Universidade do Minho

Licenciatura em Informática de Gestão

Opção III – 5º Ano

Data Mining em R

(Projecto Nº 5)

Orientador: Paulo Cortez

Fevereiro de 2005, Azurém

Realizado por:

Grupo 20

	Cláudio Alexandre Morais Pinto nº 28861 campalexandre@iol.pt
	Hugo Abel da Silva Vieira nº 27953 hasv@mail.pt
	Luís Miguel Parreira Bulhões nº 28161 lbulhoes@iol.pt

ÍNDICE

1. INTRODUÇÃO

Este trabalho surge na disciplina de Opção III do 5º ano da Licenciatura de Informática de Gestão da Universidade do Minho. Por sua vez, com esta disciplina pretende-se, para além do desenvolvimento de competências específicas, a realização de um projecto de investigação. Para a consecução deste objectivo optámos pelo tema “Data Mining em R”.

Existem várias ferramentas e para cada uma estão disponíveis diversas técnicas para o mesmo tipo de problema de Data Mining. Estas, por seu turno, devem ser escolhidas em função dos resultados obtidos para várias parametrizações.

O R é uma ferramenta destinada ao desenvolvimento de sistemas de apoio à decisão de análise de dados, bem como à execução de tarefas mais complexas que envolvam programação. Ao mesmo tempo, é uma linguagem de programação e um ambiente para computação estatística e gráficos.

Para a análise das diversas técnicas foi escolhida uma base de dados com o objectivo de criar um sistema de forma a prever o nível de qualidade da água de uma albufeira. Utilizou-se a ferramenta R com o objectivo de permitir obter uma ideia genérica sobre o tipo de funcionamento da ferramenta em questão e qual a sua potencialidade.

Na realização do plano de trabalhos, o grupo começou por efectuar pesquisa bibliográfica relativa ao tema em estudo. Inicialmente, o Hugo Vieira ficou responsável por estudar os artigos relativos à parte de Data Mining, enquanto o Cláudio Pinto e o Luís Bulhões se responsabilizaram por, em primeiro lugar, instalar a ferramenta R e, em segundo lugar, estudar essa mesma ferramenta através de alguns manuais e artigos obtidos através da pesquisa realizada. Após esta fase, reuniram-se os conhecimentos e iniciou-se o trabalho conjunto das duas partes.

Para além da introdução o relatório é composto por mais quatro capítulos. No segundo capítulo é constituído por fundamentos teóricos tanto de Data Mining como de ferramenta em questão. Assim, sobre Data Mining, abordámos os temas relacionados com o processo de descoberta de conhecimento, Data Mining, tarefas de Data Mining e metodologias de Data Mining. Quanto à teoria relacionada com o R falámos sobre como obter e instalar a ferramenta, bem como aspectos básicos da sua utilização. No capítulo três analisamos o caso de estudo referente a uma base de dados de uma albufeira. No que diz respeito ao quarto capítulo abordamos o estudo do problema de classificação através de comandos em R. A finalizar o relatório temos a conclusão com os aspectos positivos e negativos do trabalho.

O plano previsto para o trabalho foi o seguinte:

Project Start Date: Tue 19-10-04
Project Finish Date: Fri 11-02-05

Planeamento da Opção III

ID	Task_Name	Duration	Start_Date	Finish_Date	Predecessors	Resource_Names
1	Enrega de Planeamento	1 day?	Wed 20-10-04	Wed 20-10-04		
2	Pesquisa bibliografica sobre R	6,5 days?	Wed 20-10-04	Wed 27-10-04		
3	Pesquisa de data Mining	8 days?	Mon 25-10-04	Tue 02-11-04		
4	Leitura de Documentos sobre R	15 days	Mon 25-10-04	Thu 11-11-04		
5	Instalação da ferramenta e do package ezMining.	5 days	Mon 01-11-04	Fri 05-11-04		
6	Experimentação da ferramenta	10 days	Mon 08-11-04	Fri 19-11-04	5	
7	Aplicação da ferramenta a alguns exemplos já dados	15 days	Mon 22-11-04	Fri 10-12-04	6	
8	Desenvolvimento de problemas de classificação e teste	8 days	Tue 14-12-04	Thu 23-12-04	4	
9	Continuação do desenvolvimento de problemas de classificação e teste	19 days?	Mon 03-01-05	Thu 27-01-05		
10	Realização do poster para a disciplina	2 days?	Thu 27-01-05	Fri 28-01-05		
11	Entrega do relatorio Final	10 days	Mon 31-01-05	Fri 11-02-05	10	

2. FUNDAMENTOS TEÓRICOS

2.1 Data Mining

2.1.1 Processo de Descoberta de Conhecimento

Processo de descoberta de conhecimento é o processo que, globalmente, transforma dados de baixo nível em conhecimento de alto nível. Descoberta de conhecimento em bases de dados é um processo, não trivial, de identificação de padrões válidos, originais, potencialmente úteis e em último caso, de padrões compreensíveis em dados.
Sendo assim existem diversas formas de descobrir conhecimento, métodos, ferramentas, processos, etc. Aqui seguem-se alguns exemplos:

Capacidade para aceder a variadas fontes de dados:
Normalmente os dados a serem analisados, pertencem a diferentes áreas de uma empresa, corporação, etc. Esses dados têm de ser recolhidos, verificados e integrados antes de ser realizada uma análise mais detalhada

Acesso a dados Offline/Online:
Acesso a dados online significa que diferentes queries podem correr directamente sobre a mesma base de dados correndo concorrentemente com outras transacções. Acesso a dados offline significa que a análise feita aos dados é desempenhada num determinado instante numa fonte de dados, muitas vezes importando ou exportando processos da fonte de dados original para um formato de dados específico, exigido pelas ferramentas de descoberta de conhecimento.

O modelo de dados subjacente:
Para cada caso, podem existir modelos de dados específicos. Existem muitas ferramentas, hoje em dia, que apenas aceitam o seu input, na forma de apenas

uma tabela. Isto significa que para cada caso existe um modelo de dados que mais se adequa ao problema.

Número máximo de tabelas/linhas/atributos:
São limites hipotéticos nas capacidades de descoberta de conhecimento, da ferramenta utilizada.

Tamanho máximo da base de dados que a ferramenta confortavelmente manipula:
Este factor deve ser encarado como muito importante. Existem variados factores para além da capacidade de linhas, colunas, tabelas que a ferramenta pode suportar. Existem factores como a memória, o tempo de processamento, capacidades de visualização, etc.

Tipos de atributos que a ferramenta suporta:
Algumas ferramentas de descoberta de conhecimento têm algumas restrições quanto ao tipo de atributos que suporta.

Linguagem Query:
A linguagem query actua como um interface entre o utilizador, o conhecimento e a base de dados. Permite ao utilizador processar dados e conhecimento e controlar o processo de descoberta de conhecimento.

2.1.2 Data Mining

É o processo que assenta na extracção de padrões ou modelos sobre dados observados, isto é, após o estudo/observação de determinados dados, é possível a extracção de padrões ou modelos, sobre esses mesmos dados.

2.1.3 Tarefas de Data Mining

Processamento de Dados:
Dependendo dos objectivos e dos requisitos do processo de descoberta de conhecimento em bases de dados (KDD), analistas podem seleccionar, filtrar, agregar, retirar amostras, limpar ou transformar dados.

Previsão:
Dados um conjunto de dados e um modelo de previsão, prevê o valor para um atributo específico do respectivo conjunto de dados.

Regressão:
Dado um conjunto de dados, regressão é a análise da dependência do valor de alguns atributos sobre os valores de outros atributos no mesmo conjunto, e a produção automática de um modelo que consegue prever os valores dos atributos para outros registos.

Classificação:
Dado um conjunto de classes predefinidas, determina para qual destas classes, um dado específico pertence.

Clustering:
Dado um conjunto de dados, parte esse conjunto num conjunto de classes, para que dados com características semelhantes sejam agrupados.

Análise Link (associações):
Dado um conjunto de dados, identifica as relações entre atributos e os dados para que a presença de um padrão implique a presença de outro padrão.

Visualização do modelo:
Técnicas de visualização ajudam a melhor compreender o problema. Técnicas como a visualização de gráficos, histogramas, filmes 3D, etc.

Análise exploratória de dados (EDA):

Análise exploratória de dados é a exploração interactiva de um conjunto de dados sem grandes dependências em suposições preconcebidas ou modelos, como a tentativa de identificar padrões interessantes.

2.1.4 Metodologia Data Mining

Métodos estatísticos:

Aproximações estatísticas, usualmente, assentam num modelo probabilístico explícito. Estatísticas são candidatas à geração de hipóteses ou modelos.

Raciocínio com base em casos:

É uma tecnologia que tenta resolver um dado problema através do uso directo de soluções e experiências passadas.

Redes Neuronais:

São formadas de largos números de neurónios simulados, conectados uns aos outros, de uma maneira similar aos neurónios cerebrais. A força das interligações dos neurónios podem mudar (ou ser mudadas pelo algoritmo de aprendizagem) em resposta a um dado estímulo ou a um output obtido, que força a rede a “aprender”.

Árvores de Decisão:

É uma árvore em que cada nodo não terminal representa um teste ou decisão em relação a um determinado dado.

Regras de indução:

Regras que determinam uma correlação estatística entre a ocorrência de determinados atributos num conjunto de dados.

Algoritmos genéticos / Programação Evolutiva:

São estratégias de optimizações algorítmicas que são inspiradas pelos princípios observados na evolução natural. Dado um conjunto de potenciais soluções para determinado problema, estas competem entre si. As melhores soluções são seleccionadas e combinadas umas com as outras.

2.2 Introdução ao R

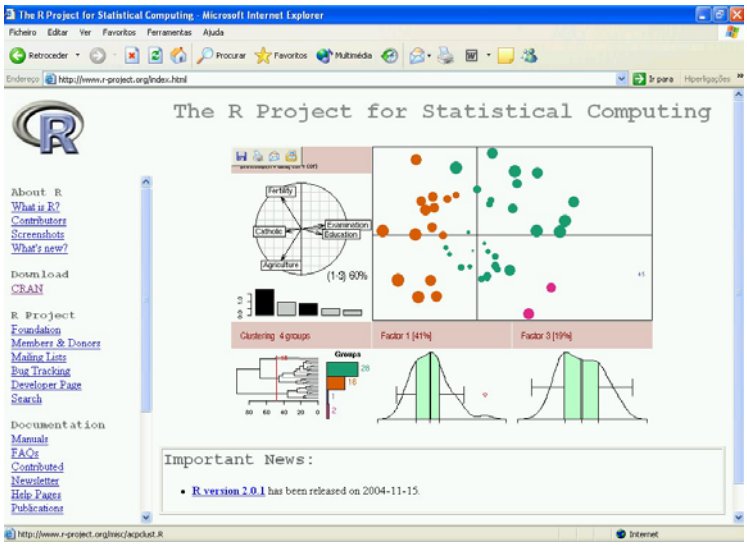
O **R** é uma ferramenta para o desenvolvimento de sistemas de apoio a decisão e análise de dados tal como a execução de tarefas mais complexas que envolvam programação. Uma das suas principais características é o seu carácter gratuito, estando disponível em diversas plataformas (Windows, Linux, MacOS).

Tem origem na linguagem S, desenvolvida nos laboratórios da AT&T Bell por Rick Becker, John Chambers e Allan Walkins. Esta é uma ferramenta de código é aberto (Open Source), ou seja, é susceptível de ser alterado pelo utilizador quer para modificação das funcionalidades existentes quer para o desenvolvimento de novas funcionalidades.

É uma ferramenta bastante poderosa com uma programação por objectos e um conjunto bastante vasto de packages que acrescentam bastantes potencialidades.

Para Data Mining são especialmente úteis algumas bibliotecas e funções, como iremos ver mais à frente. A biblioteca **rpart** (recursive partition and regression trees) pode ser utilizada tanto para árvores de regressão como de classificação. A função **lm** pode ser usada para obter regressões lineares. Já a biblioteca **nnet** (neural networks) é utilizada no âmbito das redes neuronais.

Para a instalação do **R** basta ligar o computador a Internet e fazer o download a partir do site desta aplicação, http://www.R-project.org. Este site devera ser seguido o link com o nome **CRAN** no menu disponível a esquerda. Ai deverá seguir o link Windows (95 *and later*) disponível na secção *Precompiled Binary Distributions*. No écran seguinte, deve seguir a pasta **base** e fazer o download do ficheiro **rw2001.exe** (o nome do ficheiro poderá variar em versões posteriores do R).

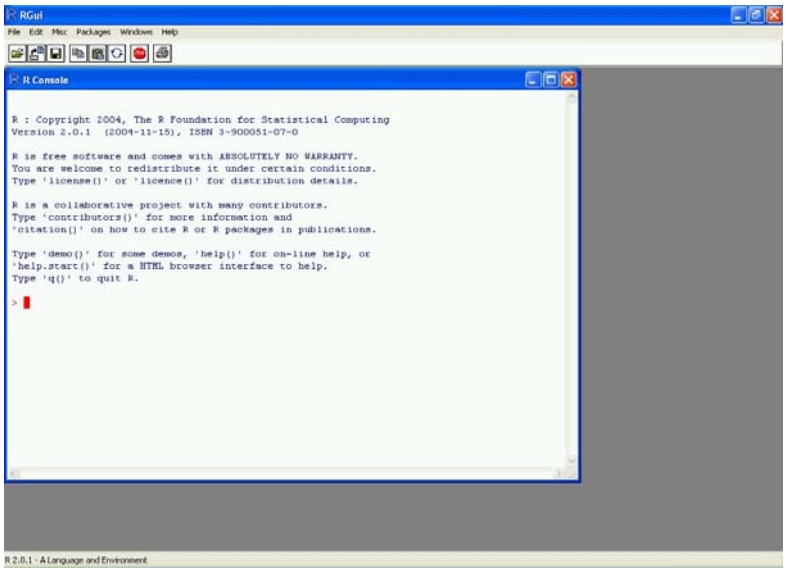


Uma vez efectuado o download do ficheiro deverá proceder-se a instalação do R, bastando para isso executar o ficheiro.

2.2.1 Começar a usar o R

Depois da instalação efectuada, para usar o R, basta clicar no ícone que, normalmente está disponível no desktop do Windows ou então ir à directoria onde foi instalado o programa e na pasta **\bin** executar o ficheiro **RGui**.

A execução do R faz aparecer a seguinte janela:



Esta janela apresenta o **prompt** do R **(>)**, com o cursor à sua frente. É aqui que vamos introduzir os comandos que pretendemos que o R execute.

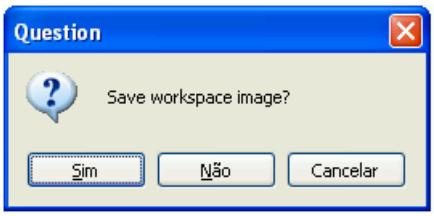
Por exemplo, para saber qual a versão do R que estamos a utilizar basta utilizar o comando *R.version* e premir Enter.

```
> R.version
               _
platform i386-pc-mingw32
arch     i386
os       mingw32
system   i386, mingw32
status
major    2
minor    0.1
year     2004
month    11
day      15
language R
```

Para terminar a execução do R basta executar o seguinte comando:

```
> q()
```

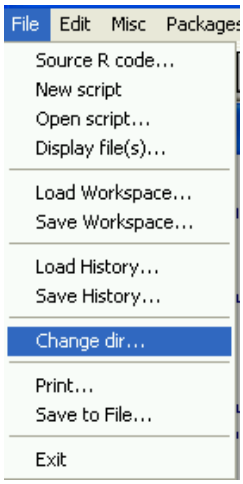
Ao executar este comando vai aparecer uma caixa de diálogo como apresentada na figura seguinte. Se respondermos *Sim* a esta pergunta o R vai guardar a informação contida num ficheiro, permitindo que, na próxima vez que executarmos o R poderemos continuar o trabalho anterior. Se optarmos por seleccionar *Não* o programa irá fechar sem guardar o trabalho efectuado nessa sessão.



Se escolhermos guardar o estado actual, o R vai criar um ficheiro **.Rhistory.** Este ficheiro é gravado no directório actual onde o programa está a funcionar. Para saber o directório actual basta fazer no prompt,

```
> getwd()
[1] "C:/R/rw2001"
```

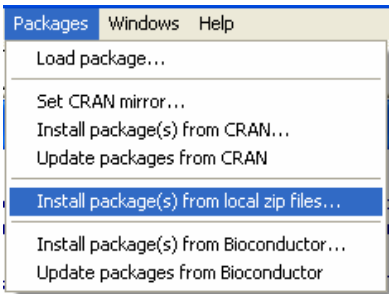
É possível alterar esse directório indo ao menu *File* e escolhendo a opção *Change dir…*



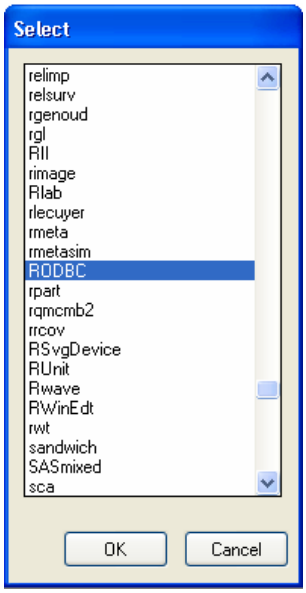
2.2.2 Packages

Quando instalado, o R já vem com alguns pacotes de base que fornecem as principais funcionalidades para análise de dados, ou seja, já possui os comandos básicos para a execução do programa. No entanto, e uma vez tratar-se de uma aplicação de "Open Source", outras funcionalidades foram e continuam a ser desenvolvidas e disponibilizadas com o propósito de serem acrescentadas programa.

É possível instalar novas funcionalidades (packages) que são disponibilizadas no CRAN. Uma forma de proceder à instalação destes novos pacotes pode ser através de ficheiros zip disponibilizados no CRAN para os vários packages, utilizando o menu *Packages* seguido de *Install Packages(s) from local zip files.*



Outra maneira de instalar packages é aceder directamente através da Internet usando, também no menu *Packages* a opção *Install Packages(s) from CRAN*. Irá, então, aparecer uma caixa com os packages existentes no CRAN como demonstra a figura seguinte.



Neste caso está a ser seleccionado o package RODBC que é um pacote que, por defeito não é instalado quando a é instalada a ferramenta.

2.2.3 Documentação de apoio

É de realçar a existência de muita documentação de apoio de suporte à ferramenta. O próprio CRAN disponibiliza bastantes documentos de apoio ao uso da aplicação. Outra forma possível de obter informação sobre o modo de

funcionamento do programa é através da função de ajuda integrada na ferramenta. O R vem com uma função de ajuda muito útil para compreender e saber utilizar os comandos disponíveis. Essa opção de ajuda também fornece, na maioria dos casos, alguns exemplos do modo como utilizar determinado comando. Por exemplo, se é pretendido obter ajuda sobre a função **sd** usamos:

```
> help(sd)
```

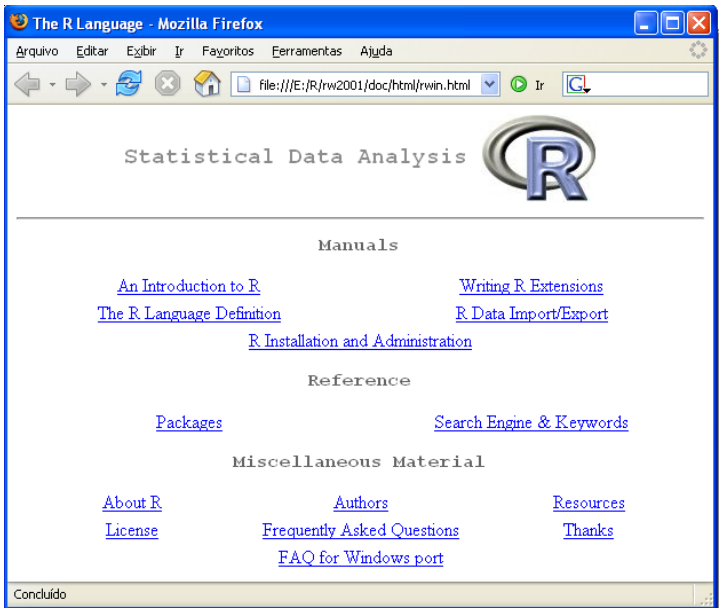
ou, em alternativa,

```
>?sd
```

Executando-se um destes comandos, irá abrir-se uma janela com a ajuda sobre a função, nomeadamente os argumentos que aceita, a explicação de cada argumento e exemplos.

Também disponível está um manual da aplicação em formato HTML que é obtido através do comando:

```
>help.start()
```



3. CASO DE ESTUDO

Para o estudo desta ferramenta foi utilizada uma base de dados já existente. Esta base de dados encontra-se disponível no url http://piano.dsi.uminho.pt/disciplinas/LIGIA/20032004/ia20022003.htm. Este repositório de dados *BDALBUFEIRA4* contém informação sobre o nível da qualidade da água de uma albufeira. É de salientar que esta base de dados se encontrava no formato Excel e foi tratada onde optámos por descartar algumas variáveis por estas terem valores muito díspares, não se mantendo ao longo do tempo. Posteriormente convertida em formato Access.

Uma vez que o objectivo, neste caso de estudo, é gerar/induzir um modelo baseado em regras de produção que permita prever, da forma mais eficiente, qual o nível de qualidade da água, estamos perante um **problema de classificação** – aprendizagem de uma função que faça o mapeamento de um elemento dos dados em uma ou várias classes.

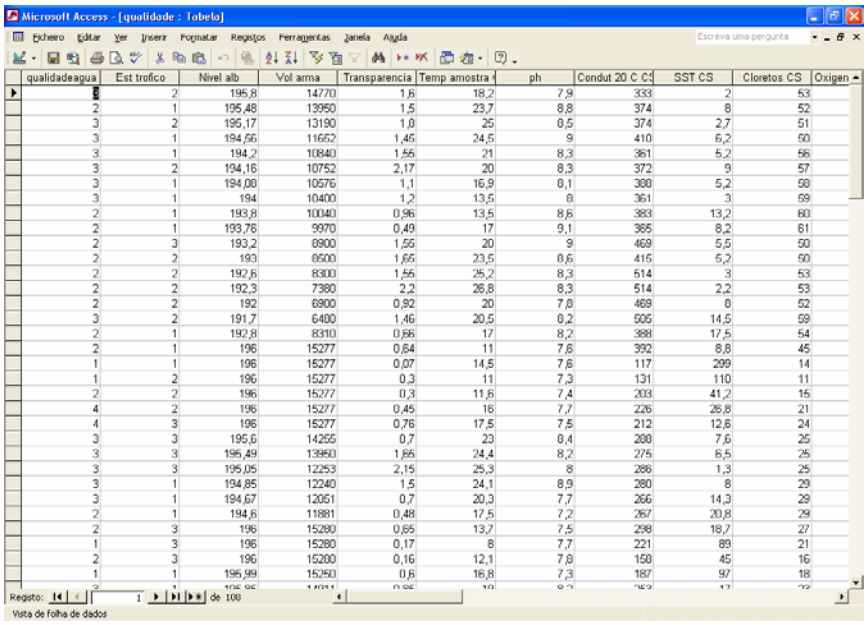
Podemos concluir que, dependendo dos valores existentes em cada uma das variáveis que afectam a qualidade da água, esta pode atingir cinco níveis diferentes (1, 2, 3, 4 ou 5).

No estudo de Data Mining nas árvores de decisão, árvores de regressão, regressão linear, análise linear discriminante e redes neuronais foi utilizada sempre a mesma base de dados.

A BDALBUFEIRA4 é constituída por uma tabela qualidade, composta por um total de 33 campos (1 de saída “qualidadeagua” e os restantes de entrada). Tudo isto, faz um total de 3564 instâncias. Todos estes campos são numéricos e não existem valores em falta. Estes campos têm o seguinte significado:

1	qualidadeagua	Qualidade da água da albufeira com captação à superfície (CS). Pode ser classificada em 1, 2, 3, 4 e 5.
2	Est trofico	Estado trófico com CS
3	Nível alb	Nível da albufeira em metros
4	Vol arma	Volume armazenado na albufeira em dam^3
5	Transparencia	Transparência da água em metros
6	Temp Amostra CS	Temperatura da amostra com CS em ºC
7	ph	Ph com captação à superfície
8	Condut 20 C CS	Condutividade a 20ºC com CS
9	SST CS	SST em mg dm^{-3} com CS
10	Cloretos CS	Cloretos em mg dm^{-3} com CS
11	Oxigenio dissCAMP CS	Oxigénio dissolvido (CAMP) com CS
12	Oxigenio diss AB CS	Oxigénio dissolvido (LAB) com CS
13	Oxidabilidade CS	Oxidabilidade em mg dm^{-3} com CS
14	CBO5 CS	CBO5 em mg dm^{-3} com CS
15	Azoto amoniacal CS	Azoto amoniacal em mg dm^{-3} com CS
16	Azoto	Azoto kjeldahl em mg dm^{-3} com CS
17	Nitratos CS	Nitratos em mg dm^{-3} com CS
18	Fosfato CS	Fosfato total em mg dm^{-3} com CS
19	Cobre CS	Cobre total em mg dm^{-3} com CS
20	Coliformes CS	Coliformes totais com CS
21	Temp ar 3m	Temperatura média do ar a 3m em ºC
22	Temp max 3m	Temperatura máxima média do ar a 3m em ºC
23	Temp max elevada 3m	Temperatura máxima mais elevada do ar a 3m em ºC
24	Temp max baixa 3m	Temperatura máxima mais baixa do ar a 3m em ºC
25	Temp min 3m	Temperatura mínima média do ar a 3m em ºC
26	Temp min elevada 3m	Temperatura mínima mais elevada do ar a 3m em ºC
27	Temp min baixa 3m	Temperatura mínima mais baixa do ar a 3m em ºC
28	Precip media	Precipitação média / mm m^{-2}
29	Vel do vento 3m	Velocidade média do vento a 3m em $m.s^{-1}$
30	Direc vento	Direcção média do vento
31	Humidade 3m	Humidade relativa média a 3m
32	Radiacao solar	Radiação solar média em W m^{-2}
33	Balanço radiacao	Balanço médio da radiação em W m^{-2}

O aspecto da base de dados em Access é representado na figura seguinte.



4. ESTUDO

4.1. Árvores de Decisão

O R possui uma gama bastante variada de métodos que podem ser usados para obter modelos para problemas de classificação. Um desses métodos é conhecido como árvores de decisão.

Uma árvore de decisão é formada por um conjunto de nós de decisão, perguntas, que permitem a classificação de cada caso. Uma árvore de decisão consiste numa hierarquia de testes a algumas das variáveis envolvidas no problema de decisão. A árvore pode ser lida a partir do teste encontrado na parte superior da mesma, normalmente chamado *nó raiz* da árvore.

Em primeiro lugar vamos carregar a base de dados através da instrução:

```
> library(RODBC)
>
> lig<-odbcConnectAccess("BDALBUFEIRA4.mdb")
>
> bd<-sqlQuery(lig,"select * from qualidade")
>
>
```

Posteriormente elaborámos uns comandos, de modo a ficarmos com duas tabelas. Uma para uma amostra aleatória para os casos de treino e outra para os casos de teste. Assim, definimos uma percentagem de 2/3 para os casos de treino. A amostra poderia não ter sido feita de forma aleatória.

```
> #definir % de casos de treino
> ptr=2/3
>
> #amostra de casos de treino aleatória
> treino<-sample(1:NROW(bd),as.integer(ptr*NROW(bd)))
> #amostra de casos de treino sequencial temporalmente
> #treino<-seq(1,ptr*NROW(bd), by=1)
> dados.treino<-bd[treino,]
> dados.teste<-bd[-treino,]
```

Deste modo, obtemos dois **dataframes** diferentes que dizem respeito a amostras complementares da base de dados original.

Após termos esse conjunto de dataframes, estamos em condições de obter uma árvore de decisão. As seguintes instruções fazem isso mesmo,

```
> library(rpart)
> arvore<-rpart(qualidadeagua ~ .,bd)
>
> #funcao para visualizar a arvore
> mostra.arvore<-function(arvore){
+ plot(arvore)
+ text(arvore,digits=3,font=6)
+ }
> mostra.arvore(arvore1)
```

Resultados:

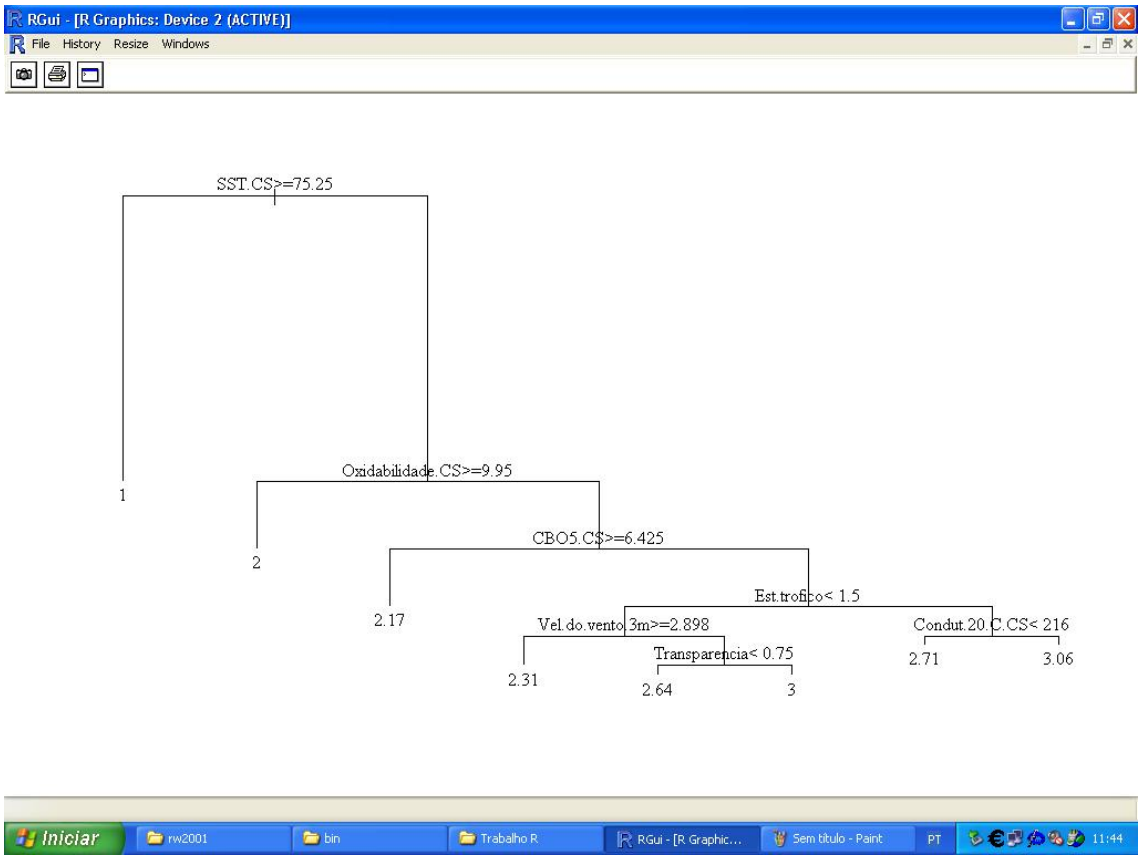
```
n= 108

node), split, n, deviance, yval
      * denotes terminal node

  1) root 108 50.962960 2.518519
    2) SST.CS>=75.25 9  0.000000 1.000000 *
    3) SST.CS< 75.25 99 28.323230 2.656566
      6) Oxidabilidade.CS>=9.95 11  0.000000 2.000000 *
      7) Oxidabilidade.CS< 9.95 88 22.988640 2.738636
       14) CBO5.CS>=6.425 12  1.666667 2.166667 *
       15) CBO5.CS< 6.425 76 16.776320 2.828947
         30) Est.trofico< 1.5 37  8.432432 2.648649
           60) Vel.do.vento.3m>=2.898239 13  2.769231 2.307692 *
           61) Vel.do.vento.3m< 2.898239 24  3.333333 2.833333
            122) Transparencia< 0.75 11  2.545455 2.636364 *
            123) Transparencia>=0.75 13  0.000000 3.000000 *
         31) Est.trofico>=1.5 39  6.000000 3.000000
           62) Condut.20.C.CS< 216 7  3.428571 2.714286 *
           63) Condut.20.C.CS>=216 32  1.875000 3.062500 *
>
```

A primeira destas instruções carrega o *package "rpart"* que contém as funções necessárias para a obtenção de árvores de decisão no R. A segunda instrução obtém um destes modelos usando a função *rpart* colocando o resultado no objecto árvore. A função *rpart* tem dois argumentos principais um indica a forma do modelo a obter, e o outro os dados a usar para o obter. A forma do modelo a obter é fornecida à função usando uma sintaxe genérica de descrição modelos que consiste, sucintamente, em indicar o nome da variável de decisão de seguida do símbolo "_" e uma lista das variáveis que podem ser usadas no

modelo para obter a decisão. Esta lista pode ser substituída pelo símbolo "." (como no exemplo acima), querendo significar que se podem usar todas as outras variáveis existentes nos dados fornecidos a função. Os dados a usar na obtenção do modelo são fornecidos no segundo argumento da função e devem ser um dataframe com as colunas com nomes que estejam de acordo com os nomes de variáveis referidos na forma do modelo.

A seguir a obter a árvore podemos pedir ao R para mostrar o conteúdo do objecto a que atribuímos o resultado. O R apresenta a árvore em forma de texto. Mas se nós quisermos este pode ser apresentado em forma de gráfico como nós podemos ver na figura seguinte:

```
> mostra.arvore<-function(arvore){
+ plot(arvore)
+ text(arvore,digits=3,font=6)
+ }
> mostra.arvore(arvore1)
```



Determinação da Percentagem de Erro do modelo

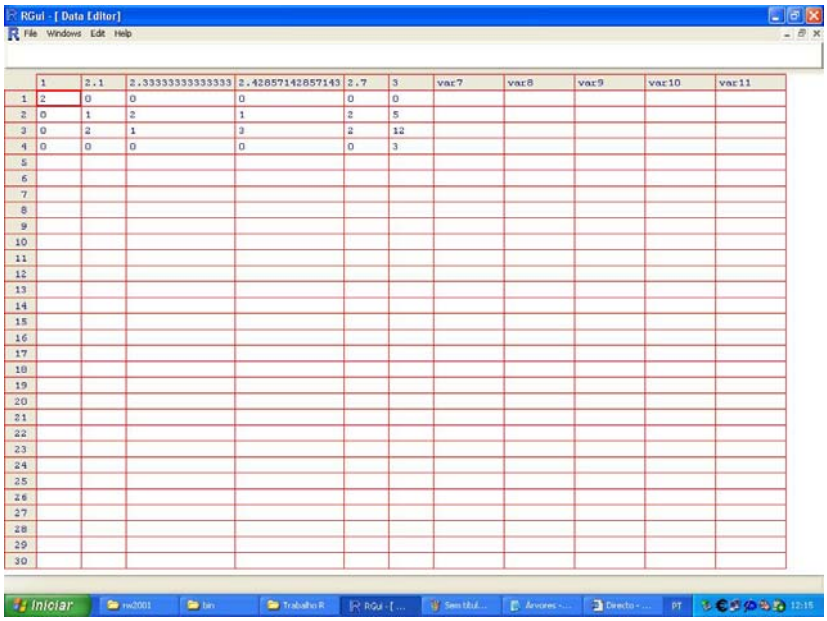
Uma aproximação a esse erro de classificação pode ser obtida utilizando “os exemplos de treino”. Depois de criado o modelo, determinamos o que o modelo proponha para o campo ”classifica” e depois “cruzamos” o que está nos dados com o que foi previsto, fazendo-se a matriz de confusão.

Neste caso concreto a função **predict** produz uma dataframe com tantas linhas quantos casos de teste.

```
> #testar a arvore
> previsoes.modelo<-predict(arvore1,dados.teste)
>
> #matriz de confusao
> matriz.confusao<-table(dados.teste$qualidadeagua,previsoes.modelo)
> #percentagem de erro
> perc.erro<-100*(matriz.confusao[1,2]+matriz.confusao[1,3]+matriz.confusao[1,4]+
+ matriz.confusao[2,1]+matriz.confusao[2,3]+matriz.confusao[2,4]+
+ matriz.confusao[3,1]+matriz.confusao[3,2]+matriz.confusao[3,4]+
+ matriz.confusao[4,1]+matriz.confusao[4,2]+matriz.confusao[4,3])/sum(matriz.confusao)
>
>
>
> avalia.arvore<-function(arv,dados.teste,objectivo=ncol(dados.teste)){
+ prevs<-predict(arv,dados.teste)
+ #previsoes.modelo<-predict(arv,dados.teste)
+ #matriz.confusao<-table(dados.teste$qualidadeagua,previsoes.modelo)
+ matriz.confusao<-table(dados.teste[,objectivo],predict(arv,dados.teste))
+ erro<-100*sum(matriz.confusao[col(matriz.confusao)!=row(matriz.confusao)])/sum(matriz.confusao)
+ list(previsoes=prevs,matriz.confusao=matriz.confusao,perc.erro=erro)
+ }
> resultados<-avalia.arvore(arvore1,dados.teste,objectivo=2)
```

```
       2        3        7       11       14       16       22       24       26       28       33
2.000000 3.040000 3.040000 2.000000 2.000000 3.040000 2.615385 2.615385 3.040000 2.615385 1.500000
      34       35       40       41       43       44       48       50       51       56       57
3.040000 3.040000 3.040000 2.615385 1.500000 1.500000 2.615385 2.250000 3.040000 2.250000 2.615385
      59       61       63       64       66       70       74       75       84       86       92
3.040000 3.040000 3.040000 3.040000 2.615385 3.040000 2.000000 3.040000 3.040000 3.040000 3.040000
      95       99      100
3.040000 2.714286 1.500000
```

No primeiro caso de teste a função produz a probabilidade de cada decisão. Assim, por exemplo, no primeiro caso de teste a árvore prevê a qualidade da água com 2.0 com 100% de confiança.



A matriz de confusão quantifica quantos exemplos da base de dados utilizada seriam classificados bem pelo modelo construído (representado na diagonal principal) sendo que os outros seriam mal classificados.

4.2 Regressão Linear

Nesta secção vamos ver como podemos obter modelos de regressão linear múltipla, usando o R. Este tipo de modelos são frequentemente usados para problemas de regressão. Sem entrarmos em detalhes técnicos referentes às condições de aplicabilidade deste tipo de modelos.

Para obter um modelo de regressão linear em R podemos usar a função **lm**.

```
> modelo.linear<-lm(qualidadeagua ~ .,dados.treino)
> prev.lm<-predict(modelo.linear,dados.teste)
> #matriz de confusao
> matriz.confusao2<-table(dados.teste$qualidadeagua,prev.lm)
> #percentagem de erro
> erro2<-100*sum(matriz.confusao2[col(matriz.confusao2)!=row(matriz.confusao2)])/sum(matriz.confusao2)
```

Atente no formato semelhante dos argumentos desta função, quando comparados com os da função **rpart**.

```
> modelo.linear

Call:
lm(formula = qualidadeagua ~ ., data = dados.treino)

Coefficients:
         (Intercept)          Est.trofico            Nivel.alb            Vol.arma         Transparencia
           7.223e+00            1.027e-01           -1.618e-02           -2.374e-05            3.213e-02
     Temp.amostra.CS                   ph         Condut.20.C.CS               SST.CS          Cloretos.CS
          -2.242e-02           -2.259e-01           -7.884e-04           -1.417e-02           -5.857e-04
Oxigenio.dissCAMP.CS   Oxigenio.diss.AB.CS      Oxidabilidade.CS              CBO5.CS    Azoto.amoniacal.CS
           6.948e-03           -3.994e-02           -8.727e-02           -3.223e-02           -1.359e-01
              Azoto          Nitratos.CS           Fosfato.CS             Cobre.CS       Coliformes.CS
           9.510e-02            7.822e-02           -1.951e+00           -1.387e+02           -3.470e-06
          Temp.ar.3m          Temp.max.3m   Temp.max.elevada.3m     Temp.max.baixa.3m          Temp.min.3m
           3.890e-02           -7.621e-02            3.233e-02            4.508e-02            3.489e-02
 Temp.min.elevada.3m     Temp.min.baixa.3m          Precip.media       Vel.do.vento.3m          Direc.vento
          -1.195e-02           -3.212e-03           -5.295e-03           -4.503e-02           -2.660e-07
         Humidade.3m        Radiacao.solar      Balanco.radiacao
           2.189e-02            3.629e-03           -4.333e-03
```

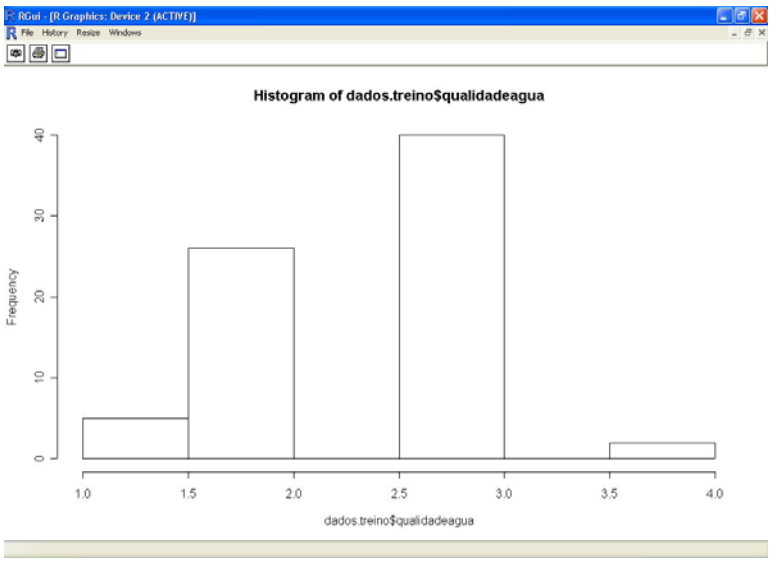
```
> (mad.lm<-mean(abs(prev.lm-dados.teste$qualidadeagua)))
[1] 0.5211393
```

4.3 Árvores de regressão

As árvores de regressão são muito idênticas às árvores de decisão. A diferença principal reside no facto de as folhas das primeiras conterem previsões numéricas e não decisões.

A obtenção de árvores de regressão usando o R é feita através da função **rpart**, tal como nas árvores de decisão. Assim, esta função vai obter uma árvore de regressão ou de decisão consoante o tipo da variável objectivo. Se esta for um factor, a função obtém uma árvore de decisão, se for uma variável numérica é obtida uma árvore de regressão. De resto toda a sintaxe é igual, embora a função possua parâmetros específicos para cada tipo de árvore.

Como optámos por ser uma variável numérica (qualidadeagua) no nosso caso as árvores vão ser iguais. Podemos então concluir que se trata de uma árvore de regressão.




```
> (mad.lm<-mean(abs(previsoes.modelo-dados.teste$qualidadeagua)))
[1] 0.3658069
```

Comparando as previsões, tanto na regressão linear como nas árvores de regressão, é possível observar que, para este problema concreto, as previsões

da árvore de regressão são mais precisas do que as do modelo linear, uma vez que obtêm um erro absoluto médio mais baixo. É de salientar que depois de vários testes os resultados foram sempre parecidos, ou seja, através dos resultados obtidos as árvores de regressão foram sempre mais precisas

4.4 Redes Neuronais

A ferramenta R possui uma quantidade variada de métodos, que podem ser usados para obter modelos para problemas de classificação. Para tal, apresenta um package **nnet** para o uso de redes neuronais, que contém as funções necessárias para a obtenção de redes neuronais no R. Como funções mais importantes, este pacote tem a função *nnet* e a função *predict*.

A função **nnet** poderá treinar a rede de um tipo Unidireccional em que poderá ser constituída por uma só camada ou por várias camadas (multicamada) em que esta ultima só permite apenas uma camada intermédia.

Os principais parâmetros desta função são:

x: uma matriz ou dataframe com as variáveis de entrada na rede

y: uma matriz ou dataframe com as variáveis de saída da rede

size: número de nodos na camada intermédia. Se zero a rede não contem camada intermédia

wts: Um vector com os valores iniciais dos pesos das conexões. Se ausente os pesos serão escolhidos aleatoriamente.

linout: Se verdadeiro a função de activação para as saídas é a linear. Por defeito utiliza a função logística

entropy: Se verdadeiro o método de minimização dos erros é o da máxima verosimilhança. Por defeito utiliza o método dos mínimos quadrados.

softmax: Se verdadeiro utiliza a função log-linear com função de activação e o método da máxima verosimilhança no algoritmo de minimização dos erros. Linout, entropy e softmax são mutuamente exclusivos.

rang: valor que vai determinar o intervalo de valores para inicialização aleatória dos pesos [-rang,rang]

decay: constante de decaimento dos pesos. Serve para evitar o overfitting da rede pois a cada iteração reduz os valores dos pesos da rede.

maxit: número máximo de iterações da rede. 100 é o valor por defeito

```
> rede<-nnet(qualidadeagua ~ .,dados.treino,size=1,rang=1, decay=0.00001, maxit=1000)
# weights:  35
initial  value 381.109270
iter  10 value 208.007647
final  value 208.006032
converged
```

De forma a obter uma matriz com os valores previstos pela rede temos a função **predict**.

```
> rede.previsao<-predict(rede,dados.teste)
```

O resultado desta operação pode ser observado na figura seguinte.

```
> rede.previsao
          [,1]
2   0.9999963
4   0.9999963
8   0.9999963
9   0.9999963
14  0.9999963
17  0.9999963
19  0.9999963
26  0.9999963
```

…

4.5 LDA

A função LDA (Linear discriminant analysis) é uma aproximação estatística que serve para classificar as amostras de classes desconhecidas, baseadas em amostras de treino com classes conhecidas.

Esta função esta disponibilizada no package **MASS**. Deste modo é preciso executar o comando library(MASS) e só então ficam disponíveis as funções relativas a este pacote tais como a função **lda**.

```
> library(MASS)
> analise.linear<-lda(qualidadeagua ~ .,dados.treino)
 > prev.lda<-predict(analise.linear,dados.teste)
```

> analise.linear

Call:

lda(qualidadeagua ~ ., data = dados.treino)

Prior probabilities of groups:

1	2	3	4
0.06944444	0.34722222	0.54166667	0.04166667

Group means:

	Est.trofico	Nivel.alb	….
1	1.800000	196.0000	….
2	1.520000	194.8100	….
3	1.794872	194.6085	….
4	2.333333	195.6667	….

Coefficients of linear discriminants:

	LD1	LD2	LD3
Est.trofico	-1.139248e-01	5.880850e-01	8.852607e-02
Nivel.alb	-2.333443e-01	1.918021e+00	5.525737e+00
Vol.arma	6.307159e-06	-9.984450e-04	-2.739511e-03

…

Proportion of trace:

LD1	LD2	LD3
0.7709	0.1458	0.0833

> prev.lda

$class

[1] 2 3 3 3 3 2 1 4 2 2 2 1 3 2 2 2 3 3 3 3 3 4 3 3 3 2 2 2 3 3 3 3 3 3 1 1

Levels: 1 2 3 4

$posterior

	1	2	3	4
2	6.796902e-18	9.859305e-01	1.406948e-02	9.312405e-12
4	2.975280e-18	2.014205e-01	7.985777e-01	1.747715e-06
8	1.193400e-22	3.004930e-02	9.699503e-01	3.657084e-07

…

$x

	LD1	LD2	LD3
2	0.13195872	-3.4107699	0.326453464
4	0.74105022	-0.2096283	-0.446370412
8	1.86664878	-0.1206564	-1.486227761

5. CONCLUSÃO

Concluído este trabalho de exploração, cumpre-nos reflectir sobre o valor e potencialidade desta ferramenta R. Evidentemente que aquilo que, para já, nos apraz dizer é que as funcionalidades que explorámos foram muito reduzidas, se tivermos em linha de conta as capacidades que um programa desta natureza comporta. Além disso, o caudal de conhecimento que possuímos sobre essa mesma ferramenta é ainda bastante ténue. No entanto, o nosso objectivo ao delinear e escrever algo sobre esta temática foi antes, e acima de tudo, um esforço de familiarização com este programa, de modo a percepcionarmos as potencialidades ou valências do mesmo.

A primeira característica positiva a notar diz respeito ao facto de estarmos perante uma aplicação que contempla todo o tipo de análises estatísticas usadas. Isto quer dizer que é possível realizar qualquer análise estatística usando esta ferramenta. Para tal, muito contribui o facto de se tratar de uma aplicação de código aberto e usada por uma comunidade cada vez mais alargada de utilizadores por todo o mundo, que vão contribuindo para o desenvolvimento de novas e sucessivas funcionalidades.

Outra característica positiva, é a possibilidade de se escrever o código e guardá-lo num ficheiro de texto, para posterior execução. Isto abre espaço, à criação de novas funções, que podem ser auxiliares para tratamento de dados e código, com uma sequência de funções a executar, e guardar todo esse código num ficheiro. Se inicialmente podemos ser levados a despender algum tempo e esforço na criação desse código, os ganhos futuros, em termos de tempo, e portanto, de rentabilidade, serão substanciais, se essas tarefas tiverem de ser executadas repetidamente, ou houver possibilidade de reutilização das mesmas.

Se atendermos às funcionalidades gráficas do programa, embora não exploradas na sua verdadeira e completa dimensão, neste pequeno trabalho,

elas são sobejamente aliciantes, de acordo com o que pudemos constatar, através da consulta da documentação fruto da pesquisa conseguida.

A existência de bastante documentação de apoio ao uso da ferramenta é outro dos pontos fortes a ter em consideração. Por outro lado, a aplicação inclui de base muitos documentos de suporte ao uso da ferramenta e a função *help* permite obter informação em relação a todas as funcionalidades existentes. Por outro lado, se pesquisarmos na Internet encontraremos a publicação de muitos trabalhos que fazem uma abordagem sistemática a variadíssimos problemas que se ligam à utilização do R.

Para lá destes pontos fortes aqui enunciados, importa salientar o facto de se tratar de uma ferramenta que nos é facultada gratuitamente nos campos da Internet, o que significa, por si só, a possibilidade de um elenco muito grande de informação que nos pode orientar fecundamente, ao longo da utilização da aplicação.

O único factor que, à partida, poderá ser considerado como negativo é o facto do interface com o utilizador ser efectuado através de linhas de código e não por meio de ambiente gráfico. Isso obriga a uma familiarização com o nome das funções e os seus parâmetros, bem como a conhecimentos avançados da estrutura dos dados que as funções recebem como argumentos, para assim se poder evitar, ou pelo menos diminuir, o aparecimento de erros. É precisamente esta razão que faz com que o processo de adaptação à ferramenta seja mais lento do que uma outra aplicação em que o interface gráfico seja por objectos. No entanto, aquilo que nos afigura afirmar a este propósito, é que à medida que ganhamos experiência com o uso da ferramenta, a sua utilização torna-se mais intuitiva e simples, e essa barreira acaba por se atenuar ou mesmo desaparecer.

Uma palavra para o package eZmining. Tivemos imensas dificuldades em instalar pois a sua versão era incompatível com a versão actual do R. Contactámos com o autor do package e mesmo assim não conseguimos instalar. Só mais tarde, depois de termos uma versão compilada com a nova

versão do R é que conseguimos pôr o package a funcionar. Já não foi mau de todo.

Em resumo, a nossa apreciação global da ferramenta é francamente positiva. O R constitui sem dúvida um excelente meio de análise de dados, podendo ser utilizado profissionalmente para este fim sem perda de qualidade, em relação a outras ferramentas de carácter comercial existentes.

BIBLIOGRAFIA

Brown, M.L., Kros, J.F. (2003) *Data Mining and the Impact of Missing Data.*

Cortez, P., *Análise Inteligente de Dados*, Unidade 2: Estatística para a aprendizagem.

Cortez, P., *Análise Inteligente de Dados*, Unidade 3: Redes Neuronais Artificiais.

Cortez, P., *Análise Inteligente de Dados*, Unidade 4:Estudo de Caso com Redes Neuronais.

Dantas, R.P. (2004) *Exploração e Análise de Ferramentas de Data Mining: Tanagra, R, Analyze*.

Dantas, R.P. (2004) *ezMining: Um Package de Data Mining para o R.*

Faraway, J.J. (2002) *Practical Regression and Anova using R.*

Goeble, M., Gruenwald, L. (1999) *A Survey of Data Mining and Knowledge Discovery Software Tools*, SIGKDD Explorations, ACM SIGKDD, Vol. 1, Issue 1, 06/99.

Gonçalves, J.A. (2003/2005) *Exploração da Ferramenta R.*

Gonçalves, J.A. (2003/2005) *Previsão com Redes Neuronais usando a Ferramenta R.*

Lee, S.J., Siau, K. (2001) *A Review of Data Mining Techniques.*

Maindonald J.H. (2001) *Using R for Data Analysis and Graphics. An Introduction.*

Mitra, S., Sankar, K., Mitra, P. (2002) *Data Mining in Soft Computing Framework: A Survey*, IEEE Transactions on Neural Networks, Vol. 13, Nº 1, 01/02

R Project, Writing R extensions (ficheiro R-exts.pdf, disponível na directoria doc\manual do R após instalação, ou em http://cran.r-project.org/doc/manuals/R-exts.pdf).

Torgo, L. (2003) Data Mining with R: learning by case studies, Porto.

Torgo, L. (2003) *Programação, Análise de Dados e Sistemas de Apoio à Decisão usando o R*.

Venables W.N., Ripley B.D. (1999) *Modern Applied Statistics With S-Plus*, Third Edition, Springer.

Venables, W. and Smith, D.M. and the R Development Core Team (2004). *An Introduction to R Versio*n. Notes on R: A Programming Environment for Data Analysis and Graphics.

Widener, T. (1996) *The KDD Process for Extracting Useful Knowledge from Volumes of Data*, Communications of The ACM, Vol. 39, Nº 11, 11/96.

ANEXOS

Código R

```
library(RODBC)

lig<-odbcConnectAccess("BDALBUFEIRA4.mdb")

bd<-sqlQuery(lig,"select * from qualidade")



#definir % de casos de treino
ptr=2/3

#amostra de casos de treino aleatória
treino<-sample(1:NROW(bd),as.integer(ptr*NROW(bd)))
#amostra de casos de treino sequencial temporalmente
#treino<-seq(1,ptr*NROW(bd), by=1)
dados.treino<-bd[treino,]
dados.teste<-bd[-treino,]

fix(bd)

#----------------------------arvores de decisao-------------------------

library(rpart)
arvore<-rpart(qualidadeagua ~ .,bd)

#funcao para visualizar a arvore
mostra.arvore<-function(arvore){
        plot(arvore)
        text(arvore,digits=3,font=6)
}
mostra.arvore(arvore1)


fix(dados.treino)
#teinar a arvore
arvore1<-rpart(qualidadeagua ~.,dados.treino)

fix(dados.teste)
#testar a arvore
previsoes.modelo<-predict(arvore1,dados.teste)

#matriz de confusao
matriz.confusao<-table(dados.teste$qualidadeagua,previsoes.modelo)
#percentagem de erro
perc.erro<-
100*(matriz.confusao[1,2]+matriz.confusao[1,3]+matriz.confusao[1,4]+
                matriz.confusao[2,1]+matriz.confusao[2,3]+matriz.confusao[2,4]+
```

```
                matriz.confusao[3,1]+matriz.confusao[3,2]+matriz.confusao[3,4]+

        matriz.confusao[4,1]+matriz.confusao[4,2]+matriz.confusao[4,3])/sum(ma
triz.confusao)



avalia.arvore<-function(arv,dados.teste,objectivo=ncol(dados.teste)){
                prevs<-predict(arv,dados.teste)
                #previsoes.modelo<-predict(arv,dados.teste)
                #matriz.confusao<-
table(dados.teste$qualidadeagua,previsoes.modelo)
                matriz.confusao<-
table(dados.teste[,objectivo],predict(arv,dados.teste))
                erro<-
100*sum(matriz.confusao[col(matriz.confusao)!=row(matriz.confusao)])/sum(ma
triz.confusao)

        list(previsoes=prevs,matriz.confusao=matriz.confusao,perc.erro=erro)
}
resultados<-avalia.arvore(arvore1,dados.teste,objectivo=2)
(mad.lm<-mean(abs(previsoes.modelo-dados.teste$qualidadeagua)))


#-----------------------------Regressao Linear---------------------------

modelo.linear<-lm(qualidadeagua ~ .,dados.treino)
prev.lm<-predict(modelo.linear,dados.teste)
#matriz de confusao
matriz.confusao2<-table(dados.teste$qualidadeagua,prev.lm)
#percentagem de erro
erro2<-
100*sum(matriz.confusao2[col(matriz.confusao2)!=row(matriz.confusao2)])/sum
(matriz.confusao2)

(mad.lm<-mean(abs(prev.lm-dados.teste$qualidadeagua)))

#-----------------------------Análise linear-----------------------------

library(MASS)
analise.linear<-lda(qualidadeagua ~ .,dados.treino)
prev.lda<-predict(analise.linear,dados.teste)
#matriz de confusao
#matriz.confusao3<-table(dados.teste$qualidadeagua,prev.lda)
#percentagem de erro
#erro3<-
100*sum(matriz.confusao3[col(matriz.confusao3)!=row(matriz.confusao3)])/sum
(matriz.confusao3)
#(mad.lm<-mean(abs(analise.linear-dados.teste$qualidadeagua)))
```

```
#----------------------------Redes Neuronais---------------------------

library(nnet)

#normalizar valor de saida [0,1]
ns<-function(x){
y<-(x-min(x))/(max(x)-min(x))
return(y)
}


#dados de entrada
entradas<-cbind(bd[2],bd[3],bd[4],bd[5],bd[6],bd[7],bd[8],bd[9],bd[10],
        bd[11],bd[12],bd[13],bd[14],bd[15],bd[16],bd[17],bd[18],bd[19],bd[20],
        bd[21],bd[22],bd[23],bd[24],bd[25],bd[26],bd[27],bd[28],bd[29],bd[30],
        bd[31],bd[32],bd[33])

#dados de saida
saidas<-bd[1]


rede<-nnet(qualidadeagua ~ .,dados.treino,size=1,rang=1, decay=0.00001,
maxit=1000)
rede.previsao<-predict(rede,dados.teste)

#matriz de confusao
matriz.confusao4<-table(dados.teste$qualidadeagua,rede.previsao)
#percentagem de erro
erro4<-
100*sum(matriz.confusao4[col(matriz.confusao4)!=row(matriz.confusao4)])/sum
(matriz.confusao4)
```