UNIVERSIDADE DO MINHO

Redes Neuronais Artificiais

Mestrado Integrado em Engenharia Informática

Sistemas de Representação de Conhecimento e Raciocínio
(2º Semestre – 2017/2018)

Grupo 23

A61855 Ana Paula Carvalho
A73831 João Pires Barreira
A61799 Rafael Braga Costa

Braga,
Maio 2018

Resumo

De modo a pôr em prática a última matéria abordada apresenta-se, ao longo deste relatório, o processo elaboração de uma solução baseada em *Redes Neuronais Artificiais* utilizando dados facultados referentes a *Bank Marketing* recorrendo à linguagem de programação *R*.

Conteúdo

1 Preliminares

Para a realização deste exercício, revelou-se necessário a aquisição de uma maior destreza na aplicação da linguagem de programação *R*, alcançada através de um estudo prévio recorrendo a algumas pesquisas *web*.

Todos os conhecimentos lecionados nas aulas referentes à matéria de Redes Neuronais manifestaram igualmente a sua importância, se os quais muito dificilmente se conseguiria prosseguir e concluir o trabalho com sucesso.

2 Introdução

Para a realização deste terceiro trabalho prático da Unidade Curricular de Sistemas de Representação de Conhecimento e Raciocínio foi proposto o desenvolvimento de mecanismos de raciocínio, nomeadamente através do desenvolvimento de *Redes Neuronais Artificiais* (*RNAs*), utilizando um *Dataset* facultado para esta análise. No caso do nosso grupo, os dados a serem utilizados correspondem aos resultados de campanhas de *telemarketing* por parte de um banco português.

Ao longo deste relatório irá ser explicitada a nossa implementação em *R* e apresentada a nossa solução indicando os atributos mais significativos de cada cliente, a topologia mais adequada da rede desenvolvida, as regras de aprendizagem para treinar a rede, etc.

3 Descrição do Trabalho e Análise dos Resultados

Além da rede sugerida, que consiste em prever se um cliente irá subscrever ou não a um depósito a prazo, foi desenvolvida uma outra rede sobre um tema que considerámos relevante. Esta nova rede consiste em determinar se um cliente é, ou não, um bom candidato a um tipo de campanha visada a clientes que não possuem qualquer tipo de empréstimos por pagar e saldo positivo.

Nesta secção explica-se, com detalhe, todos os passos efetuados no processo de construção das duas redes mencionadas.

3.1 Subscrição a um depósito a prazo

Antes de se proceder à construção da rede propriamente dita, foram analisados o ficheiro "bank-full.csv", que contém os dados propriamente ditos, e o ficheiro "bank-names.txt", que descreve os valores dos diferentes atributos envolvidos.

Deste modo, verificou-se que os diferentes atributos, e respetivos domínios, eram os seguintes:

- **age** - valor inteiro
- **job** - cadeia de carateres que pode conter um dos seguintes valores: "admin.", "unknown", "unemployed", "management", "housemaid", "entrepreneur", "student", "blue-collar", "self-employed", "retired", "technician" ou "services".
- **marital** - cadeia de carateres que pode conter um dos seguintes valores: "married", "divorced" ou "single".
- **education** - cadeia de carateres que pode possuir um dos seguintes valores: "unknown", "secondary", "primary" ou tertiary".
- **default** - cadeia de carateres que pode possuir o valor "yes" ou o valor "no".
- **balance** - valor inteiro
- **housing** - cadeia de carateres que pode possuir o valor "yes" ou o valor "no".
- **loan** - cadeia de carateres que pode possuir o valor "yes" ou o valor "no".
- **contact** - cadeia de carateres que pode possuir um dos seguintes valores: "unknown", "telephone" ou "cellular".
- **day** - valor inteiro.
- **month** - cadeia de carateres que pode possuir um dos seguintes valores: "jan", "fev", "mar", "apr", "may", "jun", "jul", "aug", "sep", "oct", "nov" ou "dec".
- **campaing** - valor inteiro.
- **pdays** - valor inteiro.
- **previous** - valor inteiro.
- **poutcome** - cadeia de carateres que pode possuir um dos seguintes valores: "unknown", "other", "failure" ou "success".
- **y** - cadeia de carateres que pode possuir o valor "yes" ou o valor "no".

Através da determinação dos domínios dos diferentes atributos pôde-se, então, proceder-se à normalização dos dados. Os atributos que correspondem a uma cadeia de carateres com diferentes possibilidades devem ser convertidos em índices (por exemplo o atributo *month* deve ser convertido num valor inteiro pertencente ao intervalo $[1, 12]$. Valores que podem ser do tipo "no" devem ser convertidos para 0 e valores do tipo "yes" para 1. Valores inteiros devem ser reduzidos a uma escala muito pequena, visto as redes neuronais lidarem muito melhor com valores entre as gamas $[0, 1]$ ou $[-1, 1]$, do que $[0, 1000]$, por exemplo.

De modo a tratar do caso da normalização de valores inteiros, foi criada uma função em *R*, que se baseia na tecnica de normalização chamada *max min*. Esta função recebe como argumento um vetor e devolve o mesmo vetor normalizado segundo esta técnica. A função é a seguinte:

```
normalize <- function(x) { (x - min(x)) / (max(x) - min(x))}
```

No caso das cadeias de carateres foi criado um vetor auxiliar com todas as opções possíveis e, dessa maneira, obter-se o índice correto. O seguinte exemplo ilustra este processo para o caso do atributo *months*:

```
# Construcao do vetor de possibilidades
months <- c("jan","feb","mar","apr","may","jun",
    "jul","aug","sep","oct","nov","dec")

# Normalizacao
dados$months <- match(dados$months, months)
```

Em que *dados* corresponde à *data.frame* que contém todos os dados do ficheiro "bank-full.csv". De notar a função *match* fornecida pelo *R* que devolve o índice do vetor *months* criado, sobre o qual houve uma ocorrência nos valores dos *months* pertencentes aos *dados*. No entanto, como se pode verificar, o resultado destas operações produz valores pertencentes ao intervalo $[1, 12]$, sendo que o ideal seria valores entre $[0, 1]$. Para que isto aconteça basta invocar a função *normalize* mencionada anteriormente.

Finalmente, para normalizar valores do tipo "no" ou "yes" basta efetuar os seguintes passos:

```
# Construcao do vetor de valores booleanos
booleans <- c("no", "yes")

# Normalizacao
dados$default <- match(dados$default, booleans) - 1
```

No exemplo ilustrado acima procede-se à normalização de valores correspondentes ao atributo *default*. Como se pode verificar os valores do tipo "no" são convertidos para 0 e os valores do tipo "yes" são convertidos para 1.

Tendo todos os dados normalizados procedeu-se então à determinação dos atributos mais relevantes. Sabendo que o atributo *y* corresponde ao *output* e os restantes atributos ao *input*, recorreu-se à seguinte instrução em *R*:

```
selecao <- regsubsets(y ~ age+job+marital+education+default+
balance+housing+loan+contact+day+month+duration+campaign+
pdays+previous+poutcome, dados, method="backward")
```

Através do comando *summary(selecao)* obtiveram-se as diferentes combinações dos atributos mais significativos. Por uma questão de facilitar os testes para as diferentes combinações, criaram-se funções e seleções dos argumentos para cada uma das diferentes possibilidades:

```
funcao1 <- y ~ duration
funcao2 <- y ~ duration+poutcome
funcao3 <- y ~ housing+duration+poutcome
funcao4 <- y ~ housing+duration+pdays+poutcome
funcao5 <- y ~ housing+contact+duration+pdays+poutcome
funcao6 <- y ~ housing+loan+contact+duration+pdays+poutcome
funcao7 <- y ~ marital+housing+loan+contact+duration+pdays+poutcome
funcao8 <- y ~ marital+balance+housing+loan+contact+
    duration+pdays+poutcome

testeargs1 <- subset(teste, select=c("duration"))
testeargs2 <- subset(teste, select=c("duration","poutcome"))
testeargs3 <- subset(teste, select=c("housing","duration","poutcome"))
testeargs4 <- subset(teste, select=c("housing","duration",
    "pdays","poutcome"))
testeargs5 <- subset(teste, select=c("housing","contact",
    "duration","pdays","poutcome"))
testeargs6 <- subset(teste, select=c("housing","loan","contact",
    "duration","pdays","poutcome"))
testeargs7 <- subset(teste, select=c("marital","housing","loan",
    "contact","duration","pdays","poutcome"))
testeargs8 <- subset(teste, select=c("marital","balance","housing",
    "loan","contact","duration","pdays","poutcome"))
```

Depois de se realizarem testes para cada uma das diferentes combinações enunciadas, verificou-se que a última vertente (envolve todos os atributos) era a que produzia melhores resultados.

O ficheiro "bank-full.csv" contém 45211 registos, pelo que foram selecionados os primeiros 4521 registos para treinar a rede (cerca de 10% dos registos). Todos os outros registos foram usados para efeitos de teste à rede.

Finalmente, procedeu-se à construção da rede e respetiva verificação dos resultados. A sequência de instruções executadas foi a seguinte:

```
#Rede com duas camadas, uma com tres e outra com dois neuronios,
#com um erro minimo de 0.01
rede <- neuralnet(funcao8, treino, hidden=c(3,2), lifesign="full",
    linear.output=FALSE, threshold=0.01)

#Representacao visual da rede
plot(rede, rep="best")

#Obtencao dos resultados
rede.resultados <- compute(rede, testeargs8)

resultados <- data.frame(atual = teste$y,
    previsao = rede.resultados$net.result)

resultados$previsao <- round(resultados$previsao, digits=0)
```

```
#Calculo do erro
rmse(c(teste$y), c(resultados$previsao))
```

Verificou-se que um aumento do número de camadas e de neurónios produzia o mesmo resultado. Além disso, verificou-se que um aumento significativo dessas duas vertentes fazia com que não fosse possível atingir o erro mínimo esperado.

O erro obtido desta rede foi cerca de 0.3517, pelo que a rede é capaz de responder com sucesso a cerca de 65% dos casos.

3.2 Candidato a uma nova campanha

Como exercício adicional, foi elaborada uma nova rede no âmbito de uma nova campanha direcionada para clientes que não possuíssem qualquer tipo de empréstimos, que tivessem saldo positivo na sua conta e que não estivessem em situação de *default*.

Assim sendo, foi criada uma nova *data frame* que contém apenas os dados relevantes para este problema, nomeadamente: *default*, *loan*, *housing* e *balance*. Estes dados já se encontravam normalizados, como descrito na secção anterior.

```
dados2 <- data.frame(default=dados$default, loan=dados$loan,
                     housing=dados$housing, balance=dados$balance,
                     y=numeric(nrow(dados)))
```

Na coluna *y*, relativa ao output esperado pela rede, cada uma das linhas passou a ter um booleano *true* caso o cliente em questão tivesse os campos *default*, *loan* e *housing* com o valor 0 (i.e. não tivesse empréstimos e não estivesse em situação de *default*) e o saldo (*balance*) positivo.

```
dados2$y <- dados$default == 0 & dados$loan == 0 &
            dados$housing == 0 & dados$balance > 0
dados2$y <- as.integer(as.logical(dados2$y))
```

Através de *regsubsets*, foram determinados os campos mais significativos e criada uma função que utilizasse esses mesmos campos.

```
selecao2 <- regsubsets(y ~ default+loan+housing+balance,
                       dados2, method="backward")
funcao2 <- y ~ default+loan+housing+balance
```

Depois, utilizaram-se 25000 registos para o treino e 20000 para os testes, tendo apenas sido considerados os campos de dados relevantes para estes novo problema (*testeargs2*).

```
treino2 <- dados2[1:25000, ]
teste2 <- dados2[25001:45211, ]
testeargs2 <- subset(teste2, select=c("default","loan","housing","balance"))
```

Finalmente, foi construída uma rede com a mesma estrutura da anterior (duas camadas intermédias com 3 e 2 neurónios). Depois foram verificados os resultados e obtido o erro.

```
rede2 <- neuralnet(funcao2, treino2, hidden=c(3,2), lifesign="full",
                   linear.output=FALSE, threshold=0.01)
plot(rede2, rep="best")
rede.resultados2 <- compute(rede2, testeargs2)
resultados2 <- data.frame(atual = teste2$y,
                          previsao = rede.resultados2$net.result)
resultados2$previsao <- round(resultados2$previsao, digits=0)
rmse(c(teste2$y), c(resultados2$previsao))
```

Obteve-se um erro de 0%, o que faz sentido pois a coluna *y* proveio de uma análise simples aos quatro campos apontados anteriormente: apenas foram selecionados so clientes que tivessem os campos *default*, *loan* e *housing* com o valor 0 e o saldo (*balance*) positivo. Desta forma, após o treino desta nova rede, foi fácil concluir a ligação entre os valores destes campos e o resultado da campanha, tendo-se obtido uma rede que responde com sucesso a 100% dos casos.

4 Conclusões e Sugestões

Dado por concluído este terceiro e último trabalho prático, o grupo pensa ter alcançado o que era esperado, tanto nesta fase como nas restantes que a precederam.

Enquanto nos dois últimos exercicios se debruçaram sobre métodos de representação de conhecimento, neste abordou-se o tratamento e análise do conhecimento descrito em dados facultados, originando uma solução baseada em *RNAs*. Desta forma, conseguiu-se consolidar os conhecimentos lecionados referentes ao desenvolvimento de mecanismos de raciocínio e obter uma uma maior competência na utilização da linguagem *R*.