*.

Para além desta análise geral, também se efectuaram um conjunto de experiências, tendo-se utilizado duas ferramentas: o Weka e o R. Os resultados obtidos em diversos problemas do mundo real, revelam as MVSs como a melhor técnica de *Data Mining* em previsão, sendo que as Árvores de Classificação/Regressão obtêm os piores resultados. No entanto, a melhoria das MVSs é conseguida à custa de um maior esforço computacional, quando comparadas com as Árvores de Decisão/Regressão. Tal facto é deveras relevante, principalmente quando o domínio de aplicação der origem quantidades de dados de elevada dimensão.

Há que referir que os resultados foram obtidos com as técnicas configuradas com os valores de omissão e por isso não requerendo conhecimentos especializados por parte do utilizador. Tal facto contradiz de certo modo o argumento de que as RNAs e/ou MVSs são de difícil utilização. De modo algo surpreendente, os resultados obtidos também contradizem a necessidade de selecção de modelos e noção que o desempenho das RNAs e MVSs é mais sensível a uma correcta escolha dos seus hiper-parâmetros (e.g. número de nós internos da RNA ou parâmetros do *kernel* da MVS), do que no caso das Árvores de Decisão/Regressão.

Acresce ainda, que há investigação em curso para desenvolvimento de formas de extracção de regras de RNAs [34][25], bem como de MVS s[29]. Se e quando essa facilidade ficar disponível, também os modelos resultantes da aplicação destas técnicas passarão a ser mais compreensíveis. Por tudo isto, e pelos bons resultados, as RNAs e em especial as MVSs devem ser tomadas em forte consideração na selecção de técnicas para a criação de modelos supervisionados em aplicações de *Data Mining*.

6.2.1. Limitações

Importa também referir que existem diversas limitações neste estudo, nomeadamente:

- Testaram-se apenas 7 problemas de classificação e 7 tarefas de regressão, não existindo garantias que estes problemas, embora variados, correspondam ao que se espere encontrar no mundo real;

- Foram utilizados os conjuntos de dados originais conforme disponibilizados no repositório UCI, ou seja, já pré-processados, não existindo neste trabalho preocupações com as fases de pré-processamento (e.g. selecção de dados, transformação de variáveis, substituição de valores omissos);

- Foram somente analisadas duas métricas de avaliação das técnicas: a capacidade de previsão e o tempo. Não foram analisados outras dimensões como: facilidade de compreensão dos modelos, ou a novidade e utilidade do conhecimento adquirido.

6.2.2. Contribuições

Faz-se um levantamento de ferramentas bastante exaustivo, que dá uma panorâmica muito real do universo de ferramentas de *Data Mining* existentes na actualidade, com especial destaque para as aplicações que implementam RNAs e MVSs. Apontam-se também, algumas preferências dos utilizadores entre as ferramentas disponíveis.

Também se efectuou uma comparação das técnicas RNAs, MVSs e Árvores de Decisão/Regressão aplicando-as a casos variados muito utilizados pela comunidade de Aprendizagem Automática/*Data Mining*.

E ainda, foram usadas duas ferramentas de utilização livre muito populares (Weka e R), cuja utilização é explicada com detalhe, de modo a guiar um utilizador inexperiente em aplicações semelhantes.

6.3. Trabalho Futuro

Esta dissertação proporciona por fim, diversas perspectivas de trabalho futuro, nomeadamente:

- O número de ferramentas de *Data Mining* disponíveis é elevado, o que complica uma análise exaustiva das mesmas. Uma das formas de reduzir este número poderá passar por uma selecção com base na quantidade dos seus utilizadores. Assim, será conveniente realizar um estudo com rigor estatístico sobre quais as ferramentas mais utilizadas. Por outro lado, algumas aplicações de *Data Mining* podem ser utilizadas apenas por serem mais baratas ou de mais fácil utilização. Por isso, torna-se importante conhecer também, as principais razões da escolha das ferramentas por parte dos seus utilizadores.

- A avaliação das técnicas pode ser alargada no âmbito das próprias ferramentas, utilizando os diversos algoritmos que implementem a mesma técnica (note-se que nesta dissertação foram utilizados os algoritmos mais conhecidos, i.e. MLPs, SMO, C4.5/REPtree). Além disso, pode-se alargar a avaliação

experimental a mais ferramentas, mesmo àquelas que não implementam RNAs e MVSs.

- Embora não tenha sido analisado em detalhe nesta dissertação, a compreensão dos modelos criados é deveras importante em aplicações de *Data Mining*. Mais do que apenas aplicar o modelo, permite ao utilizador validar e utilizar o conhecimento extraído. Sob este ponto, convém referir que a extracção de regras a partir de RNAs e MVSs é uma área de investigação actual, pelo que se torna oportuna uma pesquisa do estado da arte neste tópico, bem como um estudo experimental.

Bibliografia

[1] Adriaans P. and Zantinge D. "*Data Mining,1 ed*.", Harlow: Addison-Wesley, 1996.

[2] Basheer I.A. and Hajmeer M., "*Artificial Neural Networks: Fundamentals, Computing, Design and Application*", Journal of Microbiological Methods, 43 (2000), 3-31.

[3] Batista P. and Silva M.J. *"Mining Web Access Logs of an On-line Newspaper"*, Departamento de Informática, Faculdade de Ciências – Universidade de Lisboa, http://xldb.fc.ul.pt/data/Publications_attach/rpec02.pdf, (2006).

[4] Birrien Jean-Yvon, *"História da Informática"*, Rés-Editora, 2002.

[5] Bors A. G., *"Introduction of the Radial Basis Function (RBF) Networks"*, Department of Computer Science, University of York, Y010 5dd, UK.

[6] Breiman L., Friedman J.H., Olshen R.A., and Stone, C.J., *"Classification and Regression Trees"*, Belmont, CA: Wadsworth, 1984.

[7] Bishop C., *"Neural Networks for Pattern Recognition"*, Oxford Univ. Press, 1995.

[8] Carvalho J.A., In Falkenberg E., K. Lyytinen and A. Verrijn-Stuart (Eds.), Information Systems Concepts: An Integrated Discipline Emerging (Proceedings of the ISCO 4 Conference, Leiden, Holanda, 20 -22 September 1999), Kluwer.Academic Publishers, 2000, pp.259-280.

[9] Cisterl A. M., Ebecken N. F. F., "*CRM through DM: a case study"*, Data Mining III, A Zanasi, CA Brebbia, NFF Ebecken & P Melli (Editors), ISBN 1-85312-925-9

[10] Cortes B., *"Sistemas de Suporte à Decisão"*, FCA – Editora Informática, 2005.

[11] Cortez P., *"Modelos Inspirados na Natureza Para a Previsão de Séries Temporais"*, Tese de Doutoramento, Universidade do Minho, 2002.

[12] Fahlman S., *"Faster Learning Variations on Back-Propagation: An Empirical Study"*, In D. Touretzky G. H. and Sejnowski, T., editors, Proceedings of Connectionist Models Summer School, pages 38-51, Los Altos CA, USA. Morgan Kaufmann Publishers, 1998.

[13] Fayyad U.M., Piatetsky-Shapiro G., Smyth S. and Uthurusamy R., *"Advances in Knowledge Discovery and Data Minin*g", M.I.T. Press, 1996.

[14] Fayyad U., Shapiro G., and Smyth P., *"From Data Mining to Knowledge Discovery in Databases"*, AI Magazine, 1996.

[15] Fayyad U.M., *"Data Mining and Knowledge Discovery in Databases: Implications for Scientific Databases"*, IEEE 1997, pp.2-11.

[16] Goebel M. and Gruenwald L., "*A Survey of Data Mining and Knowledge Discovery Software Tools"* , ACM SIGKDD, June 1999, Volume 1.

[17] Fuseda Y. and Satou K., *"Toward a Data Mining Service from Large and Heterogeneous Genome Databases in GenomeNet"*, School of Knowledge Science, Japan Advanced Institute of Science and Technology, www.jsbi.org/journal/GIW99/GIW99P42.pdf (2006).

[18] Habrand A. and Bernard M., Jaquenet F.*, "Multi-Relational Data Mining in Medical Databases"*, Université de Saint-Etienne, Springer-Verlag 2003, http://eurise.univ-st-etienne.fr/bibliographie/fichiers/hbj_mdmomd2003.pdf (2006).

[19] Hearst M., "*What is text mining?*", http://www.sims.berkeley.edu/~hearst/textmining.html, (2006).

[20] Han J. and Kamber M., "*Data Mining: Concepts and Techniques"*, Morgan Kaufmann, 2000.

[21] King D., CS 4803B- "*Numerical Machine Learnin*g", 1995.

[22] King M., Elder J., Gomolka B., Schmidt E., Summers M., and Toop K., *"Evaluation of Fourteen Desktop Data Mining Tools"*, http://www.datamininglab.com/pubs/smc98_king_elder.pdf (2006).

[23] Kohavi R. and Provost F. (1998). Glossary of Terms. *"Machine Learning"*, 30(2/3):271–274.

[24] Lima V.R., *"Informática e Informação"*, Universidade Lusíada, 1992.

[25] Milare C.R., de Carvalho A.C.P.L.F., Monard M.C., *"Extracting rules from neural networks using symbolic algorithms:preliminary results"*, Computational Intelligence and Multimedia Applications, 2001. ICCIMA 2001. Proceedings. Fourth International Conference on, Volume , Issue , 2001 Page(s):384 – 388.

[26] Morgan J.N. and Sonquist J.A., *"Problems in the analysis of survey data, and a proposal"*, Journal of the American Statistical Association, 58, 415-434, 1963.

[27] Newman D., Hettich S., Blake C. and Merz, C. UCI Repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html, Irvine, CA: University of California, Department of Information and Computer Science, 1998.

[28] Nicholson S., *"The Bibliomining Process: Data Warehousing and Data Mining for Library Decision-Making"*. Information Technology and Libraries 22 (4), 2003.

[29] Núñez H., Ângulo C., Catalã A., *"Rule extraction from support vector machines"*, ESANN'2002 proceedings - European Symposium on Artificial Neural Networks, Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 107-112.

[30] Patterson D., *"Artificial Neural Networks - Theory and Applications",* Prentice Hall, Singapore, 1996.

[31] Platt J., *"Using Sparseness and Analytic QP to Speed Training of Support Vector Machines"*, in Advances in Neural Information Processing Systems 11, M. S. Kearns, S. A. Solla, D. A. Cohn, eds., MIT Press, (1999).

[32] Prather J.C., Lobach D.F., and Goodwin, L.K., *"Medical Data Mining: Knowledge Discovery in a Clinical Data Warehouse"*, Duke University Medical Center of Durham, North Carolina, http://dci.mc.duke.edu/PDF_Files/Data%20Mining.pdf (2006).

[33] Quinlan J.R., *"Induction of decision trees"*, Machine Learning, 1, 81-106, 1986.

[34] Rabuñal J.R., Dorado J., Pazos A., Pereira J., Rivero D., *"A New Approach to the Extraction of ANN Rules and to Their Generalization Capacity Through GP"*, Neural Computation, July 2004, Vol. 16, No. 7, Pages 1483-1523

[35] Riedmiller M., *"Supervised Learning in Multilayer Perceptrons – from Backpropagation to Adaptive Learning Techniques*", Computer Standards an Interfaces, 16, 1994.

[36] Santos M.F. and Azevedo C., *"Data Mining, Descoberta de Conhecimento em Bases de Dados"*, FCA – Editora de Informática, 2005.

[37] Santos M.Y. and Ramos I., *"Business Inteligence"*, FCA – Editora Informática, 2006.

[38] Soares C., 2002, *"Is the UCI Repository Useful for Data Mining?"*, N. Lavrac and H. Motoda and T. Fawcett, Proceedings of the ICML- 2002 Workshop on Data Mining Lessons Learned, 69-75.

[39] Sousa S., *"Tecnologias da Informação: O que são? Para que servem?"*, FCA – Editora Informática, 2003.

[40] Stitson M.O., Weston J.A.E., Gammerman A., Vovk V. and Vapnik V., *"Theory of Support Vector Machines"*, Technical Report, December 31, 1996, University of London.

[41] Topchy A.P., Lebedko O.A., Miagkikh, Victor V., Kasabov and Nikola K., *"Adaptive Training of Radial Basis Function Networks Based on Cooperative Evolution and Evolutionary Programming"*, Research Institute for Multiprocessor Computer Systems, Department of Information Science, University of Otago.

[42] Vapnik V., "*Estimation of Dependences Based on Empirical Data*", Nauka, Moscow, 1979.

[43] Vapnik V., *"The Nature of Statistical Learning Theory"*, Springer-Verlag, New York, 1995.

[44] Vapnik V., Golowich S. and Smola A. *"Support vector method for function approximation, regression estimation, and signal processing"*, Advances in Neural Information Processing Systems, 9:281–287, 1996.

[45] Witten I. H. and Frank E., *"Data Mining – Practical Machine Learning Tools and Techniques"*, Elsevier, 2005.

[46] Sarle W., *"Neural network frequently asked questions (faq)."*, ftp://ftp.sas.com/pub/neural/FAQ.html, 2004.

[47] Zaisne O.R., Han J., Li Z., Chee S.H., and Chiang J.Y., *"MultiMediaMiner: A System Prototype for MultiMedia Data Mining"*, Intelligent Database Systems Research Laboratory and Vision and Media Laboratory, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada, http://www.cs.ualberta.ca/~zaiane/postscript/3m98.pdf (2006).

[48] Zupan J. and Gasteiger J., 1993. *"Neural Networks For Chemists: An Introduction"*, VCH, New York.

[49] http://www.crisp-dm.org/CRISPWP-0800.pdf (2006)

[50] http://www.kdnuggets.com (2006)

[51] http://www.r-project.org/ (2006)

[52] http:// www.the-data-mine.com (2006)

Anexo A

A seguir apresentam-se exemplos de código utilizados no R. Há um exemplo para cada tipo de validação e para cada objectivo: 10-*fold cross-validation* para classificação e regressão; e também um exemplo *de percentage split* para classificação e regressão. No que diz respeito às técnicas, encontram-se todas representadas.

A.1. 10-*fold cross-validation* em classificação

O código seguinte implementa a validação de 10-*fold cross-validation* (neste exemplo são utilizadas RNAs) para o conjunto de dados "houses".

```
dados<-read.table("D:/datasets_class/houses.txt",sep=",") # leitura dos dados
pr<-0 # vector para os valores de precisão de cada run
for(t in 1:20) # 20 runs
{
precisao<-0 # variável para o valor da precisão
for(i in 1:10){ #10 folds
amostra<-sample((43*i-43+1):(43*i),43) # cada fold corresponde a 10% dos dados
teste<-dados[amostra,] # 10% de teste
treino<-dados[-amostra,] # 90% de treino
modelo<-nnet(V17~.,data=treino,size=4,rang=0.2,decay=5e-4)
tabela<-table(teste$V17,predict(modelo,teste,type="class")) # tabela de resultados
total<-0
for(j in 1:2)
{
for (k in 1:2)
{
total<-total+tabela[j,k]
}
}
diagonal<-0
for(m in 1:2){
diagonal<-diagonal+tabela[m,m]
```

```
}
precisao<-precisao+diagonal/total
}
pr[t]<-precisao/10
}
media<-0
for(t in 1:20)
{
media<-media+pr[t]
}
media<-media/20 # valor médio da precisão
V<-0  # cálculo da variância
for(t in 1:20)
{
pr[t]<-(pr[t]-media)^2
V<-V+pr[t]
}
print(media)
print(V)
print(sqrt(V/20)) #desvio padrão
```

A.2. *Percentage Split* em classificação

O código seguinte implementa a validação *percentage split* de 66%, em que a principal diferença relativamente ao caso anterior é a amostragem aleatória de 66% dos dados para treino do modelo (no caso MVSs), ficando o resto para teste:

```
dados<-read.table("D:/datasets_class/prima.txt",sep=",") # leitura dos dados
pr<-0 # vector para os valores de precisão de cada run
for(t in 1:20) # 20 runs
{
precisao<-0 # variável para o valor da precisão
amostra<-sample(1:767,506)  # amostra aleatória de 66%
teste<-dados[-amostra,] # para teste os restantes 34%
```

```
treino<-dados[amostra,] # dados de treino 66%
modelo<-svm(V9~.,data=treino)
tabela<-table(teste$V9,predict(modelo,teste,type="class")) # tabela de resultados
total<-0
for(j in 1:2)
{
for (k in 1:2)
{
total<-total+tabela[j,k]
}
}
diagonal<-0
for(m in 1:2){
diagonal<-diagonal+tabela[m,m]
}
precisao<-diagonal/total
pr[t]<-precisao
}
media<-0
for(t in 1:20)
{
media<-media+pr[t]
}
media<-media/20
V<-0 # cálculo da variância
for(t in 1:20)
{
pr[t]<-(pr[t]-media)^2
V<-V+pr[t]
}
print(media)
print(V)
print(sqrt(V/20)) # desvio padrão
```

A.3. 10-*fold cross-validation* em regressão

O exemplo de código a seguir apresentado implementa a forma de validação de 10-*fold cross-validation* para o caso da regressão (com Árvores de Decisão/Regressão). A métrica utilizada é RRSE.

```
dados<-read.table("D:/datasets_reg/brestnorm.csv",sep=";") # leitura dos dados
RRSE<-0 # vector para os valores de RRSE para cada run
for(t in 1:20) # 20 runs
{
numerador<-0 # numerador da fórmula RRMSE
denominador<-0 # denominador da fórmula RRMSE
media<-0.369
for(i in 1:10){ #10 folds
amostra<-sample((19*i-19+1):(19*i),19)  # cada fold corresponde a 10% dos dados
teste<-dados[amostra,] # 10% de teste
treino<-dados[-amostra,] # 90% de treino
modelo<-tree(V34~.,data=treino)
previsao<-predict(modelo,teste)
for(j in 1:19){
erro<-teste$V34[j]-previsao[j]
erro<-erro*erro
numerador<-numerador+erro
den<-teste$V34[j]-media
den<-den*den
denominador<-denominador+den
}
}
RRSE[t]<-sqrt(numerador/denominador)
}
media_e<-0 # média do RRSE
for(t in 1:20)
{
media_e<-media_e+RRSE[t]
```

```
}
media_e<-media_e/20
V<-0 # cálculo variância
for(t in 1:20)
{
RRSE[t]<-(RRSE[t]-media_e)^2
V<-V+RRSE[t]
}
print(media_e)
print(V)
print(sqrt(V/20)) # desvio padrão
```

A.4. *Percentage Split* em regressão

O exemplo de código a seguir apresentado implementa a forma de validação de *percentage split* de 66% para o caso da regressão (com Árvores de Decisão/Regressão). A métrica utilizada é RRSE.

```
dados<-read.table("D:/datasets_reg/ampgnorm.csv",sep=";") # leitura dos dados
RRSE<-0 # vector para os valores de RRSE para cada run
for(t in 1:20) # 20 runs
{
numerador<-0 #numerador da fórmula RRMSE
denominador<-0 #denominador da fórmula RRMSE
media<-0.386
for(i in 1:10){ #10 folds
amostra<-sample((39*i-39+1):(39*i),39)  # cada fold corresponde a 10% dos dados
teste<-dados[amostra,] # 10% de teste
treino<-dados[-amostra,] # 90% de treino
modelo<-tree(V8~.,data=treino)
previsao<-predict(modelo,teste)
for(j in 1:39){
erro<-teste$V8[j]-previsao[j]
```

```
erro<-erro*erro
numerador<-numerador+erro
den<-teste$V8[j]-media
den<-den*den
denominador<-denominador+den
}
}
RRSE[t]<-sqrt(numerador/denominador)
}
media_e<-0 # média do RRSE
for(t in 1:20)
{
media_e<-media_e+RRSE[t]
}
media_e<-media_e/20
V<-0 # cálculo variância
for(t in 1:20)
{
RRSE[t]<-(RRSE[t]-media_e)^2
V<-V+RRSE[t]
}
print(media_e)
print(V)
print(sqrt(V/20)) # desvio padrão
```

Glossário

Aprendizagem não supervisionada – a aprendizagem é feita descobrindo similaridades nos dados, ou seja, pretende-se descobrir agrupamentos de dados com características semelhantes.

Aprendizagem por reforço – a aprendizagem faz-se com recurso a "recompensas" caso as respostas do modelo sejam correctas, e "punições" caso não o sejam.

Aprendizagem supervisionada – aprendizagem realizada com recurso a casos conhecidos de dados de entrada e dados de saída.

Automatic Interaction Detection (AID) – método de criação de Árvores de Decisão através da separação dos dados em grupos sucessivos com base nos seus valores, proposto por Morgan e Sonquist [26] em 1963.

Backpropagation – algoritmo de treino de RNAs, que proporciona uma aprendizagem por propagação para trás nas várias sinapses, dos erros obtidos como resposta do modelo.

Business Intelligence – combinam recolha e armazenamento de dados com ferramentas de extracção de informação relevante para a tomada de decisão.

CHAID – método de criação de Árvores de Decisão proposto por Hartigan [10] em 1975, que tem por base o método AID, mas com o teste do chi-quadrado para identificar variáveis independentes.

Chunking – método de treino de MVSs que consiste em retirar da matriz os multiplicadores Lagrangeanos com $\alpha_i=0$.

Classificação – procura de uma função que faça o mapeamento dos dados de treino em classes pré-definidas.

Classification and Regression Trees (CART) – algoritmo de criação de Árvores de Decisão/Regressão desenvolvido em 1984 por Breiman et al. [6].

Clustering – procura de um número finito de conjuntos (ou *clusters*) que descrevam os dados (Segmentação).

CRISP-DM (*Cross Industry Standard Process for Data Mining*) – metodologia de *Data Mining* desenvolvida por um consórcio composto por NCR Systems Engineering Copenhagen, DaimlerChrysler AG, SPSS Inc. e OHRA Verzekeringen en Bank Groep B.V.

CRM – *Customer Relationship Management*.

Cross-Validation – técnica de validação de dados em que os dados são divididos em N blocos de dimensão semelhante para uma aprendizagem de N iterações, em que a cada iteração se utilizam N-1 blocos para aprendizagem e o outro para teste, sendo este diferente a cada iteração.

C4.5 – algoritmo de criação de Árvores de Decisão desenvolvido por Ross Quinlan [33], tendo por base o ID.

Data Warehouses – repositórios de informação organizacional em formato adequado à análise.

Features – atributos dos dados relevantes para definição dos vectores de suporte.

GNU – *General Public Licence*.

GUI – *Graphical User Interface*.

Iterative Dichtomizer (ID) – algoritmo de criação de Árvores de Decisão desenvolvido por Ross Quinlan [33].

Knowledge Discovery in Data (KDD) – Descoberta de Conhecimento em Bases de Dados.

Máquinas de Vectores de Suporte (MVSs, *Support Vector Machines*) – modelos que se baseiam na Teoria Estatística Aprendizagem, propostos pela primeira vez por Vapnik em 1979 [42].

Matriz de confusão – tabela que apresenta os resultados de um modelo de previsão aplicado a classificação, em que uma das entradas é constituída pelas classes desejadas, a outra pelas classes previstas pelo modelo; as células são preenchidas com o número de instâncias que correspondem ao cruzamento das entradas.

Métodos de *Kernel* – funções que fazem uma transformação não linear aos dados para um espaço multi-dimensional.

Multi-Layer Perceptrons (MLPs) – RNA do tipo unidireccional, constituída por várias camadas de neurónios artificiais.

NAs – Valores omissos.

OLAP – *On-Line Analytic Processing*.

Percentagem de acertos – valor percentual obtido da divisão do número de respostas correctas de um modelo de previsão, pelo número total de respostas.

QuickPropagation – algoritmo de treino de RNAs desenvolvido por Fahlman em 1998 [12] com base no *Backpropagation*, mas mais rápido.

Radial Basis Functions – funções monotónicas à medida que se afastam do seu centro, utilizadas como funções de activação nos neurónios artificiais de alguns tipos de RNAs (redes RBF).

Regressão – procura de uma função desconhecida cuja saída tenha um domínio de valores reais.

Redes Neuronais Artificiais (RNAs) – modelos conexionistas com a capacidade de obterem conhecimento através de aprendizagem com dados conhecidos.

Redes recorrentes – topologia de RNAs em que existam ciclos nas redes.

Redes unidireccionais – topologia de RNAs em que não existam ciclos nas redes.

Reduced-error pruning – técnica de poda de Árvores de classificação/Regressão em que parte dos dados são utilizados para teste da árvore após a sua construção.

Remote Method Invocation (RMI) – método de computação distribuída proporcionado pela ferramenta WEKA.

Resilient Backpropagation (RPPROP) – algoritmo de treino de RNAs proposto por Riedmiller [35], com base no *Backpropagation*, mas mais rápido.

RRSE – *Root Relative Squared Error*

SEMMA (*Sample, Explore, Modify, Model, Assessment*) – metodologia de *Data Mining* desenvolvida pela empresa SAS.

Sequential Minimal Optimization (SMO) – algoritmo de treino de MVSs em que os multiplicadores Lagrangeanos são calculados analiticamente.

SGBD – Sistema de Gestão de Bases de Dados.

Sobre-ajustamento (*overfitting*) – situação em que o modelo criado adapta-se muito bem aos casos utilizados na aprendizagem, mas responde mal em novos casos.

SQL – Structered Query Language.

Text Mining – extracção de conhecimento de grandes volumes de dados sob a forma de texto.

TI – Tecnologias da Informação.

UCI – repositório público onde se encontram dados disponibilizados por várias fontes (governamentais, universitárias e outras), para serem utilizados livremente em investigação.