XI CONGRESSO BRASILEIRO DE ENGENHARIA DE AVALIAÇÕES E PERÍCIAS XI COBREAP

DUAS FERRAMENTAS PODEROSAS À DISPOSIÇÃO DO ENGENHEIRO DE AVALIAÇÕES – MODELOS LINEARES GENERALIZADOS E REDES NEURAIS

GUEDES, JACKSON CARVALHO
Eng. Civil, *M.Sc.*Engª.de Produção
CREA nº 45.428-D/RJ
IEL-RJ nº 1.292
Rua Araújo Lima, 124 - Andaraí, Rio de Janeiro, RJ, CEP 20541-050
Tel. 0-xx-21-25710016, e-mail: jacksonguedes@petrobras.com.br

Resumo: Este trabalho apresenta , de forma resumida, os fundamentos de inteligência artificial e redes neurais. Compara também os resultados de uma avaliação de lotes apresentada na Dissertação de Mestrado do Engenheiro Rubens Alves Dantas, onde o tratamento dos dados foi feito usando-se Modelos Lineares Generalizados com os resultados obtidos com uma rede neural treinada.

Abstract. This paper presents the fundaments of artificial inteligence and neural networks. It also compares the perfomance of Generalized Linears Models and neural networks in a data treatement presentend by the Engineer Rubens Alves Dantas in its Master Dissertation.

GUEDES, JACKSON CARVALHO
PETROBRAS S.A.

1. INTRODUÇÃO

O artigo *O Emprego da Inteligência Artificial nos Problemas de Avaliação de Bens*, de 1995,[1] comparou os resultados de uma avaliação feita com auxílio de Análise de Regressão Linear Múltipla, suportada pelo Método dos Mínimos Quadrados Ordinários com os resultados obtidos por uma rede neural treinada no software Braimaker®, mostrando que os disvios, medido pelo erro quadrático médio, apresentados pela rede neural eram bem menores do que aqueles obtidos pelo MQO.

Lembrou ainda que a avaliação de bens poderia utilizar técnicas tão imediatas como o simples ajustamento de curvas até metodologias mais sofisticadas como a regressão múltipla e redes neurais.

Destacou-se que o emprego da análise de regressão linear, que representou um grande avanço no estado da arte da Engenharia de Avaliações e se popularizou entre os avaliadores; lembrando que os IBAPES têm promovido cursos em todos os rincões do País, disseminando o conhecimento teórico e prático da matéria, e que os softwares voltados para o assunto se multiplicam e são comercializados a preços mais acessíveis.

Entretanto nada se comentou sobre o emprego de Modelos Lineares Generalizados que, embora seja uma ferramenta poderosa, efetivamente apresenta equações não tão simples como as que suportam o método dos mínimos quadrados e, talvez o mais importante, sem a mesma disponibilidade de softwares no mercado, mormente aqueles em língua portuguesa, *costumizados* para engenharia de avaliações.

O objetivo deste trabalho é apresentar, resumidamente, os conceitos de Modelos Lineares Generalizados (MGL) e os conceitos de inteligência artificial e de redes neurais e comparar as estimativas obtidas com uma aplicação com MGL com as estimativas obtidas pela aplicação de redes neurais.

Na verdade trata-se de exercício similar aos apresentando pelo autor no IX Congresso Brasileiro de Engenharia de Avaliações e Perícias, realizado em Florianópolis, 1995 e no 1º Seminário do Latin American Real Estate Society.

Para a realização desse exercício de comparação aproveitamos os dados analisados por Dantas, em sua dissertação de mestrado[2] e também apresentados no VI Cobreap, em Pernambuco.

2. FUNDAMENTAÇÃO TEÓRICA

Mínimos quadrados ordinários

Em poucas palavras, na avaliação empregando a análise de regressão linear o objetivo é encontrar uma relação funcional entre as mudanças no valor e o fator ou fatores dos quais o valor depende.

[1] GUEDES, Jackson Carvalho - O Emprego da Inteligência Artificial nos Problemas de Avaliação de Bens; Anais do VIII Congresso Brasileiro de Engenharia de Avaliações e Perícias, ICAPE- Instituto Catarinense de Engenharia de Avaliações e Perícias – 1995, pag. 368-374.
[2] DANTAS, Rubens Alves - Avaliação de Glebas Inseridas na Malha Urbana – Dissertação de Mestrado – Depto. De Engenharia Civil – UFPE - 1987.

A análise de regressão linear conduz a um modelo estatístico que descreve o relacionamento da variável dependente, ou seja, do valor do bem, com as variáveis independentes, quais sejam, aquelas que influenciam na formação do valor.

O modelo obtido via análise de regressão, além de permitir a predição do valor do bem a avaliar, fornece elementos para entender quais os atributos influenciam na formação desse valor, de que forma e com que peso.

O modelo apresenta-se, em geral, na seguinte forma:

$$Y = B_0 + B_1 X_1 + B_2X_2 + ...+ B_k X_k ,$$

onde Y , denominada variável dependente, nos trabalhos específicos de avaliação pode representar o valor unitário ou total do bem em estudo, as variáveis Xi representam os atributos formadores dos valores e os parâmetros Bi denominados coeficientes da regressão ou regressores representam o peso que as variáveis explicativas têm na formação do valor.

Existem vários métodos para a estimação dos parâmetros de uma regressão, porém o Método dos Mínimos Quadrados é o mais utilizado. Conforme o método adotado, algumas hipóteses básicas precisam ser obedecidas para que se obtenha o melhor estimador linear não tendencioso e alguns testes de validade do modelo são exigidos pela Norma para Avaliação de Imóveis Urbanos - NB-502, dependendo o rigor da avaliação do atingimento de padrões determinados.

Modelos lineares generalizados

A teoria e o uso de modelos lineares generalizados foi exposto por Nelder e Wedderburn em 1992. Desde então, com o apoio do software GLIM, muitos têm se beneficiado desta metodologia para modelagem e ajustamento. Em sua forma mais simples, o modelo linear generalizado é especificado por:

1. Observações independentes y_i, ...y_n distribuídas de acordo com uma família exponencial,

2. Um conjunto de variáveis explicativas x, disponível para cada observação, descrevendo o componente linear sistemático através de $Y = x^T\beta$, e

3. A função de ligação $g(\nabla)$, monótona e diferenciável, entre Y e μ, de tal forma que $g(\mu) = Y$.

Do ponto de vista do usuário, o procedimento de ajustamento envolve:

- A escolha da distribuição de erro relevante;
- Determinar que variáveis incluir na componente sistemática, e
- Definir a função de ligação $g(\mu)$.

Alguns casos particulares importantes de variáveis pertencentes à família de distribuição exponencial correspondem, entre outras, às distribuições Binomial, Gama, Normal e Poisson, sendo de interesse neste trabalho as distribuições Normal e Gama.

Como foi suposto que a distribuição dos dados pertence à família exponencial, a função de verossimilhança para os modelos lineares generalizados tem a seguinte forma geral:

$$L(\beta) = \sum [\phi_i \{yi\theta_i - b(\theta_i)\} + c(y_i ; [\phi_i)]$$

Onde:

b(∇) e c(∇) são funções conhecidas, sendo que as duas primeiras derivadas de b(θ) correspondem respectivamente a média e a função de variância dos dados y; $\phi_i > 0$ é o parâmetro de escala da distribuição, suposto conhecido e ϕ_i é o parâmetro canônico da distribuição, que é função de β.

Nesta estrutura, o modelo linear generalizado pode ser ajustado e sua adequação examinada. Em particular, os resíduos devem ser examinados e plotados para determinar se ainda permanece qualquer componente sistemática.

Uma das dificuldades para encontrar as estimativas de máxima verossimilhança β do modelo é que o sistema resultante dado por $\delta L(\beta)/\delta \beta = 0$ é não-linear. Para resolve-los é necessário a aplicação de métodos de cálculo numérico, como por exemplo o algoritmo denominado escore, que é uma modificação do método de Newton-Raphson, que, após alguma álgebra para o cálculo de β, fica com a seguinte estrutura (McCullagh & Nelder, 1989).

$$X'W^{(m)} \phi X\beta^{(m+1)} = X'W^{(m)} \phi y^{*(m)}$$

onde:

(m): m-ésimo passo do algrítmo

W e ϕ são matrizes diagonais com $W = diag\{w_1.....w_n\}$ com $w_1 = (d\mu/d\eta)^2/V$

e $\Phi = diag\{\Phi_1..... \Phi_n\}$ onde são parâmetros de escala.

$y^{*(m)}$ é a variável resposta modificada da forma: $y^* = X\beta + H(y-\mu)$, onde

H é uma matriz diagonal dada por $H = diag\{ d\eta_1/d\mu_1.....d\eta_n/d\mu_n\}$.

Este algoritmo que é implementado no GLIM é denominado Interactively Reweighted Least Squares.

Redes neurais

O cérebro humano é constituído por bilhões de células chamadas neurônios. Cada uma dessas células é como um pequeno computador com capacidades extremamente limitadas, entretanto, quando conectadas entre si, formam o mais inteligente sistema conhecido.

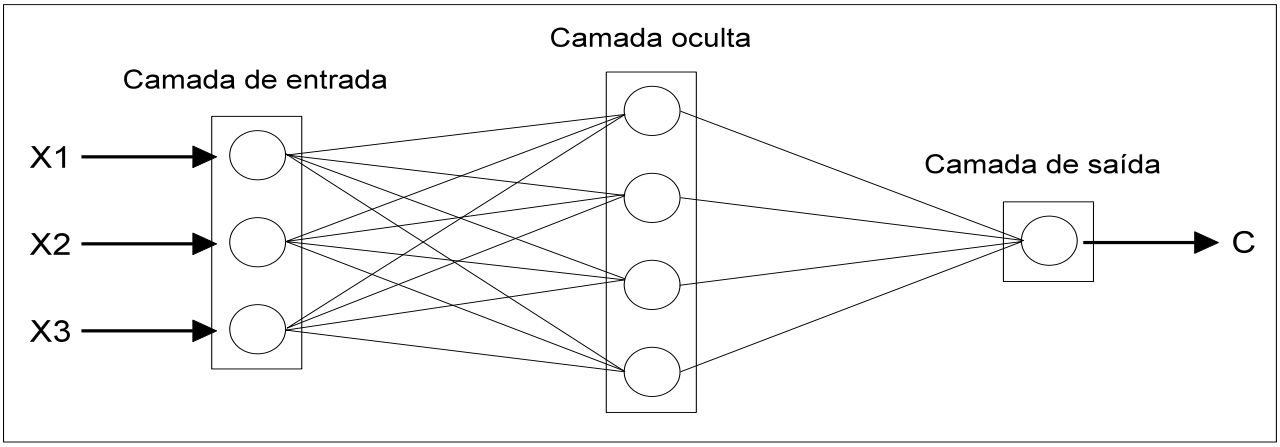
Redes neurais são uma nova classe de sistemas computacionais formados por centenas ou milhares de neurônios artificiais conectados entre si, de maneira similar ao que ocorre no cérebro humano.

Pode-se treinar as redes neurais apresentando-se a elas fatos, isto é , pares de dados de entrada e saída, permitindo a elas fazerem associações, descobrindo assim a existência de algum padrão de comportamento.

O programa utilizado nesta aplicação, chamado BrainMaker, emprega um tipo específico de rede neural, chamada de rede de retro-propagação ou de encadeamento para trás.; ele aprende da mesma maneira que as pessoas, isto é, pelo exemplo e repetição de fatos, que são constituídos de dados de entrada e saída.

A rede é treinada apresentando-se a ela um conjunto de fatos (entradas/saídas) repetidas vezes. Cada vez que os dados de entrada são apresentados ela retorna uma resposta com resultados que ela pensa ser o correto, comparando-a com o fato real ou padrão. Quando sua resposta é incorreta ela se corrige internamente. Após percorrer toda a lista de fatos, apresentando um fato por vez e fazendo as correções necessárias, o programa revê todo o rol recursivamente, até que todas as respostas sejam consideradas aceitáveis.

Em sua forma mais simples uma rede neural consiste de três camadas: uma camada de dados de entrada, uma camada oculta e uma camada de dados de saída, conforme representado a seguir:



Para melhor entendimento do processo faz-se a seguir uma comparação com a aplicação de modelos lineares generalizados onde, após a entrada de dados constituídos, por exemplo, dos valores de oferta ou venda dos imóveis pesquisados e de seus atributos (área, padrão construtivo, localização, idade, etc), busca-se uma função de ligação relacionando-se o valor (variável dependente) com os atributos (variáveis independentes). Uma vez fixadas as formas das variáveis explicativas no modelo, os regressores determinarão o peso de cada um dos atributos nos resultados, da mesma forma que ocorre no modelo de regressão obtido pelos mínimos quadrados ordinários.

A rede neural, através de seu mecanismo de retro-propagação, que se dá entre a camada de entrada e a camada oculta, ao processar os sinais de entrada, que são as informações da pesquisa imobiliária, também atribui pesos aos atributos, por intermédio de uma função de transferência. Ela porém não se expressa de maneira simples e direta através de uma equação onde o usuário possa conferir os parâmetros e saber como os atributos refletem no valor do bem.

Na modelagem por MGL ou por MQO, a leitura da equação inferida permite que se visualize como os atributos de determinado imóvel influenciam no seu preço . O sinal e a grandeza de cada um dos regressores mostra em que sentido e em que proporção as variáveis participam da formação do valor.

Em alguns casos, dependendo da complexidade do modelo, é mais fácil analisar as influências dos atributos, expressos pelas variáveis independentes, fazendo-se simulações, provocando variações de valores do atributo em estudo, mantendo-se as demais variáveis constantes.

As redes neurais porém, não expressam sua função ou funções de transferência de maneira simples para que o usuário possa compreender e quantificar de imediato as influências dos atributos do bem avaliando. Isto não significa que as equações não possam ser explicitadas. Apenas são mais complexas.

Freqüentemente as funções de transferência das redes neurais são matematicamente bastante sofisticadas. O programa BrainMaker, por exemplo, usa na maioria das vezes uma função de transferência sigmóide, podendo entretanto, à vontade do projetista da rede empregar funções lineares, em degrau, gaussiana, etc.

A melhor e mais acessível maneira de se analisar o desempenho de uma rede neural é por meio de simulações dos resultados.

Mas vale a pena entender um pouquinho a matemática utilizada no treinamento de uma rede neural; descreve-se a seguir, de forma sucinta, os passos que uma rede neural artificial com retro-propagação.

Cálculo do erro quadrático de um neurônio

Para calcular o erro quadrático de um neurônio é utilizada a seguinte fórmula:

$$(d_i - o_i)^2 = d_i^2 - o_i^2 - (2.d_i \,.\, o_i)$$

onde di é designado saída (output) do neurônio e oi é a verdadeira saída, ou seja, o valor observado. Ao adaptar os pesos, tentamos minimizar o erro quadrático médio. O erro quadrático médio é:

$$\Sigma\, (d_i^2 - o_i^2)/N$$

Onde N é o número de neurônios. As funções de erro quadráticos médios são parábolas. A melhor solução de mínimos quadrados corresponde ao fundo da curva.
A solução é obtida encontrando-se o gradiente da função do erro quadrático médio. Se estamos trabalhando com dois pesos, o erro quadrático médio é quadrático nos pesos. O gradiente é a primeira potência dos pesos, neste caso uma função linear dos pesos.

O algoritmo descendente é baseado em implementar correções que são proporcionais ao gradiente. Então, a correção é proporcional a uma função linear dos pesos. Isto significa que estamos usando feedback para nos empurrar para o fundo do vale. Podemos usar a teoria linear de feedback para descrever a dinâmica do comportamento dos pesos.

A Matemática da Retropropagação

Na implementação do BrainMaker um neurônio recebe inputs somente da camada de neurônios anterior e envia outputs somente para os inputs dos neurônios da próxima camada. A camada 2 recebe os sinais das saídas da camada 1. A camada 2 envia sinais para a entrada da camada 3. Cada camada pode ser interpretada como um vetor de saídas dos neurônios. As forças das conexões entre cada duas camadas constituem os elementos de uma matriz de valor de valores reais. É denominada matriz de pesos W. Wij representa o peso da conexão do neurônio j com o neurônio i.

Se tivermos N pares de input/output que precisam ser aprendidos, podemos indexar estes pares com a letra p; onde os valores de p percorrem de 1 a N. Designamos o p-ésimo input como $input_p$, e o correspondente output desejado como $Pattern_p$. Devemos forçar a variação dos pesos de forma que finalmente atinja uma rede que delineie $Input_p$ ao $Pattern_p$ para todos os valores de p.

Vamos designar a saída de cada neurônio individual com o índice i como $Output_i$. De forma similar a ativação do neurônio i como A_i. Existe uma função de transferência TF, que deve ser contínua e diferenciável, de forma que $Output_i = TF(A_i)$.

$Pattern_{pi}$ representará a saída alvo para o i-ésimo neurônio da camada de saída da rede, no p-ésimo par input-outup. $Output_{pi}$ será a saída real para cada neurônio. O $outputi_{pi}$ inicialmente não será igual ao $Pattern_{pi}$, porque a rede começa a rodar ainda não treinada. Precisamos definir o erro no $pattern_p$, da i-ésima saída do neurônio como:

$$Erro_{pi} = ½(Pattern_{pi} – Output_{pi})^2$$

Elevar ao quadrado assegura que todos os erros são positivos, e o fator ½ é para simplificar a matemática mais tarde. O erro total no $Pattern_p$ é agora:

$$Erro_p = ½ \Sigma i\ (pattern_{pi} – Output_{pi})^2$$

O erro total para todos os exemplos é a soma dos erros em cada exemplo sobre todo p:

$$Erro = \Sigma p\ Erro$$

$$=½ \Sigma p \Sigma i (pattern_{pi} – Output_{pi})^2$$

O treinamento de uma rede neural para associar padrões de entrada e saída pode ser visto como um problema de minimização, onde a quantidade a ser minimizada é E, o erro total de todos os exemplos. As variáveis independentes a serem usadas na minimização são os W_{ij}. Desde que as menores redes têm centenas de neurônios e milhares de conexões , estamos falando em minimizar um campo escalar sobre um vetor no espaço com centenas de dimensões.

Gradiente descendente

O método simples para achar o mínimo é conhecido como gradiente descendente ou descendente íngreme. Não é um algoritmo computacionalmente ótimo, mas é aceitável. O gradiente descendente involve mover um pequeno degrau para baixo o gradiente local do campo escalar. É diretamente análogo a um esquiador sempre se movendo para baixo da montanha, até chegar ao sopé. Um grande obstáculo é que é possível atingir-se um mínimo local ao invés do mínimo global, que é o objetivo. O algoritmo empaca neste mínimo local até que algum impulso é adicionado aos pesos, tirando o algoritmo do falso mínimo.

Se a mudança no peso W_{ij} no exemplo p é denotada por $\Delta p\ W_{ij}$, então teremos, como o gradiente descendente no erro E_{pi}, o seguinte:

$$\Delta p\ W_{ij} = \eta * -\delta\ E_{pi} / \delta\ W_{ij} \text{ , onde } \eta \text{ é alguma constante}$$

Note que desde que $\delta E/\delta W_{ij} = \Sigma p\ \delta E_{pi} / \delta W_{ij}$

Este algoritmo não implementa o verdadeiro gradiente descendente em E se os pesos são alterados a cada apresentação do exemplo. Entretanto já foi verificado que funciona na maioria absoluta dos casos.

A regra da cadeia

A regra da cadeia nos permite dizer:

$$\delta E_{pi} /\delta W_{ij} = (\delta E_{pi} / \delta A_{pi} / \delta W_{ij})$$

Sabemos que

$$A_{pi} = = \Sigma_j W_{ij} O_{pi}$$

Então

$$\delta A_{pi} / \delta W_{ij} = O_{pj}$$

Então definimos

$$\delta_{pi} = -\delta E_{pi} / \delta A_{pi} , \text{ de forma que}$$

$$\Delta_p W_{ij} = \eta \delta_{pi} O_{pj} , \eta \text{ é constante}$$

Desde que

$$\delta E_{pi} / \delta A_{pi} = \delta E_{pi} / \delta O_{pi} \delta O_{pi} / \delta A_{pi}$$

e

$$Opi = TF(A_{pi}),$$

Temos

$$\delta_{pi} / \delta A_{pi} = - (\delta E_{pi} / \delta O_{pi}) TF'(A_{pi})$$

Se o neurônio i está na camada de saída, podemos computar $\delta E_{pi}/\delta O_{pi}$ imediatamente da definição de E_{pi}.

$$Erro_{pi} = ½ \Sigma i (pattern_{pi} – Output_{pi})^2$$

$$Erro_{pi} = ½ (T_{pi} – O_{pi})^2$$

$$\delta E_{pi} =/\delta O_{pi} = -(T_{pi} – O_{pi})$$

O fator ½ foi incluído para cancelar o 2 da diferenciação anterior. Desde que O = TF(A), $\delta O_{pi} /\delta A_{pi}$,=dTF/dA. Esta é a razão porque se exige que as funções de transferência dos neurônios sejam continuamente diferenciáveis. Então, para cada neurônio i na camada de saída, podemos escrever:

$$\delta_{pi} =- (\delta E_{pi} / \delta O_{pi})(TF´(A_{pi})$$

$$= (T_{pi} - O_{pi})(TF´(A_{pi})$$

Se o neurônio não está na camada de saída, usamos a regra da cadeia para obter:

$$\delta E_{pi} / \delta O_{pi} = \Sigma (\delta E_{pi} / \delta A_{pk})(\delta A_{pk} / \delta O_{pi})$$

$$= \Sigma_k - \delta_{pk} W_{ki}$$

então

$$\delta_{pi} =TF´(A_{pi}) \Sigma_k d_{pk} W_{ki} ,$$

se o neurônio não está na camada de saída. Estes k0s são conhecidos como os erros de sinais locais, e são retropropagados durante o treinamento, daí o nome do algoritmo. Basicamente treinar consiste em rodar padrões através da rede para frente e propagar os erros para trás e atualizar os pesos de acordo com a equação

$$\Delta_p W_{ij} = \eta \delta_{pi} O_{pj}$$

Onde η é uma constante conhecida como taxa de aprendizagem. A versão atual do Brainmaker usa a versão da regra usada por Sejnowski e Rosemberg em sua aplicação Nettalk, onde:

$$\Delta_p W_{ij} = \eta (1-\mu)\delta_{pi} O_{pj} + \mu \Delta_{p-1} W_{ij})$$

Aqui, μ é outro parâmetro conhecido como um fator de abrandamento (smoothing factor). Ele melhora a convergência, mesmo que μ seja fixado como zero, o algoritmo ainda assim convergirá, embora levando mais tempo.

A função de transferência TF que geralmente usamos é a logística (sigmóide); sua forma geral é:

$$TF(A) = (\text{Lim.Sup.} - \text{Lim. Inf.})/(1 + e^{(-\text{ganho} * (A - \text{Centro.})} + \text{Lim. Inf.}$$

Se fixamos Limite Superior igual a 1, Limite Inferior igual a 0, ganho igual a 1, Centro igual a 0, a fórmula se reduz a

$$TF(A) = 1/(1+e^{-A})$$

$$TF`(A) = e^{-A}/(1+e^{-A})^2$$

$$=1/(1+e^{-A}) * - e^{-A} /(1+e^{-A})$$

$$= TF(A) * (1-TF(A))$$

4. APLICAÇÃO

Base de dados

Para comparar as duas técnicas utilizou-se os dados apresentados por Dantas, referentes a 50 lotes urbanos nos bairros de Casa Forte, Torre e Iputinga, em Recife.

Os dados, em sua totalidade podem ser vistos no Apêndice A, e tinham como variável dependente o valor e como variáveis explicativas a testada efetiva, a profundidade equivalente, o nível de urbanização, a natureza do evento, se oferta ou transação, e a localização, tratadas como variáveis dicotômicas.

Avaliação utilizando o Glim

Em sua dissertação Dantas analisou quatro modelos, um normal linear, um modelo normal linear com transformação log na variável dependente, um terceiro modelo utilizando as transformações para a testada e profundidade sugeridos pela NBR-5676 e finalmente um modelo com erro gama.

O fato de que em avaliações de bens, os valores da variável dependente serem sempre positivos costumam indicar o modelo Gama como uma boa opção para o ajuste.

O último modelo foi escolhido para comparação os resultados obtidos pelo Brainmaker.

O erro quadrático médio (EQM) foi calculado para os 50 eventos usados na equação obtida pelo GLIM, como segue:

$$EQM = \frac{\sum (Y_i - Y_{real})^2}{N} = 18.356.054$$

onde:

Y_i = valor do imóvel i calculado pela equação inferida,
Y_{real} = valor do imóvel i, na pesquisa,
N = número total de dados da pesquisa.

Rede neural

As mesmas informações, sem modificações na forma de entrada dos dados, foram usadas para avaliação, empregando-se o programa BrainMaker V.3.2, programado para a criação de redes neurais.

Por tratar-se de um assunto ainda não tão bem conhecido em nosso meio como a análise de regressão, antes de apresentar-se os resultados obtidos, descreve-se de forma bem sucinta os passos necessários para projetar-se uma rede neural:

1 - o projetista precisa decidir o que ele quer que a rede neural prediga ou reconheça, equivale a escolher a variável dependente na análise de regressão, tanto no MQO ou MLG;

2 - deve-se ainda definir que informações serão usadas para as predições, ou seja, quais as variáveis explicativas;

3 - após a entrada de dados no programa e a definição do status das variáveis, monta-se e treina-se a rede neural, de acordo com parâmetros definidos pelo projetista;

4 - finalmente testa-se a rede neural com os dados separados para a validação cruzada. O Brainmaker separa cerca de 10% dos dados para testar se sua resposta *confere* com a realidade observada.

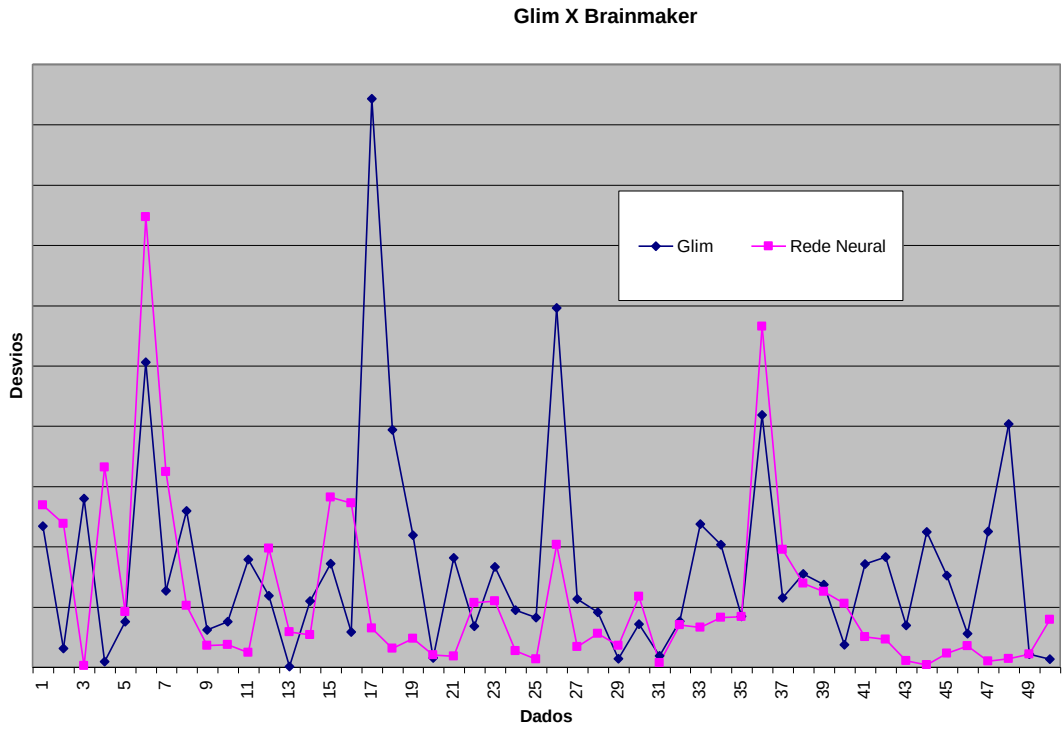
O quarto passo é fundamental, visto que não são transparentes os cálculos feitos na camada oculta da rede neural, ficando o analista limitado a conhecer as informações de entrada, resultado de sua coleta de dados, e de saída , fornecidas pelo programa.

Comparação de resultados

Observou-se que, em geral, mas não em todos os eventos, os resultados obtidos pela rede neural treinada eram melhores que os obtidos pelo modelo linear generalizado, e o valores unitários calculados (Y_i) aproximavam-se mais dos valores obtidos na coleta de dados (Y_{real}), resultando no seguinte erro quadrático médio:

$$EQM = \frac{\sum (Y_i - Y_{real})^2}{N} = 1.472.211$$

O gráfico a seguir mostra os desempenho comparativo entre as duas ferramentas.



Para uma melhor visualização registra-se em um mesmo quadro os erros quadráticos médios obtidos com o conjunto de dados usados na modelagem da equação de regressão linear e no treinamento da rede neural:

ORIGEM	EQM	%
MLG	18.356.054	100,00
Rede neural	1.472.211	8,02

5. CONCLUSÃO

Os resultados confirmaram com um caso prático o que a teoria afirmava a priori: que as predições via redes neurais deveriam ser melhores do aquelas obtidas com a análise de regressão linear utilizando mínimos quadrados ordinários ou modelos lineares generalizados . No caso em estudo, o erro quadrático médio obtido via rede neural foi situou-se em torno de 10% do que aquele gerado pela modelo linear generalizado, entretanto não se espera que diferença tão significativa ocorra sempre. A grande diferença no total deveu-se a algumas grandes diferenças pontuais e não ao melhor desempenho na grande maioria dos dados.

O melhor desempenho da rede neural deve-se, fundamentalmente , ao fato de os fenômenos sociais e relações do mundo real , expressos por variáveis, não serem necessariamente lineares. Mesmo quando lineariza-se a função, transformando-se as variáveis para melhor captar essa relação não retilínea, continua havendo a possibilidade de existir um melhor estimador não linear.

Ao final listamos os resultados para comparação, verificando-se que a rede neural respondeu melhor que o modelo de regressão em 29 vezes das 50 possíveis.

Neste trabalho partiu-se do pressuposto que o melhor modelo é aquele que apresenta os menores desvios em relação aos verdadeiros valores pesquisados. Recomendamos a leitura do excelente ensaio, publicado na Revista Brasileira de Estatística denominado Seleção de Modelos para Predição via Validação Cruzada: Uma Aplicação em Avaliação de Imóveis, que também faz uma análise do modelo utilizado na Dissertação do Engenheiro Rubens Dantas.

Referido artigo propõe como critério para escolha entre possíveis modelos concorrentes, entre eles os modelos de regressão Gama e Log-Normal, o uso de Validação Cruzada ao invés de medidas de ajuste, como a Deviance.

Finalmente registre-se que a Norma Brasileira de Avaliações de Imóveis que só reconhecia como ferramenta capaz de suportar uma avaliação rigorosa no Método Comparativo Direto a Análise de Regressão, visto que normalizava seus testes de tal forma que, pelo menos aparentemente, induzia á escolha da ferramenta, já admite o emprego de outras ferramentas; destacando a atuação do Engenheiro de Avaliações, que deverá fundamentar consistentemente seus cálculos.

Uma vez que uma rede neural treinada fornece valores que podem ser comparados com os originais, pode-se perfeitamente analisar sua robustez pela Validação Cruzada, bem como analisar a distribuição dos resíduos para construção do intervalo de confiança para previsão, atendendo os requisitos para um trabalho consistente, que atenda a Norma de Avaliação de Bens, recém aprovada.

Espera-se que referido trabalho estimule novos companheiros na busca dos conhecimentos necessários para bem empregar a ferramenta hoje disponível.

6. REFERÊNCIAS BIBLIOGRÁFICAS

ABNT (Associação Brasileira de Normas Técnicas). Norma NB-502 — Avaliações de Imóveis Urbanos. Rio de Janeiro. ABNT, 1989.

Barbosa, Emanuel Pimentel e Bidurin, Claudio P. – Seleção de Modelos de Regressão para Predição via Validação Cruzada: Uma Aplicação em Avaliação de Imóveis – Revista Brasileira de Estatística, 52 (Nºs 197/198) pág. 105-120 – Jan/Dez-1991.

Dantas, Rubens Alves. – Avaliação de Glebas Inseridas na Malha Urbana – Dissertação de Mestrada – Departamento de Engenharia Civil - UFPE, 1988.

Drang, Diane E. , Edelson, Barry e Levine, Robert I. - Inteligência Artificial e Sistemas Especialistas, tradução de Maria Claudia Santos Ribeiro, São Paulo, McGraw-Hill, 1988.

Garza, Jesus M., Rouhana, Khalil, Neural Networks Versus Parameter-Based Applications in Cost Estimating, Cost Engineering, vol. 37, fevereiro 19995, p. 15-17.

Guedes, Jakson C. , Avaliação de Bens Utilizando Metodologia Científica - Tese de Mestrado - COPPE/UFRJ, Rio de Janeiro, 1992.

Harman, Paul - Expert Systems tools and applications, New York, John Willey, 1988.

Hoffman, Rodolfo e Vieira, Sônia, Análise de Regressão: Uma Introdução à Econometria, São Paulo, Hucitec, 1977.

Lawrence, Jeannette, Introduction to Neural Networks – Design, Theory, and Applications – California Scientific Software Press, 6th edition, 1994.

Nelder, I.A. and Wedderburn, R.W.M., Generalized Linear Models. JRSS A 135. P. 370-384.

Neter, John; Wasserman, William and Kutner, Michael H., Applied Linear Statistical Models, Boston, Irwin, 1990.

APÊNDICE 1 - BANCO DE DADOS

ITEM	TEST	PROF	TRANS	MES	NURB	VUNIT	BAIRRO
1	19	30	0	31	6	2983.00	Iputinga
2	17	29	1	22	7	2695.00	Iputinga
3	36	29	0	32	6	3831.00	Iputinga
4	10	20	0	26	4	2250.00	Iputinga
5	12	36	1	25	6	3588.00	Torre
6	11	31	1	20	7	1261.00	Torre
7	10	30	1	14	8	1587.00	Torre
8	15	20	0	31	4	2667.00	Torre
9	10	30	0	31	4	3333.00	Torre
10	13	36	0	31	6	4273.00	Torre
11	16	34	0	52	7	29994.00	Torre
12	8	53	1	27	8	4288.00	C.Fortesan
13	24	39	0	31	6	7478.00	C.Fortesan
14	15	82	0	31	8	10339.00	C.Fortesan
15	36	39	1	3	6	2422.00	C.Fortesan
16	17	135	1	32	8	6734.00	C.Fortesan
17	23	22	1	39	7	4018.73	C.Fortesan
18	40	233	1	50	8	16094.00	C.Fortesan
19	14	84	1	42	8	9267.00	C.Fortesan
20	16	33	0	54	5	32567.00	C.Fortesan
21	17	28	1	45	6	9918.00	C.Fortesan
22	17	30	1	29	8	7843.00	C.Fortesan
23	12	247	1	26	8	7093.00	C.Fortesan
24	18	21	0	55	5	29790.00	C.Fortesan
25	25	37	0	55	5	32258.00	C.Fortesan
26	24	30	0	44	5	4047.00	Iputinga
27	14	26	1	29	5	37.10	Iputinga
28	12	40	0	44	5	6250.00	Iputinga
29	13	30	0	44	8	10970.00	Iputinga
30	14	28	0	43	4	5639.00	Iputinga
31	13	41	0	43	7	9259.00	Iputinga
32	10	25	0	44	7	10737.00	Iputinga
33	10	22	0	43	4	4067.00	Iputinga
34	15	32	0	43	5	5208.00	Iputinga
35	15	30	0	32	5	3333.00	Iputinga
36	15	64	0	32	5	2083.00	Iputinga
37	66	29	0	3	5	2712.00	Iputinga
38	15	25	0	31	5	5333.00	Iputinga
39	14	24	1	23	7	3535.00	Iputinga
40	25	50	1	20	7	2306.00	Iputinga
41	19	30	1	44	7	13789.00	Iputinga
42	10	23	1	43	6	10714.00	Iputinga
43	9	30	0	57	8	30476.00	Torre
44	14	30	0	57	3	28571.00	Torre
45	14	71	0	57	7	35211.00	Torre
46	10	25	0	57	4	14846.00	Torre
47	15	67	0	57	6	15344.00	Torre
48	15	25	1	49	7	8040.00	Torre
49	18	23	0	54	5	24154.00	C.Fortesan
50	14	30	0	30	5	3333.00	Iputinga

APÊNDICE B – COMPARAÇÃO DE RESULTADOS

ITEM	OBSERVADO	GAMA	NEURAL	MELHOR
1	2.984	4.379	4.588	MGL
2	2.695	2.530	3.974	NEURAL
3	3.831	5.972	3.817	MGL
4	2.251	2.292	3.742	NEURAL
5	3.587	3.049	2.930	MGL
6	1.261	2.537	3.146	MGL
7	1.588	1.988	2.615	MGL
8	2.668	4.050	3.212	MGL
9	3.334	3.742	3.096	NEURAL
10	4.273	4.916	3.958	MGL
11	30.031	19.330	28.599	MGL
12	4.290	5.303	5.980	NEURAL
13	7.480	7.492	6.610	NEURAL
14	10.342	8.085	11.450	MGL
15	2.421	1.590	3.784	NEURAL
16	6.735	7.519	3.071	MGL
17	4.020	11.600	4.538	MGL
18	16.091	28.750	15.106	NEURAL
19	9.265	13.310	10.141	NEURAL
20	32.533	31.570	31.252	NEURAL
21	9.917	13.510	10.273	NEURAL
22	7.840	6.775	6.171	NEURAL
23	7.094	4.740	5.541	MGL
24	29.733	24.130	28.135	NEURAL
25	32.209	26.950	33.025	NEURAL
26	4.048	8.870	5.690	NEURAL
27	3.711	2.875	3.461	NEURAL
28	6.248	7.386	6.942	NEURAL
29	10.971	10.670	11.757	MGL
30	5.636	6.437	4.323	MGL
31	9.256	8.918	9.130	MGL
32	10.732	9.109	9.229	MGL
33	4.068	5.995	4.604	NEURAL
34	5.208	7.321	6.060	NEURAL
35	3.334	3.893	3.892	NEURAL
36	2.084	3.826	4.439	MGL
37	2.711	2.088	3.767	MGL
38	5.335	3.685	3.858	NEURAL
39	3.533	2.569	2.648	NEURAL
40	2.305	2.474	2.789	NEURAL
41	13.794	9.092	12.420	MGL
42	10.711	6.797	11.691	NEURAL
43	30.333	26.130	30.970	NEURAL
44	28.567	15.770	28.757	NEURAL
45	35.242	24.540	33.655	NEURAL
46	14.839	16.480	13.813	NEURAL
47	15.337	22.240	15.048	MGL
48	8.039	14.520	7.820	NEURAL
49	24.101	23.090	25.110	NEURAL
50	3.334	3.419	3.858	MGL

CURRICULUM VITAE

47 anos, Engenheiro Civil graduado pela Universidade Federal do Rio de Janeiro (1979) e Mestre em Ciências em Engenharia de Produção na área de Projetos Industriais pela COPPE/UFRJ, tendo defendido Tese de Mestrado em maio/1992 com o título "Avaliação de Bens Utilizando Metodologia Científica", e atualmente cursando o Doutorado em Planejamento Energético no Programa de Planejamento Energético da COPPE/UFRJ.

Profissional com 21 anos na área de Avaliações Técnicas de Bens e Estudos Econômicos.

DESTAQUES NA ATUAÇÃO PROFISSIONAL

Empregado da **PETROBRÁS S.A. (1987/2001),** estando atuando, desde 1987, no Setor de Engenharia de Perícias e Avaliações (SEPAV) da unidade de Engenharia, na elaboração de avaliação técnica de bens (setor imobiliário urbano/rural e setor industrial), e a partir do corrente ano exercendo a função de **Consultor Técnico de Avaliação de Mercado e Econômicas**.

Jackson Carvalho Guedes
Engª Civil, M. Sc. Engenharia de Produção
CREA-RJ-45.428-D
e-mail : jacksonguedes@petrobras.com.br