Mestrado em Inteligência Artificial e Sistemas Inteligentes

Extracção de Conhecimento de Dados I

Trabalho Regressão

10 de Janeiro de 2005

António Jesus Monteiro de Castro

Aluno 040594004

Faculdade de Engenharia da Universidade do Porto

ÍNDICE

Objectivo

O objectivo principal deste trabalho é obter uma previsão da mediana dos valores das casas numa determinada área com base nos valores de outros indicadores. Para o efeito foram fornecidos dois ficheiros: um com os dados sobre uma área residencial da Califórnia e que são parte das respostas ao Census de 1990 realizado nos EUA, o outro, contém as observações para as quais se pretende obter a respectiva previsão. Para atingir este objectivo é sugerida a seguinte metodologia:

1) Análise exploratória dos dados.
2) Processo de modelação adequado para obtenção das previsões.
3) Previsão dos valores para o ficheiro de teste enviado.

Finalmente, a ferramenta indicada para fazer este trabalho é o R.

Análise exploratória dos dados

Dados

Em primeiro lugar convém conhecer um pouco mais do *DataSet* que foi dado. Assim, a tabela dos atributos e respectivos significados é a seguinte:

Nome	*Descrição*	*Tipo Dados*
Longitude	Longitude da área.	Numérico
Latitude	Latitude da área.	Numérico
Med_idade_casas	Mediana da idade das casas na área.	Numérico
Tot_divisoes	Número total de divisões da área.	Numérico
Tot_quartos	Número total de quartos da área.	Numérico
Populacao_viver	População a viver na área.	Numérico
Propriedades	Número de propriedades na área.	Numérico
Salário_med	Salário médio na área.	Numérico
Med_valor_casas	Mediana do valor das casas na área.	Numérico

Finalmente o ficheiro com os dados de treino contém 15000 registos e o ficheiro com dados de teste (para o qual tenho de obter previsões) contém 5639 observações e menos um atributo, ou seja, aquele que queremos prever e que é a Mediana do valor das casas na área.

Preparação e análise dos dados

Antes de passarmos ao processo de modelação, convém conhecer um pouco mais dos dados que nos foram disponibilizados e, ao mesmo tempo, fazer a sua preparação tendo em vista a utilização da ferramenta proposta.

Assim, começamos por carregar a informação num *dataset* a que chamei census_california, usando o seguinte comando:

```
> census_california <- read.table('census_california_treino.txt', sep = ' ', header =
FALSE, dec = '.', col.names =
c('longitude','latitude','med_idade_casas','tot_divisoes',
'tot_quartos','populacao_viver','propriedades','salario_med',
'med_valor_casas'))
```

Agora interessa verificar se existem ou não *missing values*. Podemos fazê-lo através do seguinte comando:

```
> census_california[!complete.cases(census_california),]
[1] longitude       latitude        med_idade_casas tot_divisoes    tot_quartos
populacao_viver propriedades    salario_med     med_valor_casas
<0 rows> (or 0-length row.names)
```

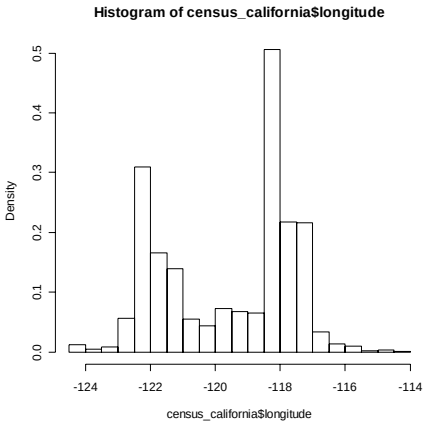
Como se verifica pela resultado este *dataframe* não contém *missing values*. Caso existissem, seria necessários substitui-los com, por exemplo, o valor mais frequente, o valor mais frequente nos casos de treino mais semelhantes ou, então, com outras técnicas mais sofisticadas, tais como, usar o atributo mais "correlacionado" com o atributo em falta (CART). Também pelo facto de não haver *missing values* optei por não estudar a correlação entre os atributos. Se existissem, então a análise da correlação entre os atributos era importante para sabermos a melhor forma de substituir esse valores em falta.

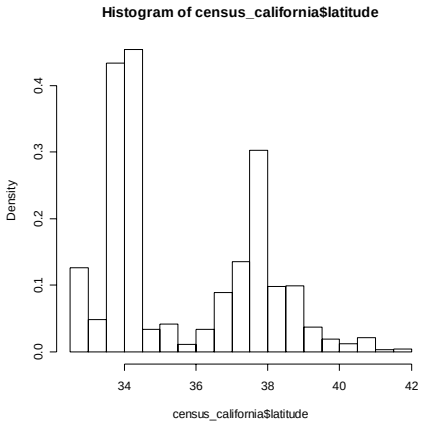
Também interessa conhecer algumas medidas estatísticas básicas sobre este *dataframe*, podemos fazê-lo através do comando *summary* e, assim, obter para cada atributo os seguintes dados (pela ordem que aparecem): Valor mínimo, 1Q, Mediana, Média, 3Q, Valor máximo. Aplicando o comando o resultado é o seguinte:

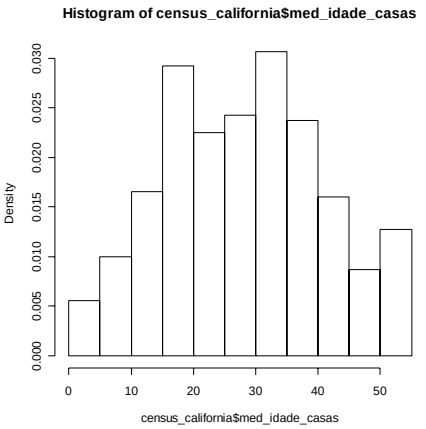
```
> print(summary(census_california))
   longitude         latitude     med_idade_casas  tot_divisoes
 Min.   :-124.3   Min.   :32.55   Min.   : 1.00   Min.   :    2
 1st Qu.:-121.8   1st Qu.:33.93   1st Qu.:18.00   1st Qu.: 1450
 Median :-118.5   Median :34.26   Median :29.00   Median : 2131
 Mean   :-119.6   Mean   :35.63   Mean   :28.59   Mean   : 2649
 3rd Qu.:-118.0   3rd Qu.:37.72   3rd Qu.:37.00   3rd Qu.: 3155
 Max.   :-114.3   Max.   :41.95   Max.   :52.00   Max.   :39320

  tot_quartos     populacao_viver    propriedades     salario_med
 Min.   :   1.0   Min.   :    3.0   Min.   :   1.0   Min.   : 0.4999
 1st Qu.: 297.0   1st Qu.:  785.8   1st Qu.: 281.0   1st Qu.: 2.5592
 Median : 438.0   Median : 1171.0   Median : 412.0   Median : 3.5291
 Mean   : 540.8   Mean   : 1431.7   Mean   : 502.2   Mean   : 3.8635
 3rd Qu.: 649.0   3rd Qu.: 1732.3   3rd Qu.: 607.0   3rd Qu.: 4.7464
 Max.   :6210.0   Max.   :35682.0   Max.   :5358.0   Max.   :15.0001

 med_valor_casas
 Min.   : 14999
 1st Qu.:119375
 Median :179500
 Mean   :206804
 3rd Qu.:264700
 Max.   :500001
```

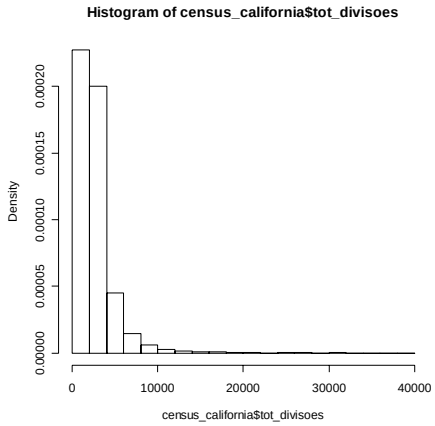
Para completar estas informações é interessante conhecermos um pouco da distribuição dos dados para cada atributo. Para isso vou socorrer-me de histogramas onde, em vez de frequências, aparecerá probabilidades para cada intervalo de valores.
Todos este gráficos foram obtidos com um comando semelhante ao que indico de seguida (todos os utilizados poderão ser encontrados no anexo deste documento):

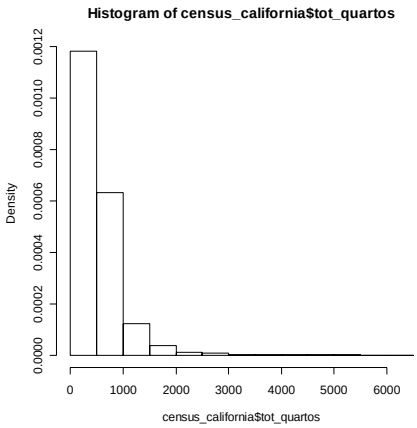
```
hist(census_california$longitude, prob=T)
```

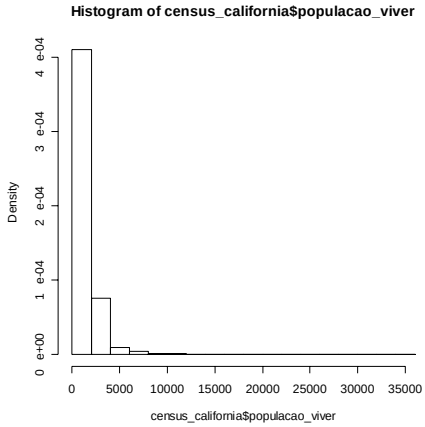

Histogram of census_california$longitude
Density
0.0
0.1
0.2
0.3
0.4
0.5
-124
-122
-120
-118
-116
-114
census_california$longitude

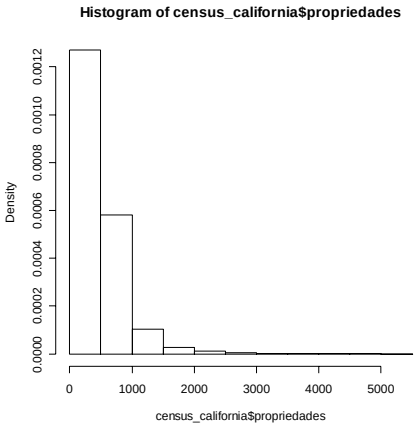

Histogram of census_california$latitude
Density
0.0
0.1
0.2
0.3
0.4
34
36
38
40
42
census_california$latitude

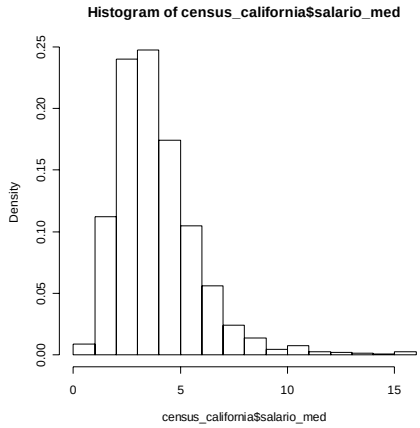

Histogram of census_california$med_idade_casas
Density
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
10
20
30
40
50
census_california$med_idade_casas

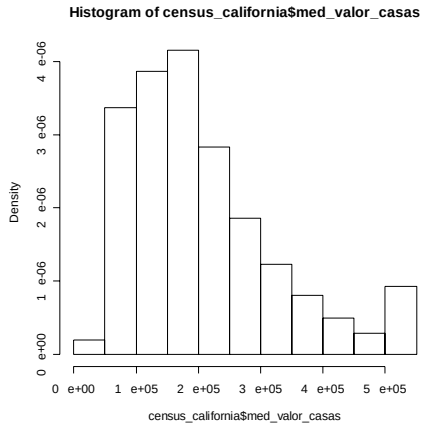

Histogram of census_california$tot_divisoes
Density
0.00000
0.00005
0.00010
0.00015
0.00020
0
10000
20000
30000
40000
census_california$tot_divisoes


Histogram of census_california$tot_quartos
Density
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0
1000
2000
3000
4000
5000
6000
census_california$tot_quartos


Histogram of census_california$populacao_viver
Density
0 e+00
1 e-04
2 e-04
3 e-04
4 e-04
0
5000
10000
15000
20000
25000
30000
35000
census_california$populacao_viver


Histogram of census_california$propriedades
Density
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0
1000
2000
3000
4000
5000
census_california$propriedades


Histogram of census_california$salario_med
Density
0.00
0.05
0.10
0.15
0.20
0.25
0
5
10
15
census_california$salario_med


Histogram of census_california$med_valor_casas
Density
0 e+00
1 e-06
2 e-06
3 e-06
4 e-06
0 e+00
1 e+05
2 e+05
3 e+05
4 e+05
5 e+05
census_california$med_valor_casas

Processo de Modelação

Dos modelos apresentados nas aulas, escolhi os seguintes:

- Regressão linear.
- Projection Pursuit Regression.
- Mars – Multivariate Adaptive Regression Splines.
- Árvores de Regressão.
- Redes Neuronais.

Para gerar os modelos começo por criar uma amostra de treino e outra de teste a partir do ficheiro de treino dado, utilizando a técnica habitual de amostragem 70/30. De realçar que dado o número considerável de registos de treino (15000) optei por aceitar os erros calculados, sem ter de repetir várias vezes e achar a respectiva média dos mesmos. Se o número de registos fosse bem menor, teria de obter a média dos erros das várias "execuções" que fizesse ou, então, optar por validação cruzada, de forma a obter valores de erros dos modelos mais correctos.

Para criar a amostra de treino e de teste, usei os seguintes comandos:

```
amostra <- sample(nrow(census_california), as.integer(0.7*nrow(census_california)))
amostra.treino <- census_california[amostra,]
amostra.teste <- census_california[-amostra,]
```

A partir deste momento, os *dataframes* amostra.treino e amostra.teste, serão usados para gerar os vários modelos. O processo para gerar os vários modelos será idêntico para todos e resume-se ao seguinte:

(1) Calcular o modelo usando a amostra de treino.
(2) Calcular as previsões usando o modelo calculado e aplicando-o ás amostras de teste.
(3) Calcular as medidas de erro de previsão para que, depois, seja possível fazer a avaliação dos modelos. As medidas de previsão calculadas são: MAE – *Mean Absolute Erros*, MSE – *Mean Squared Error*, NMSE – *Normalized Mean Squared Error*, RMSE – *Root Mean Squared Error*.

De realçar que, em princípio, não é necessário calcular todas estas medidas de erro. Poderia calcular somente o MAE ou o MSE ou, então, somente o NMSE que será aquela que eu vou optar por usar no final. Dito isto, porquê calcular todas as outras medidas? Pela simples razão de que um dos objectivos também é aprender R e, assim, contribuo para esse objectivo. Finalmente, opto pelo NMSE porque este indicador é independente das unidades do atributo e, assim é mais fácil para o utilizador interpretá-lo. Este indicador calcula um rácio entre o desempenho do modelo e uma predição base (normalmente a média do atributo objectivo). O valor do NMSE varia entre 0 e 1 e, normalmente, quanto mais baixo, melhor. Se for superior a 1 significa que o modelo está a ter um desempenho pior do que se usarmos a média do atributo como previsão. Finalmente, através do calculo do coeficiente de correlação do modelo, também obtenho algumas indicações sobre se o modelo gerado é ou não bom. O coeficiente de correlação poderá ter um valor entre -1 e 1, sendo mais perto de 1 quando está mais correlacionado e, como tal, melhor.

Modelo Regressão Linear

Para obter o modelo e calcular as previsões, utilizei os seguinte comandos:

```
mod.linear.census_california  <- lm(med_valor_casas ~ ., data=amostra.treino)
mod.linear.previsoes   <- predict(mod.linear.census_california, amostra.teste[,-9])
```

Para calcular os erros e o coeficiente de correlação:

```
mod.linear.mae <- mean(abs(mod.linear.previsoes-amostra.teste[,'med_valor_casas']))

mod.linear.mse <- mean((mod.linear.previsoes-amostra.teste[,'med_valor_casas'])^2)

mod.linear.nmse <- mean((mod.linear.previsoes-amostra.teste[,'med_valor_casas'])^2) /
mean((mean(amostra.teste[,'med_valor_casas'])-amostra.teste[,'med_valor_casas'])^2)

mod.linear.rmse<- sqrt(mod.linear.mse)

mod.linear.cor <- cor(amostra.teste[,ncol(amostra.teste)], mod.linear.previsoes)
```

Usando o comando *summary*, podemos obter informações sobre o modelo, informação essa que poderão encontrar no anexo 1. Neste momento é mais importante analisar os resultados dos erros, sobretudo o NMSE pelas razões já referidas, bem como o coeficiente de correlação.
Os dados obtidos são os seguintes:

```
Erro Previsao MAE:  51550.85

Erro Previsao MSE:  5013945628

Erro Previsao RMSE:  70809.22

Erro Previsao NMSE:  0.373728

Coef. Correlação:  0.7915845
```

Como podemos ver um valor aproximado de 0,367 para o erro NMSE é um valor razoável o que, à partida, pode significar que estamos perante um bom modelo. Além disso o coeficiente de correlação que é de aproximadamente 79% também é um valor razoável.

Modelo Projection Pursuit

Para obter o modelo e calcular as previsões:

```
mod.purs.census_california  <- ppr(med_valor_casas ~ ., data=amostra.treino,nterms=5)
mod.purs.previsoes <- predict(mod.purs.census_california, amostra.teste[,-9])
```

Convém aqui realçar que usei parâmetros normais que encontrei na documentação, sem ter feito nenhuma experiência com os mesmos.

Para calcular os erros e o coeficiente de correlação, fazemos:

```
mod.purs.mae <- mean(abs(mod.purs.previsoes - amostra.teste[,'med_valor_casas']))

mod.purs.mse <- mean((mod.purs.previsoes - amostra.teste[,'med_valor_casas'])^2)

mod.purs.nmse <- mean((mod.purs.previsoes - amostra.teste[,'med_valor_casas'])^2) /
mean((mean(amostra.teste[,'med_valor_casas']) - amostra.teste[,'med_valor_casas'])^2)

mod.purs.rmse  <- sqrt(mod.purs.mse)

mod.purs.cor <- cor(amostra.teste[,ncol(amostra.teste)], mod.purs.previsoes)
```

Os erros e o coeficiente de correlação obtidos são:

```
Erro Previsao MAE:  41549.44

Erro Previsao MSE:  3497102494

Erro Previsao RMSE:  59136.3

Erro Previsao NMSE:  0.2606660

Coef. Correlação:  0.8598872
```

Por comparação com o anterior NMSE, bem como pelo coeficiente de correlação obtido, podemos ver que este modelo é melhor.

Modelo MARS

Obtenção do modelo e calculo das previsões:

```
mod.mars.census_california  <- mars(amostra.treino[,-9],amostra.treino[,9])
mod.mars.previsoes  <- predict(mod.mars.census_california, amostra.teste[,-9])
```

Calculo dos erros e do coeficiente de correlação:

```
mod.mars.mae <- mean(abs(mod.mars.previsoes - amostra.teste[,'med_valor_casas']))

mod.mars.mse <- mean((mod.mars.previsoes - amostra.teste[,'med_valor_casas'])^2)

mod.mars.nmse <- mean((mod.mars.previsoes - amostra.teste[,'med_valor_casas'])^2) /
mean((mean(amostra.teste[,'med_valor_casas']) - amostra.teste[,'med_valor_casas'])^2)

mod.mars.rmse <- sqrt(mod.mars.mse)

mod.mars.cor <- cor(amostra.teste[,ncol(amostra.teste)], mod.mars.previsoes)
```

Os erros e o coeficiente de correlação obtidos são:

```
Erro Previsao MAE:  48044.67

Erro Previsao MSE:  4367618521

Erro Previsao RMSE:  66087.96

Erro Previsao NMSE:  0.3255522

Coef. Correlação:  0.8212942
```

Modelo Árvore de Regressão

Obtenção do modelo e calculo das previsões:

```
mod.arv.census_california  <- rpart(med_valor_casas ~ ., data=amostra.treino)
mod.arv.previsoes <- predict(mod.arv.census_california, amostra.teste[,-9])
```

Calculo dos erros e do coeficiente de correlação:

```
mod.arv.mae <- mean(abs(mod.arv.previsoes - amostra.teste[,'med_valor_casas']))

mod.arv.mse <- mean((mod.arv.previsoes - amostra.teste[,'med_valor_casas'])^2)

mod.arv.nmse <- mean((mod.arv.previsoes - amostra.teste[,'med_valor_casas'])^2) /
mean((mean(amostra.teste[,'med_valor_casas']) - amostra.teste[,'med_valor_casas'])^2)

mod.arv.rmse <- sqrt(mod.arv.mse)

mod.arv.cor <- cor(amostra.teste[,ncol(amostra.teste)], mod.arv.previsoes)
```

Os erros e o coeficiente de correlação obtidos são:

```
Erro Previsao MAE:  56484.19

Erro Previsao MSE:  5879947343

Erro Previsao RMSE:  76680.81

Erro Previsao NMSE:  0.4382777

Coef. Correlação:  0.7496077
```

É normal quando se está a lidar com árvores de decisão de se fazer o *pruning* da árvore com o objectivo de a simplificar. Podemos optar pelo *Pré-pruning* – podamos a árvore antes de ela crescer completamente, ou pelo *Post-pruning* – deixámos a árvore crescer completamente e, depois, podamo-la. No entanto e tendo em conta a dimensão da árvore que foi gerada, entendi que não haveria melhorias nos resultados que o justificassem. Concordo que esta minha decisão é discutível, sobretudo quando se trata de um trabalho académico e cujo objectivo é aprender.

De qualquer forma, o processo para o fazer seria o seguinte:

1) Através do comando printcp(mod.arv.census_california) obteríamos uma tabela de CP (*Cost Complexity* Pruning), onde, para cada valor de CP aparece o erro relativo na amostra de treino, o erro estimado por validação cruzada e o erro padrão da estimativa.
2) Como a cada valor de CP corresponde uma sub-árvore, escolheríamos um deles a partir do qual quiséssemos simplificar a árvore. Aplicando o comando arv.podada <- prune(mod.arv.census_california, cp = valor escolhido)
3) Finalmente, fazíamos a previsão usando a arv.podada e verificaríamos se os erros de previsão calculados seriam ou não melhores que os obtidos com a árvore sem ser podada.

Modelo Redes Neuronais

Obtenção do modelo e calculo das previsões:

```
mod.rede.census_california  <- nnet(med_valor_casas ~ ., data=amostra.treino,
size=20,decay=1,maxit=1000,linout=T)

mod.rede.previsoes     <- predict(mod.rede.census_california, amostra.teste[,-9])
```

Calculo dos erros e do coeficiente de correlação:

```
mod.rede.mae <- mean(abs(mod.rede.previsoes - amostra.teste[,'med_valor_casas']))

mod.rede.mse <- mean((mod.rede.previsoes - amostra.teste[,'med_valor_casas'])^2)

mod.rede.nmse <- mean((mod.rede.previsoes - amostra.teste[,'med_valor_casas'])^2) /
mean((mean(amostra.teste[,'med_valor_casas']) - amostra.teste[,'med_valor_casas'])^2)

mod.rede.rmse <- sqrt(mod.rede.mse)

mod.rede.cor <- cor(amostra.teste[,ncol(amostra.teste)], mod.rede.previsoes)
```

Erros obtidos e coeficiente de correlação:

```
Erro Previsao MAE:  92411.67

Erro Previsao MSE:  13523957233

Erro Previsao RMSE:  116292.6

Erro Previsao NMSE:  1.008045

Coeficiente Corr:  -0.07226135
```

Como se pode verificar o erro NMSE obtido indica que algo não está bem neste modelo. No entanto uma das maiores dificuldades de uma rede neuronal é utilizarmos a melhor configuração para o problema em causa. Usando uma função que encontrei na documentação [Torgo04b] é possível tentar encontrar os melhores valores para os parâmetros *size* e *decay*. Nesta função que se chama *config.nn.nmse* mantive a mesma metodologia experimental (30% dos dados de fora) mas alterei a medida de erro a ser usada de MSE para NMSE, por uma questão de coerência. Vamos então invocar a função experimentando parâmetros diferentes, neste caso, *size=8 ou 12 ou 16* e *decay = 1 ou 2 ou 3*.

```
> res.nmse <- config.nn.nmse(expand.grid(size = c(8, 12, 16), decay = c(1, 2, 3)),
census_california)
```

O resultado obtido foi o seguinte:

```
> print(res.nmse)
$Resultados
  size decay      NMSE
1    8     1 0.7670639
2   12     1 0.8439867
3   16     1 0.7579374
4    8     2 1.0011563
5   12     2 0.8675026
6   16     2 0.3154774
7    8     3 0.2786844
8   12     3 1.0019241
9   16     3 0.2898213

$Melhor
  size decay      NMSE
7    8     3 0.2786844
```

Agora, experimentamos novamente a obtenção do modelo e o calculo das previsões, mas com os novos parâmetros. Assim:

```
mod.rede.census_california  <- nnet(med_valor_casas ~ ., data=amostra.treino,
size=8,decay=3,maxit=1000,linout=T)

mod.rede.previsoes     <- predict(mod.rede.census_california, amostra.teste[,-9])
```

Os erros e o coeficiente de correlação obtidos com os novos parâmetros foram:

```
Erro Previsao MAE:  44629.96

Erro Previsao MSE:  3894586322

Erro Previsao RMSE:  62406.62

Erro Previsao NMSE:  0.2902935

Coeficiente Corr:  0.8426532
```

Como se pode verificar existe uma diferença substancial. Logicamente que se poderia continuar tentando encontrar melhores valores de configuração, no entanto e para efeito deste trabalho, pararei por aqui porque considero que são óptimos valores sobretudo se olharmos também para o coeficiente de correlação.

Avaliação dos Modelos

Com os erros de previsão de cada modelo e respectivo coeficiente de correlação, foi criado um *dataframe* chamado err.prev.census_california (ver o comando no anexo) para o qual passo a apresentar os resultados:

```
  mod.linear.mae mod.linear.mse mod.linear.rmse mod.linear.nmse mod.linear.cor
       51550.85     5013945628        70809.22        0.373728      0.7915845

  mod.purs.mae mod.purs.mse mod.purs.rmse mod.purs.nmse mod.purs.cor
     41549.44   3497102494       59136.3     0.2606660    0.8598872

  mod.mars.mae mod.mars.mse mod.mars.rmse mod.mars.nmse mod.mars.cor
     48044.67   4367618521      66087.96     0.3255522    0.8212942

  mod.arv.mae mod.arv.mse mod.arv.rmse mod.arv.nmse mod.arv.cor
    56484.19  5879947343     76680.81    0.4382777   0.7496077

  mod.rede.mae mod.rede.mse mod.rede.rmse mod.rede.nmse mod.rede.cor
     44629.96   3894586322      62406.62     0.2902935    0.8426532
```

Como disse anteriormente, não era necessário calcular todos estes erros para poder avaliar o modelo. Assim, optei por definir o modelo a utilizar, através da análise do NMSE e, neste caso, o modelo adequado é o *Projection Pursuit* . Da tabela acima também se pode verificar que em todos os outros erros, este modelo também é o melhor, assim como o seu coeficiente de correlação também é o melhor de todos eles.

Previsão dos valores

Para aplicar o modelo escolhido ao conjunto de testes fornecido será necessário utilizar os seguintes comandos:

Para carregar o ficheiro de teste

```
> teste.census_california <-read.table('census_california_teste.txt',
        sep = ' ', header = FALSE, dec = '.',
        col.names = c('longitude','latitude','med_idade_casas',
        'tot_divisoes','tot_quartos','populacao_viver',
        'propriedades','salario_med'))
```

Para prever os dados com o modelo escolhido

```
> previsao.census_california <- predict(mod.purs.census_california,
teste.census_california)
```

Para escrever as previsões num ficheiro

```
> write.table(previsao.census_california, 'ecd_regressao_previsao_antonio_castro.txt')
```

Nota Final

Para além do ficheiro com este relatório em PDF, foram entregues mais dois ficheiros:

Ecd_regressao_antonio_castro.r com o código R usado neste trabalho
Ecd_regressao_previsao_antonio_castro.txt com as previsões pedidas.

Referências

[Torgo04] Luís Torgo. *Acetatos da Cadeira de Extracção de Conhecimento de Dados 1*. Faculdade de Economia da Universidade do Porto, 2004.

[Torgo04b] Luís Torgo. *Redes Neuronais – Guia Prático*. Faculdade de Economia da Universidade do Porto, 2004.

[Torgo03] Luís Torgo. *Data Mining with R – Learning By Case Studies*. Faculdade de Economia da Universidade do Porto, Maio de 2003.

[Ven04] W. N. Venables, D. M. Smith. *An Introduction to R*. Network Theory Limited, 2004.

Anexo 1

Summary do Modelo de Regressão Linear

```
> print(summary(mod.linear.census_california))

Call:
lm(formula = med_valor_casas ~ ., data = amostra.treino)

Residuals:
    Min      1Q  Median      3Q     Max
-420341  -44584  -11717   30331  691966

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -3.473e+06  8.864e+04 -39.184  < 2e-16 ***
longitude       -4.165e+04  1.011e+03 -41.202  < 2e-16 ***
latitude        -4.215e+04  9.559e+02 -44.098  < 2e-16 ***
med_idade_casas  1.220e+03  6.120e+01  19.936  < 2e-16 ***
tot_divisoes    -8.471e+00  1.125e+00  -7.530 5.47e-14 ***
tot_quartos      1.069e+02  9.854e+00  10.851  < 2e-16 ***
populacao_viver -3.407e+01  1.460e+00 -23.331  < 2e-16 ***
propriedades     4.665e+01  1.059e+01   4.404 1.07e-05 ***
salario_med      4.071e+04  4.769e+02  85.362  < 2e-16 ***
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Residual standard error: 70370 on 10491 degrees of freedom
Multiple R-Squared: 0.6312,     Adjusted R-squared: 0.6309
F-statistic:  2244 on 8 and 10491 DF,  p-value: < 2.2e-16
```

Summary do Modelo de Regressão Linear

```
> print(summary(mod.gener.census_california))

Call:
glm(formula = med_valor_casas ~ ., data = amostra.treino)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-420341   -44584   -11717    30331   691966

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -3.473e+06  8.864e+04 -39.184  < 2e-16 ***
longitude       -4.165e+04  1.011e+03 -41.202  < 2e-16 ***
latitude        -4.215e+04  9.559e+02 -44.098  < 2e-16 ***
med_idade_casas  1.220e+03  6.120e+01  19.936  < 2e-16 ***
tot_divisoes    -8.471e+00  1.125e+00  -7.530 5.47e-14 ***
tot_quartos      1.069e+02  9.854e+00  10.851  < 2e-16 ***
populacao_viver -3.407e+01  1.460e+00 -23.331  < 2e-16 ***
propriedades     4.665e+01  1.059e+01   4.404 1.07e-05 ***
salario_med      4.071e+04  4.769e+02  85.362  < 2e-16 ***
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

(Dispersion parameter for gaussian family taken to be 4952416650)

    Null deviance: 1.4087e+14  on 10499  degrees of freedom
Residual deviance: 5.1956e+13  on 10491  degrees of freedom
AIC: 264202

Number of Fisher Scoring iterations: 2
```

Summary do Modelo *Projection Pursuit*

```
> print(summary(mod.purs.census_california))
Call:
ppr(formula = med_valor_casas ~ ., data = amostra.treino, nterms = 5)

Goodness of fit:
     5 terms
3.537945e+13

Projection direction vectors:
                term 1        term 2        term 3        term 4        term 5
longitude        7.194576e-02 -7.029282e-01 -1.341451e-01  7.688050e-03 -6.022025e-01
latitude        -5.645022e-02 -7.111018e-01  2.273410e-01 -7.192052e-01 -7.963433e-01
med_idade_casas  4.843719e-02  1.122391e-04 -1.219606e-01 -4.189947e-01  1.979009e-02
tot_divisoes    -1.505109e-04  2.395584e-05  2.741937e-03 -6.922738e-03 -3.330070e-03
tot_quartos      1.996645e-03 -1.206306e-04 -1.338135e-02  5.249288e-02  1.458407e-02
populacao_viver -1.425086e-03  2.376953e-05  1.251781e-03 -3.916665e-02 -2.866150e-03
propriedades     3.149443e-03 -2.990645e-05 -5.290382e-03  9.585309e-02  7.423517e-03
salario_med      9.946231e-01 -1.503728e-02 -9.566769e-01 -5.418514e-01  5.010712e-02

Coefficients of ridge terms:
  term 1   term 2   term 3   term 4   term 5
90759.47 40178.38 25005.84 20128.44 23298.16
```

Summary do Modelo MARS

```
> print(summary(mod.mars.census_california))
             Length Class  Mode
call              3 -none- call
all.terms        16 -none- numeric
selected.terms   16 -none- numeric
penalty           1 -none- numeric
degree            1 -none- numeric
nk                1 -none- numeric
thresh            1 -none- numeric
gcv               1 -none- numeric
factor          168 -none- numeric
cuts            168 -none- numeric
residuals     10500 -none- numeric
fitted.values 10500 -none- numeric
lenb              1 -none- numeric
coefficients     16 -none- numeric
x            168000 -none- numeric
```

Summary do Modelo Árvore de Regressão

```
> print(summary(mod.arv.census_california))
Call:
rpart(formula = med_valor_casas ~ ., data = amostra.treino)
  n= 10500

          CP nsplit rel error    xerror        xstd
1 0.31750842      0 1.0000000 1.0000968 0.014781736
2 0.07960125      1 0.6824916 0.6913029 0.011678437
3 0.05351934      2 0.6028903 0.6112365 0.010844599
4 0.02064788      3 0.5493710 0.5582999 0.010696568
5 0.01515176      4 0.5287231 0.5410969 0.010400462
6 0.01103702      6 0.4984196 0.5137677 0.009972822
7 0.01088837      8 0.4763455 0.4843412 0.009656111
8 0.01056542      9 0.4654572 0.4758785 0.009467768
9 0.01000000     13 0.4231955 0.4542481 0.009060719

Node number 1: 10500 observations,    complexity param=0.3175084
  mean=206668.8, MSE=1.341651e+10
  left son=2 (8336 obs) right son=3 (2164 obs)
  Primary splits:
      salario_med     < 5.0707   to the left,  improve=0.31750840, (0 missing)
      latitude        < 37.925   to the right, improve=0.07001514, (0 missing)
      longitude       < -121.865 to the right, improve=0.03022440, (0 missing)
      med_idade_casas < 51.5     to the left,  improve=0.02531783, (0 missing)
      tot_divisoes    < 2382.5   to the left,  improve=0.02506913, (0 missing)
  Surrogate splits:
      tot_divisoes    < 13799    to the left,  agree=0.795, adj=0.006, (0 split)
      tot_quartos     < 5161.5   to the left,  agree=0.794, adj=0.001, (0 split)
      populacao_viver < 14144    to the left,  agree=0.794, adj=0.001, (0 split)
      propriedades    < 4952.5   to the left,  agree=0.794, adj=0.001, (0 split)

Node number 2: 8336 observations,    complexity param=0.07960125
  mean=173414.6, MSE=8.448165e+09
  left son=4 (4177 obs) right son=5 (4159 obs)
  Primary splits:
      salario_med     < 3.13215  to the left,  improve=0.15923140, (0 missing)
      latitude        < 37.915   to the right, improve=0.06389560, (0 missing)
      med_idade_casas < 51.5     to the left,  improve=0.03920648, (0 missing)
      longitude       < -121.865 to the right, improve=0.03056744, (0 missing)
      propriedades    < 362.5    to the left,  improve=0.01601905, (0 missing)
  Surrogate splits:
      tot_divisoes    < 1932.5   to the left,  agree=0.575, adj=0.149, (0 split)
      longitude       < -118.155 to the left,  agree=0.544, adj=0.087, (0 split)
      latitude        < 38.515   to the right, agree=0.537, adj=0.073, (0 split)
      med_idade_casas < 36.5     to the right, agree=0.537, adj=0.072, (0 split)
      propriedades    < 705.5    to the left,  agree=0.522, adj=0.042, (0 split)

Node number 3: 2164 observations,    complexity param=0.05351934
  mean=334768.4, MSE=1.188586e+10
  left son=6 (1500 obs) right son=7 (664 obs)
  Primary splits:
      salario_med     < 6.81755  to the left,  improve=0.29312430, (0 missing)
      med_idade_casas < 27.5     to the left,  improve=0.09596095, (0 missing)
      latitude        < 37.955   to the right, improve=0.06256267, (0 missing)
      longitude       < -118.035 to the right, improve=0.04091528, (0 missing)
      populacao_viver < 1496.5   to the right, improve=0.01812468, (0 missing)
  Surrogate splits:
      populacao_viver < 64       to the right, agree=0.696, adj=0.011, (0 split)
      tot_quartos     < 49.5     to the right, agree=0.696, adj=0.009, (0 split)
      propriedades    < 28.5     to the right, agree=0.695, adj=0.008, (0 split)
      tot_divisoes    < 143.5    to the right, agree=0.695, adj=0.006, (0 split)

Node number 4: 4177 observations,    complexity param=0.01515176
  mean=136816.5, MSE=5.707319e+09
  left son=8 (2142 obs) right son=9 (2035 obs)
  Primary splits:
      latitude        < 34.455   to the right, improve=0.08283214, (0 missing)
      salario_med     < 2.37655  to the left,  improve=0.06453312, (0 missing)
      longitude       < -118.795 to the left,  improve=0.05172830, (0 missing)
      propriedades    < 524.5    to the left,  improve=0.03169324, (0 missing)
      med_idade_casas < 51.5     to the left,  improve=0.02882239, (0 missing)
  Surrogate splits:
```

```
      longitude       < -118.69  to the left,  agree=0.967, adj=0.933, (0 split)
      populacao_viver < 1497.5   to the left,  agree=0.599, adj=0.177, (0 split)
      propriedades    < 524.5    to the left,  agree=0.562, adj=0.101, (0 split)
      tot_quartos     < 484.5    to the left,  agree=0.554, adj=0.086, (0 split)
      med_idade_casas < 23.5     to the left,  agree=0.542, adj=0.060, (0 split)

Node number 5: 4159 observations,    complexity param=0.02064788
  mean=210171, MSE=8.504625e+09
  left son=10 (3302 obs) right son=11 (857 obs)
  Primary splits:
      med_idade_casas < 38.5     to the left,  improve=0.08223571, (0 missing)
      latitude        < 37.945   to the right, improve=0.06280383, (0 missing)
      longitude       < -122.355 to the right, improve=0.05452718, (0 missing)
      salario_med     < 4.06825  to the left,  improve=0.04209622, (0 missing)
      tot_quartos     < 395.5    to the left,  improve=0.01512322, (0 missing)
  Surrogate splits:
      longitude < -122.175 to the right, agree=0.797, adj=0.014, (0 split)

Node number 6: 1500 observations,    complexity param=0.01088837
  mean=295496.7, MSE=8.940267e+09
  left son=12 (1251 obs) right son=13 (249 obs)
  Primary splits:
      med_idade_casas < 36.5     to the left,  improve=0.11438000, (0 missing)
      salario_med     < 5.71225  to the left,  improve=0.06674471, (0 missing)
      latitude        < 37.955   to the right, improve=0.05958429, (0 missing)
      longitude       < -122.105 to the right, improve=0.04192564, (0 missing)
      populacao_viver < 1378.5   to the right, improve=0.01398366, (0 missing)

Node number 7: 664 observations
  mean=423484.6, MSE=7.185469e+09

Node number 8: 2142 observations,    complexity param=0.01103702
  mean=115623.8, MSE=4.639281e+09
  left son=16 (1274 obs) right son=17 (868 obs)
  Primary splits:
      longitude       < -121.72  to the right, improve=0.15519070, (0 missing)
      med_idade_casas < 51.5     to the left,  improve=0.09561219, (0 missing)
      salario_med     < 2.3699   to the left,  improve=0.08223196, (0 missing)
      latitude        < 36.845   to the left,  improve=0.05590225, (0 missing)
      propriedades    < 500.5    to the left,  improve=0.02386524, (0 missing)
  Surrogate splits:
      latitude        < 37.675   to the left,  agree=0.687, adj=0.227, (0 split)
      med_idade_casas < 43.5     to the left,  agree=0.661, adj=0.164, (0 split)
      propriedades    < 1454     to the left,  agree=0.599, adj=0.010, (0 split)
      tot_quartos     < 1549     to the left,  agree=0.598, adj=0.008, (0 split)
      salario_med     < 3.09505  to the left,  agree=0.597, adj=0.005, (0 split)

Node number 9: 2035 observations,    complexity param=0.01515176
  mean=159123.6, MSE=5.861158e+09
  left son=18 (1664 obs) right son=19 (371 obs)
  Primary splits:
      longitude    < -118.305 to the right, improve=0.19235320, (0 missing)
      salario_med  < 2.2147   to the left,  improve=0.07537072, (0 missing)
      latitude     < 34.015   to the left,  improve=0.04621424, (0 missing)
      propriedades < 730.5    to the left,  improve=0.02392257, (0 missing)
      tot_quartos  < 638.5    to the left,  improve=0.01970131, (0 missing)
  Surrogate splits:
      latitude     < 34.165   to the left,  agree=0.876, adj=0.318, (0 split)
      tot_quartos  < 2722     to the left,  agree=0.819, adj=0.008, (0 split)
      propriedades < 2542     to the left,  agree=0.819, adj=0.008, (0 split)

Node number 10: 3302 observations,    complexity param=0.01056542
  mean=196698.2, MSE=6.786446e+09
  left son=20 (618 obs) right son=21 (2684 obs)
  Primary splits:
      latitude        < 37.945   to the right, improve=0.05342179, (0 missing)
      salario_med     < 4.30325  to the left,  improve=0.04615308, (0 missing)
      longitude       < -117.795 to the right, improve=0.02662317, (0 missing)
      med_idade_casas < 19.5     to the left,  improve=0.02586798, (0 missing)
      tot_quartos     < 414.5    to the left,  improve=0.01912518, (0 missing)
  Surrogate splits:
      longitude < -122.5   to the left,  agree=0.849, adj=0.191, (0 split)

Node number 11: 857 observations
  mean=262081.6, MSE=1.173064e+10

Node number 12: 1251 observations
```

```
  mean=281230.1, MSE=7.706728e+09

Node number 13: 249 observations
  mean=367173.6, MSE=8.977514e+09

Node number 16: 1274 observations
  mean=93475.82, MSE=2.01474e+09

Node number 17: 868 observations,    complexity param=0.01103702
  mean=148131.2, MSE=6.714722e+09
  left son=34 (454 obs) right son=35 (414 obs)
  Primary splits:
      latitude        < 37.905   to the right, improve=0.26893640, (0 missing)
      med_idade_casas < 51.5     to the left,  improve=0.12017370, (0 missing)
      salario_med     < 2.37085  to the left,  improve=0.08588345, (0 missing)
      longitude       < -122.845 to the left,  improve=0.07534780, (0 missing)
      propriedades    < 505.5    to the left,  improve=0.02680909, (0 missing)
  Surrogate splits:
      med_idade_casas < 36.5     to the left,  agree=0.725, adj=0.423, (0 split)
      longitude       < -122.305 to the left,  agree=0.722, adj=0.418, (0 split)
      salario_med     < 2.37085  to the left,  agree=0.570, adj=0.099, (0 split)
      tot_divisoes    < 1252.5   to the right, agree=0.569, adj=0.097, (0 split)
      populacao_viver < 1385     to the left,  agree=0.551, adj=0.058, (0 split)

Node number 18: 1664 observations
  mean=143269.2, MSE=3.675645e+09

Node number 19: 371 observations
  mean=230233.7, MSE=9.479508e+09

Node number 20: 618 observations
  mean=157017.6, MSE=2.779336e+09

Node number 21: 2684 observations,    complexity param=0.01056542
  mean=205834.7, MSE=7.263075e+09
  left son=42 (2339 obs) right son=43 (345 obs)
  Primary splits:
      longitude       < -121.865 to the right, improve=0.06553952, (0 missing)
      salario_med     < 4.3592   to the left,  improve=0.04305380, (0 missing)
      med_idade_casas < 17.5     to the left,  improve=0.02615855, (0 missing)
      tot_quartos     < 414.5    to the left,  improve=0.02060920, (0 missing)
      propriedades    < 406.5    to the left,  improve=0.01563395, (0 missing)
  Surrogate splits:
      latitude < 37.225   to the left,  agree=0.924, adj=0.412, (0 split)

Node number 34: 454 observations
  mean=107551.3, MSE=2.01649e+09

Node number 35: 414 observations
  mean=192631.9, MSE=8.080747e+09

Node number 42: 2339 observations,    complexity param=0.01056542
  mean=197455.5, MSE=6.873865e+09
  left son=84 (500 obs) right son=85 (1839 obs)
  Primary splits:
      latitude        < 34.475   to the right, improve=0.08412876, (0 missing)
      longitude       < -117.795 to the right, improve=0.04374596, (0 missing)
      salario_med     < 4.4146   to the left,  improve=0.03933571, (0 missing)
      med_idade_casas < 19.5     to the left,  improve=0.02315112, (0 missing)
      tot_quartos     < 414.5    to the left,  improve=0.01802370, (0 missing)
  Surrogate splits:
      longitude < -119.305 to the left,  agree=0.938, adj=0.712, (0 split)

Node number 43: 345 observations
  mean=262643.8, MSE=6.198522e+09

Node number 84: 500 observations
  mean=151336.6, MSE=4.293669e+09

Node number 85: 1839 observations,    complexity param=0.01056542
  mean=209994.6, MSE=6.839867e+09
  left son=170 (1348 obs) right son=171 (491 obs)
  Primary splits:
      longitude       < -118.275 to the right, improve=0.16903230, (0 missing)
      salario_med     < 4.36915  to the left,  improve=0.03335159, (0 missing)
      latitude        < 32.785   to the left,  improve=0.02480151, (0 missing)
      tot_quartos     < 407.5    to the left,  improve=0.01909426, (0 missing)
```

```
      med_idade_casas < 17.5     to the left,  improve=0.01802346, (0 missing)
  Surrogate splits:
      latitude     < 34.145   to the left,  agree=0.858, adj=0.468, (0 split)
      propriedades < 3516     to the left,  agree=0.734, adj=0.002, (0 split)
      salario_med  < 5.06005  to the left,  agree=0.734, adj=0.002, (0 split)

Node number 170: 1348 observations
  mean=189473.3, MSE=4.54465e+09

Node number 171: 491 observations
  mean=266334.1, MSE=8.810902e+09

n= 10500

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

  1) root 10500 1.408734e+14 206668.80
    2) salario_med< 5.0707 8336 7.042391e+13 173414.60
      4) salario_med< 3.13215 4177 2.383947e+13 136816.50
        8) latitude>=34.455 2142 9.937340e+12 115623.80
         16) longitude>=-121.72 1274 2.566778e+12  93475.82 *
         17) longitude< -121.72 868 5.828379e+12 148131.20
           34) latitude>=37.905 454 9.154863e+11 107551.30 *
           35) latitude< 37.905 414 3.345429e+12 192631.90 *
        9) latitude< 34.455 2035 1.192746e+13 159123.60
         18) longitude>=-118.305 1664 6.116274e+12 143269.20 *
         19) longitude< -118.305 371 3.516897e+12 230233.70 *
      5) salario_med>=3.13215 4159 3.537074e+13 210171.00
       10) med_idade_casas< 38.5 3302 2.240884e+13 196698.20
         20) latitude>=37.945 618 1.717630e+12 157017.60 *
         21) latitude< 37.945 2684 1.949409e+13 205834.70
           42) longitude>=-121.865 2339 1.607797e+13 197455.50
             84) latitude>=34.475 500 2.146834e+12 151336.60 *
             85) latitude< 34.475 1839 1.257852e+13 209994.60
              170) longitude>=-118.275 1348 6.126188e+12 189473.30 *
              171) longitude< -118.275 491 4.326153e+12 266334.10 *
           43) longitude< -121.865 345 2.138490e+12 262643.80 *
       11) med_idade_casas>=38.5 857 1.005316e+13 262081.60 *
    3) salario_med>=5.0707 2164 2.572100e+13 334768.40
      6) salario_med< 6.81755 1500 1.341040e+13 295496.70
       12) med_idade_casas< 36.5 1251 9.641117e+12 281230.10 *
       13) med_idade_casas>=36.5 249 2.235401e+12 367173.60 *
      7) salario_med>=6.81755 664 4.771152e+12 423484.60 *
```

Summary da Rede Neuronal

```
> print(summary(mod.rede.census_california))
a 8-8-1 network with 81 weights
options were - linear output units  decay=3
b->h1    i1->h1    i2->h1    i3->h1    i4->h1    i5->h1    i6->h1    i7->h1    i8->h1
 0.03     -3.26      0.95      1.00     96.96     18.20     45.03     16.99      0.24
b->h2    i1->h2    i2->h2    i3->h2    i4->h2    i5->h2    i6->h2    i7->h2    i8->h2
-261.33   -3.08     -3.07     -0.03      0.00      0.00      0.00      0.00      0.51
b->h3    i1->h3    i2->h3    i3->h3    i4->h3    i5->h3    i6->h3    i7->h3    i8->h3
-55.36     -0.59     -0.63      0.06      0.00      0.00      0.00      0.00      0.87
b->h4    i1->h4    i2->h4    i3->h4    i4->h4    i5->h4    i6->h4    i7->h4    i8->h4
-0.10     11.49     -3.43     -0.39     -0.20     -0.20     -0.59     -0.20     -0.23
b->h5    i1->h5    i2->h5    i3->h5    i4->h5    i5->h5    i6->h5    i7->h5    i8->h5
2245.84  -2344.57  -8705.96   85.34     70.79   -366.35   -175.14    424.30   6143.51
b->h6    i1->h6    i2->h6    i3->h6    i4->h6    i5->h6    i6->h6    i7->h6    i8->h6
 0.59    -73.62     23.28     24.76     32.87    -23.87     -6.23     -1.74      1.74
b->h7    i1->h7    i2->h7    i3->h7    i4->h7    i5->h7    i6->h7    i7->h7    i8->h7
-0.04      4.36     -1.34      2.93    -13.44    -27.07    -25.62    -13.20      0.47
b->h8    i1->h8    i2->h8    i3->h8    i4->h8    i5->h8    i6->h8    i7->h8    i8->h8
-10743.93 -160.75  -314.16    20.81     -0.70      2.90      0.20     -0.68    725.10
b->o     h1->o     h2->o     h3->o     h4->o     h5->o     h6->o     h7->o     h8->o
51829.94 34422.94 105643.44 232190.30   -0.06  22344.06   7008.87     -1.15  26682.91
```

Anexo 2

```
#
# Criação do Dataframe a partir do ficheiro de dados
#
census_california <- read.table('census_california_treino.txt', sep = ' ', header =
FALSE, dec = '.', col.names =
c('longitude','latitude','med_idade_casas','tot_divisoes','tot_quartos','populacao_viv
er','propriedades','salario_med','med_valor_casas'))

#
# Verificar se existem missing values
#
census_california[!complete.cases(census_california),]

#
# Min, 1Q, Mediana, Média, 3Q, Max
#
print(summary(census_california))

#
# Histogramas das variáveis
#
hist(census_california$longitude, prob=T)
lines(density(census_california$longitude,na.rm=T))

hist(census_california$latitude, prob=T)
lines(density(census_california$latitude,na.rm=T))

hist(census_california$med_idade_casas, prob=T)
lines(density(census_california$med_idade_casas,na.rm=T))

hist(census_california$tot_divisoes, prob=T)
lines(density(census_california$tot_divisoes,na.rm=T))

hist(census_california$tot_quartos, prob=T)
lines(density(census_california$tot_quartos,na.rm=T))

hist(census_california$populacao_viver, prob=T)
lines(density(census_california$populacao_viver,na.rm=T))

hist(census_california$propriedades, prob=T)
lines(density(census_california$propriedades,na.rm=T))

hist(census_california$salario_med, prob=T)
lines(density(census_california$salario_med,na.rm=T))

hist(census_california$med_valor_casas, prob=T)
lines(density(census_california$med_valor_casas,na.rm=T))

#
# Obtenção dos Modelos
#
# Criação dos dataframes de treino e teste utilizando a técnica habitual
# de amostragem 70/30
#
amostra <- sample(nrow(census_california), as.integer(0.7*nrow(census_california)))
amostra.treino <- census_california[amostra,]
amostra.teste <- census_california[-amostra,]

#
# Modelo regressão linear
#
# Medidas de erro de previsão calculadas
# MAE - Mean Absolute Error
# MSE - Mean Squared Error
# NMSE - Normalized Mean Squared Error
# RMSE - Root Mean Squared Error
# COR - Coeficiente de correlação

mod.linear.census_california  <- lm(med_valor_casas ~ ., data=amostra.treino)
mod.linear.previsoes     <- predict(mod.linear.census_california, amostra.teste[,-9])

mod.linear.mae    <- mean(abs(mod.linear.previsoes -
amostra.teste[,'med_valor_casas']))
```

```
mod.linear.mse    <- mean((mod.linear.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.linear.nmse   <- mean((mod.linear.previsoes -
amostra.teste[,'med_valor_casas'])^2) / mean((mean(amostra.teste[,'med_valor_casas'])
- amostra.teste[,'med_valor_casas'])^2)
mod.linear.rmse       <- sqrt(mod.linear.mse)
mod.linear.cor <- cor(amostra.teste[,ncol(amostra.teste)], mod.linear.previsoes)

print(summary(mod.linear.census_california))
cat("\nErro Previsao MAE: ",mod.linear.mae,"\n")
cat("\nErro Previsao MSE: ",mod.linear.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.linear.nmse,"\n")
cat("\nErro Previsao RMSE: ",mod.linear.rmse,"\n")
cat("\Coef. Correlação: ",mod.linear.cor,"\n")


#
# Modelo regressão linear generalizada
#
# Medidas de erro de previsão calculadas
# MAE - Mean Absolute Error
# MSE - Mean Squared Erro
# NMSE - Normalized Mean Squared Error
# RMSE - Root Mean Squared Error
# COR - Coeficiente de correlação

mod.gener.census_california  <- glm(med_valor_casas ~ ., data=amostra.treino)
mod.gener.previsoes     <- predict(mod.gener.census_california, amostra.teste[,-9])

mod.gener.mae           <- mean(abs(mod.gener.previsoes -
amostra.teste[,'med_valor_casas']))
mod.gener.mse           <- mean((mod.gener.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.gener.nmse          <- mean((mod.gener.previsoes -
amostra.teste[,'med_valor_casas'])^2) / mean((mean(amostra.teste[,'med_valor_casas'])
- amostra.teste[,'med_valor_casas'])^2)
mod.gener.rmse        <- sqrt(mod.gener.mse)
mod.gener.cor         <- cor(amostra.teste[,ncol(amostra.teste)], mod.gener.previsoes)

print(summary(mod.gener.census_california))
cat("\nErro Previsao MAE: ",mod.gener.mae,"\n")
cat("\nErro Previsao MSE: ",mod.gener.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.gener.nmse,"\n")
cat("\nErro Previsao RMSE: ",mod.gener.rmse,"\n")
cat("\Coef. Correlação: ",mod.gener.cor,"\n")


#
# Modelo Projection Pursuit
#
# Medidas de erro de previsão calculadas
# MAE - Mean Absolute Error
# MSE - Mean Squared Erro
# NMSE - Normalized Mean Squared Error
# RMSE - Root Mean Squared Error
# COR - Coeficiente de correlação

library(stats)
mod.purs.census_california    <- ppr(med_valor_casas ~ .,
data=amostra.treino,nterms=5)
mod.purs.previsoes                  <- predict(mod.purs.census_california,
amostra.teste[,-9])

mod.purs.mae                <- mean(abs(mod.purs.previsoes -
amostra.teste[,'med_valor_casas']))
mod.purs.mse                <- mean((mod.purs.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.purs.nmse               <- mean((mod.purs.previsoes -
amostra.teste[,'med_valor_casas'])^2) / mean((mean(amostra.teste[,'med_valor_casas'])
- amostra.teste[,'med_valor_casas'])^2)
mod.purs.rmse         <- sqrt(mod.purs.mse)
mod.purs.cor          <- cor(amostra.teste[,ncol(amostra.teste)], mod.purs.previsoes)

print(summary(mod.purs.census_california))
cat("\nErro Previsao MAE: ",mod.purs.mae,"\n")
cat("\nErro Previsao MSE: ",mod.purs.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.purs.nmse,"\n")
```

```
cat("\nErro Previsao RMSE: ",mod.purs.rmse,"\n")
cat("\Coef. Correlação: ",mod.purs.cor,"\n")



#
# Modelo MARS
#
# Medidas de erro de previsão calculadas
# MAE - Mean Absolute Error
# MSE - Mean Squared Erro
# NMSE - Normalized Mean Squared Error
# RMSE - Root Mean Squared Error
# COR - Coeficiente de correlação

library(mda)
mod.mars.census_california  <- mars(amostra.treino[,-9],amostra.treino[,9])
mod.mars.previsoes     <- predict(mod.mars.census_california, amostra.teste[,-9])

mod.mars.mae           <- mean(abs(mod.mars.previsoes -
amostra.teste[,'med_valor_casas']))
mod.mars.mse           <- mean((mod.mars.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.mars.nmse          <- mean((mod.mars.previsoes -
amostra.teste[,'med_valor_casas'])^2) / mean((mean(amostra.teste[,'med_valor_casas'])
- amostra.teste[,'med_valor_casas'])^2)
mod.mars.rmse          <- sqrt(mod.mars.mse)
mod.mars.cor           <- cor(amostra.teste[,ncol(amostra.teste)], mod.mars.previsoes)

print(summary(mod.mars.census_california))
cat("\nErro Previsao MAE: ",mod.mars.mae,"\n")
cat("\nErro Previsao MSE: ",mod.mars.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.mars.nmse,"\n")
cat("\nErro Previsao RMSE: ",mod.mars.rmse,"\n")
cat("\Coef. Correlação: ",mod.mars.cor,"\n")


#
# Modelo Árvore de Regressão
#
# Medidas de erro de previsão calculadas
# MAE - Mean Absolute Error
# MSE - Mean Squared Erro
# NMSE - Normalized Mean Squared Error
# RMSE - Root Mean Squared Error
# COR - Coeficiente de correlação

library(rpart)
mod.arv.census_california  <- rpart(med_valor_casas ~ ., data=amostra.treino)
mod.arv.previsoes     <- predict(mod.arv.census_california, amostra.teste[,-9])

mod.arv.mae            <- mean(abs(mod.arv.previsoes -
amostra.teste[,'med_valor_casas']))
mod.arv.mse            <- mean((mod.arv.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.arv.nmse   <- mean((mod.arv.previsoes - amostra.teste[,'med_valor_casas'])^2) /
mean((mean(amostra.teste[,'med_valor_casas']) - amostra.teste[,'med_valor_casas'])^2)
mod.arv.rmse   <- sqrt(mod.arv.mse)
mod.arv.cor            <- cor(amostra.teste[,ncol(amostra.teste)], mod.arv.previsoes)

plot(mod.arv.census_california)
text(mod.arv.census_california)

print(summary(mod.arv.census_california))
cat("\nErro Previsao MAE: ",mod.arv.mae,"\n")
cat("\nErro Previsao MSE: ",mod.arv.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.arv.nmse,"\n")
cat("\nErro Previsao RMSE: ",mod.arv.rmse,"\n")
cat("\Coef. Correlação: ",mod.arv.cor,"\n")


#
# Modelo Redes Neuronais
#
# Medidas de erro de previsão calculadas
# MAE - Mean Absolute Error
# MSE - Mean Squared Erro
```

```
# NMSE - Normalized Mean Squared Error
# RMSE - Root Mean Squared Error
# COR - Coeficiente de correlação

library(nnet)
mod.rede.census_california  <- nnet(med_valor_casas ~ ., data=amostra.treino,
size=20,decay=0.001,maxit=1000,linout=T)
mod.rede.previsoes     <- predict(mod.rede.census_california, amostra.teste[,-9])

mod.rede.mae           <- mean(abs(mod.rede.previsoes -
amostra.teste[,'med_valor_casas']))
mod.rede.mse           <- mean((mod.rede.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.rede.nmse          <- mean((mod.rede.previsoes -
amostra.teste[,'med_valor_casas'])^2) / mean((mean(amostra.teste[,'med_valor_casas'])
- amostra.teste[,'med_valor_casas'])^2)
mod.rede.rmse          <- sqrt(mod.rede.mse)
mod.rede.cor           <- cor(amostra.teste[,ncol(amostra.teste)], mod.rede.previsoes)

print(summary(mod.rede.census_california))
cat("\nErro Previsao MAE: ",mod.rede.mae,"\n")
cat("\nErro Previsao MSE: ",mod.rede.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.rede.nmse,"\n")
cat("\nErro Previsao RMSE: ",mod.rede.rmse,"\n")
cat("\Coef. Correlação: ",mod.rede.cor,"\n")


#
# Configuração da Rede Neuronal ERRO = RMSE
#
config.nn <- function(Alternativas, Dados)
       {
        Alternativas$RMSE <- rep(NA, nrow(Alternativas))
        treino.idxs <- sample(nrow(Dados), as.integer(0.7 * nrow(Dados)))
        treino <- Dados[treino.idxs, ]
        teste <- Dados[-treino.idxs, ]
        for (a in 1:nrow(Alternativas))
              {
                     nn <- nnet(as.formula(paste(colnames(treino)[ncol(treino)], "~
.")), treino, size = Alternativas[a, "size"],
                     decay = Alternativas[a, "decay"], maxit = 1000, linout = T)
                     prevs <- predict(nn, teste)
                     Alternativas[a, "RMSE"] <- sqrt( mean((teste[, ncol(teste)] -
prevs)^2) )
              }
        list(Resultados = Alternativas, Melhor =
Alternativas[which.min(Alternativas$RMSE),])
 }

res <- config.nn(expand.grid(size = c(30, 40, 50), decay = c(0.01, 0.001, 0.001)),
census_california)
print(res)

#
# Configuração com medida de erro = NMSE
#
config.nn.nmse <- function(Alternativas, Dados)
       {
        Alternativas$NMSE <- rep(NA, nrow(Alternativas))
        treino.idxs <- sample(nrow(Dados), as.integer(0.7 * nrow(Dados)))
        treino <- Dados[treino.idxs, ]
        teste <- Dados[-treino.idxs, ]
        for (a in 1:nrow(Alternativas))
              {
                     nn <- nnet(as.formula(paste(colnames(treino)[ncol(treino)], "~
.")), treino, size = Alternativas[a, "size"],
                     decay = Alternativas[a, "decay"], maxit = 1000, linout = T)
                     prevs <- predict(nn, teste)
                     Alternativas[a, "NMSE"] <- mean((prevs - teste[,ncol(teste)])^2)
/ mean((mean(teste[,ncol(teste)]) - teste[,ncol(teste)])^2)
              }
        list(Resultados = Alternativas, Melhor =
Alternativas[which.min(Alternativas$NMSE),])
 }
res.nmse <- config.nn.nmse(expand.grid(size = c(5, 10, 15), decay = c(0.01, 0.001,
0.001)), census_california)
print(res.nmse)
```

```
#
# Código usado para ir experimentando os novos valores
#
mod.rede.census_california  <- nnet(med_valor_casas ~ ., data=amostra.treino,
size=10,decay=0.001,maxit=1000,linout=T)
mod.rede.previsoes     <- predict(mod.rede.census_california, amostra.teste[,-9])

mod.rede.mae           <- mean(abs(mod.rede.previsoes -
amostra.teste[,'med_valor_casas']))
mod.rede.mse           <- mean((mod.rede.previsoes -
amostra.teste[,'med_valor_casas'])^2)
mod.rede.nmse          <- mean((mod.rede.previsoes -
amostra.teste[,'med_valor_casas'])^2) / mean((mean(amostra.teste[,'med_valor_casas'])
- amostra.teste[,'med_valor_casas'])^2)
mod.rede.rmse          <- sqrt(mod.rede.mse)
mod.rede.cor           <- cor(amostra.teste[,ncol(amostra.teste)], mod.rede.previsoes)

print(summary(mod.rede.census_california))
cat("\nErro Previsao MAE: ",mod.rede.mae,"\n")
cat("\nErro Previsao MSE: ",mod.rede.mse,"\n")
cat("\nErro Previsao NMSE: ",mod.rede.nmse,"\n")
cat("\nErro Previsao RMSE: ",mod.rede.rmse,"\n")
cat("\Coef. Correlação: ",mod.rede.cor,"\n")


#
# Para facilitar a avaliação dos modelos
# vai-se construir um dataframe com os erros de cada um
# e o Coef. Correlação
#

erro.prev.census_california <- data.frame(mod.linear.mae=mod.linear.mae,
                                  mod.linear.mse=mod.linear.mse,
                                           mod.linear.rmse=mod.linear.rmse,
                                  mod.linear.nmse=mod.linear.nmse,
                                           mod.linear.cor=mod.linear.cor,
                                  mod.gener.mae=mod.gener.mae,
                                  mod.gener.mse=mod.gener.mse,
                                  mod.gener.rmse=mod.gener.rmse,
                                  mod.gener.nmse=mod.gener.nmse,
                                  mod.gener.cor=mod.gener.cor,
                                  mod.purs.mae=mod.purs.mae,
                                  mod.purs.mse=mod.purs.mse,
                                           mod.purs.rmse=mod.purs.rmse,
                                  mod.purs.nmse=mod.purs.nmse,
                                  mod.purs.cor=mod.purs.cor,
                                  mod.mars.mae=mod.mars.mae,
                                  mod.mars.mse=mod.mars.mse,
                                           mod.mars.rmse=mod.mars.rmse,
                                  mod.mars.nmse=mod.mars.nmse,
                                  mod.mars.cor=mod.mars.cor,
                                  mod.arv.mae=mod.arv.mae,
                                  mod.arv.mse=mod.arv.mse,
                                           mod.arv.rmse=mod.arv.rmse,
                                  mod.arv.nmse=mod.arv.nmse,
                                  mod.arv.cor=mod.arv.cor,
                                  mod.rede.mae=mod.rede.mae,
                                  mod.rede.mse=mod.rede.mse,
                                           mod.rede.rmse=mod.rede.rmse,
                                  mod.rede.nmse=mod.rede.nmse,
                                  mod.rede.cor=mod.rede.cor)

print(erro.prev.census_california[,1:5])
print(erro.prev.census_california[,6:10])
print(erro.prev.census_california[,11:15])
print(erro.prev.census_california[,16:20])
print(erro.prev.census_california[,21:25])
print(erro.prev.census_california[,26:30])

#
# Após a análise dos erros de previsão optei
# pelo modelo
# que agora aplico aos dados de teste fornecidos
#
```

```
teste.census_california <- read.table('census_california_teste.txt', sep = ' ', header
= FALSE, dec = '.', col.names =
c('longitude','latitude','med_idade_casas','tot_divisoes','tot_quartos','populacao_viv
er','propriedades','salario_med'))

previsao.census_california <- predict(mod.purs.census_california,
teste.census_california)
write.table(previsao.census_california, 'ecd_regressao_previsao_antonio_castro.txt')
```