Universidade do Minho
Mestrado Integrado em Engenharia Informática
Departamento de Informática

Sistemas de Representação de Conhecimento e Raciocínio

Exercício 3
Conhecimento não simbólico: Redes Neuronais Artificiais

Grupo 4
Isabel Sofia da Costa Pereira A76550
José Francisco Gonçalves Petejo e Igreja Matos A77688
Maria de La Salete Dias Teixeira A75281
Tiago Daniel Amorim Alves A78218

20 de Maio de 2018

1 Resumo

Ao longo deste projeto foi realizado o tratamento e análise de dois *data-sets* que permitem classificar um vinho pela sua qualidade a partir de Redes Neuronais Artificiais. Assim, neste relatório é descrito todo o trabalho desenvolvido, desde as decisões tomadas no tratamento dos dados até à topologia de rede considerada no final, bem como o seu treino.

Conteúdo

2 Introdução

Este trabalho centra-se na utilização de sistemas não simbólicos na representação de conhecimento e no desenvolvimento de Redes Neuronais Artificiais, RNAs, para determinar os diferentes níveis de qualidade de vinhos. Os níveis de qualidade estão numerados de 0 a 10, sendo determinados por uma fórmula que utiliza como parâmetros algumas propriedades referentes ao vinho, tais como o teor alcoólico e os açúcares residuais.

Assim, é necessário manipular os dados fornecidos e definir várias topologias de rede até se alcançar a rede com menor dispersão em torno da linha de regressão. Para além disso, é também necessário analisar as melhores regras de aprendizagem para o treino da rede.

Para a elaboração deste projeto utilizou-se a linguagem R.

3 Preliminares

Para a realização deste trabalho é essencial perceber em que consistem as RNAs e como funcionam, sendo o mais importante compreender que estas são capazes de aprender a gerar conhecimento a partir de conhecimento adquirido, neste caso por *datasets*, juntamente com um processo de aprendizagem. Tendo isto em mente, é possível desenvolver algoritmos de aprendizam e manipular dados para aumentar a eficácia da rede.

Para além disso, é também essencial a análise dos dados fornecidos, pois só percebendo em que consistem os *datasets* e como estão apresentadas as diferentes propriedades é que é possível transformar corretamente os dados e melhorar a aprendizagem da rede.

Sendo que neste caso os *datasets* são relativos à qualidade de vinhos, os atributos apresentados referem-se a diferentes propriedades dos vinhos. Assim, será analisado o formato em que estas são apresentadas bem como a sua importância na qualidade de cada vinho.

4 Descrição do Trabalho e Análise dos Resultados

4.1 Considerações Iniciais do Trabalho

O tema proposto, *Wine Quality*, engloba dois *datasets*, sendo o primeiro relativo a vinhos tintos e o segundo a vinhos brancos. Cada um dos *datasets* apresenta onze variáveis de *input* e uma variável de *output*.

As variáveis de *input* são:

- *fixed acidity* - soma dos ácidos fixos do vinho;
- *volatile acidity* - soma dos ácidos voláteis do vinho que, por lei, não pode ultrapassar 1.2 gr/L, sendo um indicador da qualidade do vinho;
- *critic acid* - ácido que proporciona acidez fresca ao vinho;
- *residual sugar* - açúcar que permanece no vinho após o processo de fermentação;
- *chlorides* - quantidade de sal no vinho;
- *free sulfur dioxide*
- *total sulfur dioxide*
- *density* - medida de concentração de açúcares;
- *pH* - afeta o aspeto visual, aroma, paladar e longevidade do vinho;
- *sulphates*
- *alcohol* - percentagem de álcool presente no vinho.

A variável de *output*, *quality*, refere-se ao nível de qualidade do vinho, que é classificado de 0 a 10, sendo 0 o nível de qualidade mais baixo e 10 o mais alto.

Sendo que possivelmente existem diferenças entre os vinhos brancos e tintos, deve-se ter em consideração os dois *datasets* para avaliar a qualidade dos mesmos. Assim, para que a RNA a desenvolver analise corretamente as informações, optou-se por adicionar uma nova variável de *input* a cada *dataset*,

à qual se deu o nome de *wine color*, para de seguida se unir a informação de ambos. Nos vinhos tintos esta variável toma o valor 1 e nos vinhos brancos toma o valor 2.

Tendo-se unido os *datasets*, para a RNA não se tornar tendenciosa, foram também randomizados os valores, de maneira a que os vinhos brancos/tintos não estejam todos seguidos, o que aconteceria se apenas se unisse os *datasets*.

Para a realização destes passos, implementou-se o seguinte código, em que a variável dadosR corresponde aos dados dos vinhos tintos e a variável dadosW corresponde aos dados dos vinhos brancos.

```
dadosR <- cbind(wine.color = 1, dadosR)
dadosW <- cbind(wine.color = 2, dadosW)
dadosRW <- rbind(dadosR, dadosW)
dados <- dadosRW[sample(nrow(dadosRW)),]
```

4.2 Identificação dos Atributos mais Significativos

O primeiro passo para construir uma Rede Neuronal Artificial passa por determinar os atributos mais significativos do problema em questão. Tal é essencial porque, sabendo os atributos que são realmente importantes para determinar o *output*, é mais fácil de criar uma rede eficaz.

Tomando o caso de estudo em questão como exemplo, é dito expressamente na descrição dos *dataset* que não se sabe se todas as variáveis de *input* são relevantes para o problema. Então, para identificar as variáveis mais relevantes, aplicou-se o seguinte código:

```
funcao <- quality ~ wine.color + fixed.acidity +
 ↪  volatile.acidity + citric.acid + residual.sugar + chlorides
 ↪  + free.sulfur.dioxide + total.sulfur.dioxide + density + pH
 ↪  + sulphates + alcohol

selecao <- regsubsets(funcao,dados,method = "forward")
summary(selecao)

selecao <- regsubsets(funcao,dados,method = "backward")
summary(selecao)
```

Como apresentado acima, começou-se por definir uma função que considera todas as variáveis de *input* e, de seguida, aplicou-se o *regsubsets* a esta função juntamente com os dados do problema.

Relativamente ao *regsubsets*, optou-se por aplicar os métodos *forward* e *backward*, no entanto os resultados foram equivalentes. Analisando estes resultados, é evidente que, quando o objetivo é determinar a qualidade de um vinho, a variável mais significativa é o *alcohol*, seguindo-se a *volatile acidity*, os *sulphates*, o *residual sugar*, o *wine color*, a *density*, o *free sulfur dioxide* e por último o *total sulfur dioxide*.

```
> selecao <- regsubsets(funcao,dados,method = "backward")
> summary(selecao)
Subset selection object
Call: regsubsets.formula(funcao, dados, method = "backward")
12 Variables  (and intercept)
                     Forced in Forced out
wine.color               FALSE      FALSE
fixed.acidity            FALSE      FALSE
volatile.acidity         FALSE      FALSE
citric.acid              FALSE      FALSE
residual.sugar           FALSE      FALSE
chlorides                FALSE      FALSE
free.sulfur.dioxide      FALSE      FALSE
total.sulfur.dioxide     FALSE      FALSE
density                  FALSE      FALSE
pH                       FALSE      FALSE
sulphates                FALSE      FALSE
alcohol                  FALSE      FALSE
1 subsets of each size up to 8
Selection Algorithm: backward
         wine.color fixed.acidity volatile.acidity citric.acid residual.sugar chlorides free.sulfur.dioxide total.sulfur.dioxide density pH  sulphates alcohol
1  ( 1 ) " "        " "           " "              " "         " "            " "       " "                " "                " "     " " " " "       "*"
2  ( 1 ) " "        " "           "*"              " "         " "            " "       " "                " "                " "     " " " " "       "*"
3  ( 1 ) " "        " "           "*"              " "         " "            " "       " "                " "                " "     " " "*"       "*"
4  ( 1 ) " "        " "           "*"              " "         "*"            " "       " "                " "                " "     " " "*"       "*"
5  ( 1 ) "*"        " "           "*"              " "         "*"            " "       " "                " "                " "     " " "*"       "*"
6  ( 1 ) "*"        " "           "*"              " "         "*"            " "       " "                " "                "*"     " " "*"       "*"
7  ( 1 ) "*"        " "           "*"              " "         "*"            " "       " "                " "                "*"     "*" "*"       "*"
8  ( 1 ) "*"        "*"           "*"              " "         "*"            " "       " "                " "                "*"     "*" "*"       "*"
>
```

Figura 1: Resultado do *regsubsets*

Tendo em conta que se tem, no total, 12 variáveis de *input* e 8 destas como as mais significativas, começou-se então por desenvolver uma rede, para determinar quais as variáveis a considerar. Para isso, foi-se adicionando e/ou removendo variáveis da fórmula considerada na RNA, recorrendo à *neuralnet*, até se alcançar a combinação de variáveis que geram o menor RMSE.

O RMSE, *root-mean-square error*, corresponde ao erro quadrático médio, sendo uma das maneiras existentes para determinar a rede com menor dispersão em torno da linha de regressão. Assim, quanto menor o RMSE, menor a dispersão.

Tendo em conta os testes efetuados com as várias variáveis, concluiu-se que a fórmula mais indicada para o problema seria considerando as seguintes variáveis:

```
formula <- quality ~ alcohol + volatile.acidity + sulphates +
↪  residual.sugar + wine.color
```

4.3 Transformação dos Dados

Outro fator decisivo na formulação de uma RNA é o estado em que os dados são apresentados. Por vezes, os *datasets* fornecidos para a resolução dos problemas não apresentam as variáveis nos tipos mais favoráveis, aparecendo, por exemplo, em forma de *Strings* ou até em escalas de valores muito extensas. Sendo as RNA muito sensíveis, este género de valores resultam em erros mais acentuados, ou seja, numa dispersão maior dos valores (RMSE maiores). A fim de dar a volta a este problema, é usual recorrer à transformação, também denominada de normalização, dos dados.

No caso dos *datasets* a ser utilizados, apesar de não haver valores com tipos complexos, têm gamas de valores extensas.

Sabendo que o programa apresenta resultados melhores com valores enquadrados numa gama de 0 a 1, começou-se por testar a rede com os valores normalizados neste intervalo, utilizando o seguinte código:

```
dados$volatile.acidity <- dados$volatile.acidity * 0.1
dados$residual.sugar <- dados$residual.sugar * 0.01
dados$sulphates <- dados$sulphates * 0.1
dados$alcohol <- dados$alcohol * 0.01
```

Por praticidade, tendo em conta que a RNA só utiliza os atributos considerados na fórmula, apenas se transformou os atributos presentes na mesma. Para além disso, é correto apenas normalizar variáveis que não se encontrem de início no intervalo de 0 a 1.

Para determinar o valor a multiplicar pelos atributos, recorreu-se aos comandos existentes em R *max()* e *min()* aplicados a um vetor.

Para além desta forma de normalizar dados, testou-se também a discretização dos atributos, ou seja, dividir os dados de um atributo que apresentam uma distribuição contínua em intervalos de valores. Por norma, esta ação também leva à diminuição do erro, no entanto, este tipo de normalização não é aplicado a todas as variáveis, mas apenas aquelas que apresentam uma quantidade extensa de valores únicos. Nesta caso, concluiu-se que só era justificável aplicar discretização ao *residual sugar*, que apresenta o maior número de valores diferentes, 316, comparativamente às outras variáveis.

```
residual.sugar <- discretize(dados$residual.sugar, method =
↪  "frequency", categories = 5, labels = c(1,2,3,4,5))

dados$residual.sugar <- as.numeric(residual.sugar)
```

Por último, testou-se também a normalização dos dados num intervalo de 0 a 1 aplicando a fórmula *Feature scaling* que é apresentada de seguida.

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}$$

Sendo uma fórmula estatística própria para a normalização de dados, considerou-se que poderia ser vantajoso utilizar os dados desta forma. Assim, foi desenvolvida a seguinte função em R.

```
normalizar <- function(arg1){
        arg1 <- (arg1 - min(arg1) / (max(arg1) - min(arg1))
        return(arg1)
}
```

Analisando todos os testes realizados para os três diferentes tipos de transformação de dados considerados, verificou-se que os valores obtidos para o RMSE são bastante semelhantes, alterando-se apenas a partir da casa das milésimas. Apesar de a diferença ser bastante pequena, no geral, a normalização que apresentou RMSE ligeiramente melhor foi a primeira, ou seja, a mudança da escala dos dados, daí se ter considerado esta transformação para a RNA final.

4.4 Topologia de Rede

Para determinar a melhor topologia de rede, teve-se em consideração o número de camadas intermédias, bem como o respetivo número de neurónios, o *treshold* e o algoritmo utilizado para calcular a rede neuronal. Desta forma, foi-se alterando vários fatores e testando diferentes combinações.

Relativamente ao número de camadas intermédias e número de neurónios nestas, foram realizados testes para uma camada com 5, 4 e 3 neurónios e para duas camadas com (5,4), (4,3) e (3,2) neurónios, sendo que o primeiro valor corresponde aos neurónios na primeira camada e o segundo aos neurónios na segunda. Estes valores foram considerados porque o número de camadas e neurónios ideais, por norma, estão no intervalo de valores entre o número de variáveis de *input* e o número de variáveis de *output*.

Quanto ao *treshold*, foram realizados testes para *treshold* igual a 0.1 e 0.01. Este valor foi variando de acordo com o que a rede permitia. Isto é, primeiro testou-se a rede com *treshold* igual a 0.1 e, caso a rede convergisse, testava-se com *treshold* igual a 0.01.

Tendo em conta que existem vários algoritmos para cálculo da rede, foram realizados testes considerando os seguintes algorimos:

- *rprop+*
- *rprop-*
- *slr*
- *sag*

No geral, os melhores resultados foram obtidos com o algoritmo *rprop+*.

Analisando todos estes fatores, no final, a rede que apresentou melhores resultados foi definida por uma camada intermédia com 4 neurónios, *treshold* igual a 0.1 e algoritmo *rprop+*.

4.5 Treino da Rede Neuronal

Para a RNA funcionar é necessário que esta seja treinada. Para isso, deve-se dividir os dados em dois conjuntos diferentes, sendo o primeiro o conjunto de dados para treino e o segundo o conjunto de dados para teste. Tal como o nome indica, o segundo conjunto é utilizado para testar a eficácia da Rede Neuronal.

A eficácia da rede também é sensível à quantidade de dados utilizada para o treino da mesma. Assim, foram testados vários valores até se concordar num valor que resulta no menor RMSE. É também importante referir que, alterando as restantes características da rede, também pode ser necessário alterar este valor para diminuir o RMSE.

O seguinte excerto de código apresenta um exemplo dos vetores treino e teste.

```
treino <- dados[1:2500, ]
teste <- dados[2501:6497, ]
```

4.6 Análise dos Resultados

Tendo em conta todos os fatores considerados neste relatório, realizaram-se alguns testes para determinar a Rede Neuronal Artificial mais adequada, sendo esta a que apresenta menor RMSE. Os testes foram realizados à medida que se foram alterando especificações do programa.

Nas seguintes tabelas apresentam-se os resultados obtidos em apenas alguns dos testes realizados.

Treino	**Nodos**		***Treshold***	**Algoritmo**	**RMSE**
	Camada 1	**Camada 2**			
[1, 2500]	4	-	0.1	rprop+	0.742118547
[1, 3000]	4	-	0.1	rprop+	0.7408157954
[1, 2500]	4	-	0.1	rprop-	0.7428453823
[1, 2500]	4	-	0.1	slr	0.7429822216
[1, 2500]	3	2	0.1	rprop+	0.741783797
[1, 3000]	3	2	0.1	rprop+	0.7465901619
[1, 2500]	5	4	0.1	-	0.7421187683
[1, 2500]	5	-	0.1	-	0.7483426603

Tabela 1: Redes Neuronais Artificiais testadas

Pela análise das tabelas, e tal como já se tinha referido nos tópicos anteriores, pode-se concluir que a rede mais eficaz para determinar a qualidade de vinhos é a que apresenta as seguintes especificações:

```
dados$volatile.acidity <- dados$volatile.acidity * 0.1
dados$residual.sugar <- dados$residual.sugar * 0.01
dados$sulphates <- dados$sulphates * 0.1
dados$alcohol <- dados$alcohol * 0.01

treino <- dados[1:3000, ]
teste <- dados[3001:6497, ]

formula <- quality ~ alcohol + volatile.acidity + sulphates +
↪  residual.sugar + wine.color
```

```
rna <- neuralnet(formula,treino, hidden = c(4), lifesign =
↪  "full", linear.output = TRUE, threshold = 0.1, algorithm =
↪  "rprop+")
```

Desta forma, a Rede Neuronal Artificial gerada foi a seguinte.

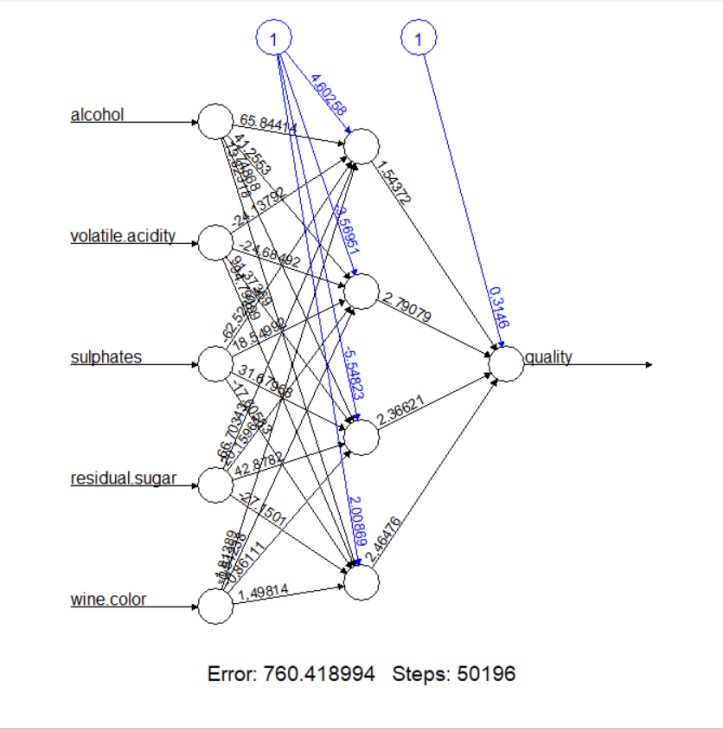

Figura 2: Resultado do *regsubsets*

5 Conclusões e Sugestões

A elaboração deste trabalho permitiu-nos aprofundar a nossa compreensão de conhecimento não simbólico, mais especificamente de Redes Neuronais Artificiais. Para além disso, conseguimos igualmente aprofundar conhecimentos na linguagem R.

No desenvolvimento do projeto, tivemos em consideração vários fatores que afetam a RNA, como diferentes maneiras de normalizar os dados e diferentes algoritmos para cálculo da rede. No entanto, tendo em conta a imensidão de possibilidades na criação de redes neuronais, podemos apontar como trabalho futuro testar a rede com outras formas de transformação dos dados, diferentes valores de *treshold*, outros algoritmos de cálculo da rede e mesmo fatores que não foram considerados pelo grupo.

Tendo em conta os objetivos do projeto, consideramos que realizamos um trabalho bem sucedido e que, apesar de não ficarmos completamente satisfeitos com o RMSE obtido, conseguimos adquirir os conhecimentos esperados para a unidade curricular.