Comparação entre Regressão Logística e Rede Neural (MLP) para análise de risco de crédito

Leonardo Souza Campos, Tércio Borges Ribeiro e Uemerson Pinheiro Junior
Universidade Estadual Paulista "Júlio de Mesquita Filho" - UNESP

João Paulo Papa
UNESP, Av. Eng. Luís Edmundo Carrijo Coube, 14-01 - Jardim Marabá, Bauru - SP

Ferramentas de machine learning têm ganho cada vez mais relevância devido a sua grande flexibilidade e precisão.
Este trabalho utilizará duas ferramentas bem difundidas de machine learning, a regressão logística e as redes neurais artificiais tipo MLP, para a criação de dois modelos de classificação de risco de crédito bancário a partir do banco de dados *Credit approval data set*, do repositório UCI . O trabalho incluirá as etapas de tratamento de dados, com variáveis contínuas e atributivas, análise de sensibilidade das arquiteturas de rede neural e por fim analisará a precisão de cada modelo quanto à performance de classificação.

***Palavras-chaves* _ Regressão logística, redes neurais artificiais, MLP, perceptron, crédito bancário**

I. Introdução

Desde a última década, aplicações com redes neurais e aprendizado de máquina despertaram grande interesse nos meios acadêmico e industrial. Grande parte desse interesse deve-se ao desempenho elevado obtido pelas redes neurais com aprendizado supervisionado nas tarefas de classificação e regressão, tais como na classificação de imagens para carros autônomos [2], reconhecimento de fala para assistentes virtuais, predição de preço de ações e detecção de falhas em linhas de produção.

Entre essas aplicações, àquelas relacionadas ao âmbito financeiro tem se tornado cada vez mais frequentes, sendo que essas ferramentas são utilizadas no balancete financeiro, auditoria fiscal, concessões de crédito, empréstimos e financiamentos. Elas são essenciais por possuírem capacidade de prever com boa precisão o potencial de retorno financeiro, inadimplência do consumidor, retorno de investimento, entre outros, mesmo lidando com um volume grande de variáveis de entrada, dos mais variados tipos.

O objetivo deste trabalho é utilizar dois tipos bem difundidos de algoritmos de *machine learning* para criar dois modelos de correlação a partir de um banco de dados relacionado à aprovação de crédito bancário, utilizando-se a regressão logística e redes neurais artificiais, do tipo MLP.

II. REFERENCIAL TEÓRICO

2.1. Perceptron

Em 1958 o psicólogo americano Frank Rosenblatt foi o responsável pela criação do Perceptron: uma aproximação de um neurônio biológico a partir da reavaliação de algumas restrições do modelo neuronal de McCulloch e Pitts. As restrições removidas envolviam os valores dos pesos sinápticos (posteriormente restritos ao valor unitário) que agora poderiam assumir valores entre -1 e 1 possibilitando alterar a influência das entradas [12]. A Equação (Fig 1) exibe a saída produzida pelo modelo de McCulloch e Pitts [10] é válido ressaltar que o valor do peso sináptico que multiplica a entrada xk é sempre unitário. A partir da habilitação i o neurônio irá ativar (saída = 1) caso a soma das entradas seja maior que um limiar de ativação Θ.

$$a(x) = \begin{cases} 1 & : \sum_{k=1}^{N} x_k > \Theta \wedge i = 1 \\ 0 & : \text{caso contrário} \end{cases}$$

Fig. 1. Modelo de McCulloch e Pitts

Com as modificações propostas por Rossenblatt a equação da saída produzida comporta-se como definido em (Fig. 2) [12], os pesos são agora definidos no vetor ~ω e o inverso do valor que antes representava o limiar necessário para ativação do neurônio é utilizado como peso considerando uma entrada unitária. Esse valor é comumente denominado de bias na literatura. A habilitação do neurônio também foi removida no *perceptron* de Rosenblatt.

$$a(x) = \begin{cases} 1 & : \sum_{k=1}^{N} \omega_k \cdot x_k + b > 0 \\ 0 & : \text{caso contrário} \end{cases}$$

Fig. 2. Modelo Neurônio Perceptron

A Fig. 2 ilustra a estrutura do neurônio perceptron. O modelo é composto por N + 1 entradas onde o vetor de entrada x tem tamanho N, porém existe uma entrada extra com valor unitário e o valor do bias no lugar do peso. O somatório acumula as multiplicações entre pesos e entradas gerando na saída zk, o peso ponderado do neurônio. O bloco seguinte corresponde a função de ativação, responsável por produzir a saída do neurônio.

2.2. Multilayer Perceptron

A organização de vários neurônios *perceptron* em camadas distintas ficou conhecida na literatura como a arquitetura de redes *Multilayer Perceptron* (MLP) [12]. O uso de redes de neurônios com mais de uma camada visava resolver os problemas encontrados na classificação de dados não linearmente separáveis como a emulação de uma porta ou exclusiva (XOR) apresentado em [8].

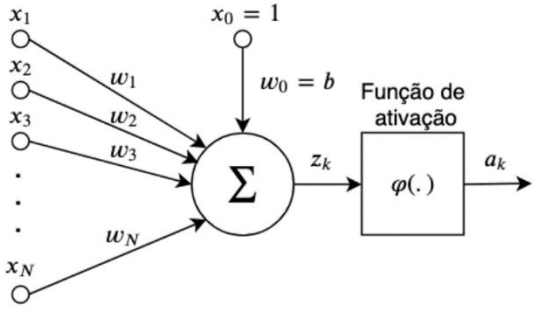

Fig.3. Modelo Função de Ativação

Redes com essa arquitetura aliada ao treinamento com o algoritmo *backpropagation* são capazes de classificar corretamente dados não linearmente separáveis [18]. A equação das ativações e pesos ponderados de uma rede MLP é a seguinte [20]:

$$z_j^l = \sum_{k=1}^{N} \left(\omega_{jk}^l \cdot a_k^{l-1} \right) + b_j^l$$

$$a_j^l = \varphi(z_j^l)$$

Fig.4. Modelo Função de Ativação

Onde N é o número de entradas do neurônio, ω l jk representa o peso vindo do neurônio k da camada l − 1 para o neurônio j da camada l, e a l k , b l j e z l k correspondem respectivamente a ativação, bias e peso ponderado do neurônio k da camada l.

2.3. Regressão Logística

Um modelo de regressão pode ser definido como uma equação matemática em que se expressa o relacionamento de variáveis. Nestes modelos, define-se uma variável dependente (Y), ou variável de saída, e procura-se verificar a influência de uma ou mais variáveis ditas variáveis independentes, causais ou explicativas (X's) sobre esta variável dependente. Na equação (1) a seguir, vê-se um exemplo de um modelo de regressão linear.

$$Yi = \beta 0 + \beta 1X1 + \beta 2X2 + ... + \beta iXi + \varepsilon i \quad (1)$$

Onde:
Yi: representa a variável dependente;
βi : são os coeficientes de regressão;
Xi : são as variáveis independentes;
εi: erro aleatório 2 .

A regressão logística consiste em um tipo de regressão aplicável e preferida quando se tem uma variável dependente categórica dicotômica, ou seja, uma variável nominal ou não métrica que possui apenas dois grupos ou classificações como resultados possíveis como, por exemplo, alto ou baixo, homem e mulher, sim ou não etc.
Na regressão logística, a probabilidade de ocorrência de um evento pode ser estimada diretamente. No caso da variável dependente Y assumir apenas dois possíveis estados (1 ou 0) e haver um conjunto de p variáveis independentes X1 , X2 , ... , Xp, o modelo de regressão logística pode ser escrito da seguinte forma:

$$P(Y=1) = \frac{1}{1+e^{-g(x)}}$$

Fig.5. Modelo de Regressão Logística

Onde: g(x)= β0 + β1X1i + ... + βpXpi
Considerando certa combinação de coeficientes β0, β1, ..., βp e variando os valores de X, observa-se que a curva logística tem comportamento probabilístico no formato da letra S, o que é característica da regressão logística. Esse formato dá à regressão logística alto grau de generalidade, aliada a aspectos muito desejáveis:
a) Quando g(x) → + ∞ , então P (Y = 1) → 1;
b) Quando g(x) → − ∞ , então P (Y = 1) → 0.

Assim, como se pode estimar diretamente a probabilidade de ocorrência de um evento, pode-se estimar a probabilidade de não ocorrência por diferença: P (Y = 0) = 1 - P (Y = 1). Ao se utilizar a regressão logística, a principal suposição é a de que o logaritmo da razão entre as probabilidades de ocorrência e não ocorrência do evento é linear:

$$\frac{P(Y=1)}{P(Y=0)} = e^{\beta_0+\beta_1 X_i+\beta_2 X_i+...+\beta_p X_{pi}}$$

Fig.6. Probabilidades de Ocorrência e não ocorrência do evento

$$\ln\left[\frac{P(Y=1)}{P(Y=0)}\right] = \beta_0 + \beta_1 X_i + \beta_2 X_i + ... + \beta_p X_{pi}$$

Fig.7. Probabilidades de Ocorrência e não ocorrência do evento

Por essa razão, ao interpretar os coeficientes da regressão logística, optou-se pela interpretação de e β e não diretamente de β. Para utilizar o modelo de regressão logística para discriminação de dois grupos, a regra de classificação é a seguinte: *Se P(Y=1) > 0,5 então classifica-se Y=1; * Em caso contrário, classifica-se Y=0.
Fazendo uma síntese, pode-se dizer que um modelo de regressão logística prevê a probabilidade direta de um evento ocorrer. Como se sabe, a probabilidade deve ser um valor limitado entre 0 (zero) e 1 (um) de forma que, se o valor previsto estiver acima de 0,5, aceita-se a hipótese atribuída ao número 1. Do contrário, aceita-se a atribuição dada ao valor 0, qual seja sim ou não, alta ou baixa etc. Esta relação limitada por 0 e 1 caracteriza uma relação não linear que pode ser representada graficamente por uma curva em forma de sigmóide.

III. METODOLOGIA

3.1. Tratamento dos dados

Para a execução deste trabalho, foi selecionado o banco de dados "Credit approval data set", do repositório UCI.
Este banco de dados diz respeito a autorizações de liberação de cartão de crédito, tendo como variável de resposta uma classificação binária (0,1). Neste banco de dados, cada resposta está relacionada a 15 X 's independentes, com valores e escalas confidenciais.
Um fator que motivou a escolha deste banco de dados em específico foi que ele apresenta tanto variáveis contínuas quanto variáveis atributivas. Entre as 15 variáveis, 6 são

variáveis contínuas, 4 são classificações binárias e as demais 5 são classificações multiclasse.
O banco de dados é composto de 690 linhas de entrada, sendo que em 37 delas, 5% do total, há informações faltantes.

A fim de propiciar a utilização dos algoritmos de aprendizado de máquina no banco de dados em questão, algumas etapas de tratamento dos dados foram aplicadas, sendo elas a etapa de separação das variáveis multiclasse em múltiplas colunas, remoção de dados ausentes, normalização dos dados, e separação da população de treino e teste.

O grupo optou por remover as linhas que apresentavam dados ausentes, uma vez que estas representam apenas 5% do total. Sendo assim, 654 dados de entrada foram utilizados.
A normalização dos dados é uma etapa fundamental para garantir o correto funcionamento das funções de ativação da rede neural, o que foi executado utilizando-se a função *sklearn.preprocessing.StandardScaler.*
Do total de 654 dados de entrada, 70% foram utilizados para treino enquanto 30% para validação da rede. Foi utilizada a função *sklearn.model_selection_train_test split* visando garantir o balanceamento das variáveis de saída entre as duas classes.

3.2. Arquitetura Regressão Logística

A biblioteca scikit-learn foi utilizada para realizar a regressão logística, sendo utilizado as seguintes instâncias da biblioteca: *sklearn.linear_model.LogisticRegression*.
Durante a execução do algoritmo ocorreu a convergência do algoritmo, que acontece quando o erro da resolução fica variando dentro de uma faixa muito pequena, quase não mudando e as diferenças entre os erros por interação são maiores do que tolerância definida, desta forma o algoritmo não converge. Assim, foi necessário aumentar o número de iterações do algoritmo por meio do parâmetro *max iter* da função *Logistic Regression* para o algoritmo convergir.

3.3. Arquitetura rede neural

A arquitetura de rede neural implementada neste problema foi uma MLP convencional. Foram utilizadas duas camadas ocultas de neurônios, além da camada de saída. Cada camada conta com um total de 29 neurônios, representando aproximadamente dois terços da quantidade de variáveis de entrada, de 43. Este valor foi adotado, tendo como base a literatura de referência. Em cada neurônio pertencente às camadas ocultas, foi utilizada a função retificador linear, também conhecida como “relu”.

A camada de saída possui um neurônio, uma vez que a saída deste problema é binária. Em função também da variável saída binária, a função de ativação selecionada para a camada de saída foi a função sigmóide.

Visando melhorar a performance de treinamento da rede neural, a técnica de *dropout* foi empregada. Esta técnica consiste em utilizar, durante a etapa de *forward propagation* da rede, apenas uma fração dos neurônios pertencentes a determinada camada para o treinamento da camada posterior. Para a etapa de *back propagation* do treinamento, todos os neurônios são utilizados. Este procedimento possui reconhecidamente o potencial de reduzir o viés da rede, evitando o *overfitting* e gerando melhor performance de classificação dos dados de teste, tanto em precisão quanto recall.

A fim também de evitar o overfitting dos dados, foi usada a técnica de *early stopping*, cujo objetivo é interromper o treinamento quando determinada variável monitorada deixa de apresentar uma melhora em função de um número determinado de época de treinamento. O erro dos dados de treino foi utilizado para a validação do modelo, e o treinamento interrompido quando a função erro, *loss function*, deixou de reduzir 1e-3 em 5 épocas de treinamento.

A biblioteca *Keras* foi utilizada para criação da rede, sendo que as instâncias usadas foram *keras.models.Sequential, keras.layers.Dense, keras.layers.Dropout e keras.callbacks.EarlyStopping*.

A imagem abaixo mostra um diagrama da arquitetura de rede, incluindo as camadas ocultas com suas respectivas funções de ativação.

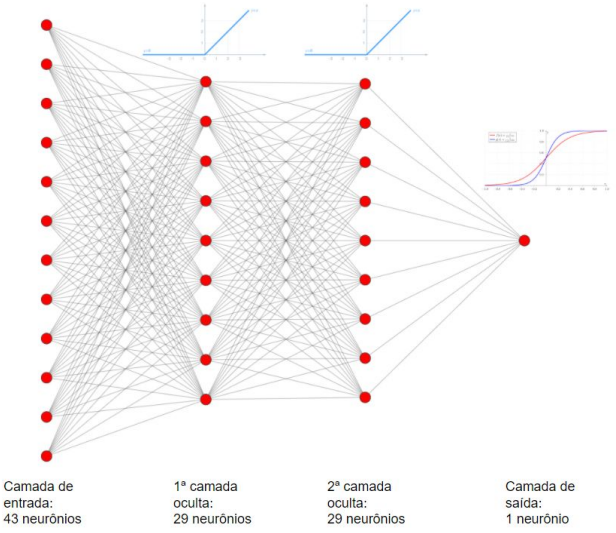

Fig.8. Diagrama da Arquitetura de Rede

IV. PROCEDIMENTO EXPERIMENTAL

4.1.Regressão Logística

Neste trabalho, para realizar o experimento com a regressão logística optou-se em separar os dados na proporção 70% para o conjunto de treinamento e 30% para realizar os testes. Executou-se cinco vezes o algoritmo para realizar uma média da performance de acerto das classificações realizadas, como é possível ver na tabela abaixo:

TABELA I. Acuracidade das réplicas de treinamento

	1ª execução	2ª execução	3ª execução	4ª execução	5ª execução	média
AC	0.87	0.88	0.86	0.85	0.83	0.86

Observa-se que a média da acurácia das execuções foi de 85.89%. Na tabela abaixo é apresentado a precisão que o algoritmo conseguiu alcançar em cada execução:

TABELA II. PRECISÃO DAS RÉPLICAS DE TREINAMENTO

	1ª execução	2ª execução	3ª execução	4ª execução	5ª execução	Média
1	0.85	0.88	0.88	0.80	0.77	0.84
0	0.89	0.89	0.83	0.90	0.89	0.88

A precisão é um termo relacionado ao quanto em porcentagem estão classificados corretamente entre cada classe, onde a classe 1 são os que não tiveram créditos aprovados e 0 os que tiveram créditos aprovados. Na tabela a seguir é apresentado o *recall*:

TABELA III. *RECALL* DAS RÉPLICAS DE TREINAMENTO

	1ª execução	2ª execução	3ª execução	4ª execução	5ª execução	Média
1	0.88	0.87	0.83	0.87	0.86	0.86
0	0.85	0.90	0.89	0.84	0.82	0.86

O *recall* refere-se à proporção de positivos que foram identificados corretamente, sendo o termo positivo como a classe que deseja prever. Na tabela abaixo é mostrado os *f-score* obtidos nas execuções do algoritmo:

TABELA IV. *F-SCORE* DAS RÉPLICAS DE TREINAMENTO.

	1ª execução	2ª execução	3ª execução	4ª execução	5ª execução	Média
1	0.87	0.87	0.86	0.83	0.81	0.85
0	0.87	0.89	0.86	0.87	0.85	0.87

O *f-score* informa o balanço entre precisão e recall obtidos pelo algoritmo.

4.2. Rede Neural MLP

O primeiro objetivo do experimento executado foi realizar um estudo de sensibilidade para entender a influência de dois fatores da arquitetura da rede na precisão do modelo treinado: A quantidade de camadas ocultas e a aplicação ou não de *dropout* entre as camadas.

A estrutura de experimento usada foi um fatorial completo de dois níveis, com as amplitudes de número de camadas variando de 1 a 2 camadas e o *dropout* variando de 0% a 30 %.

Foram executadas três réplicas com cada estrutura, tendo em vista que a separação dos dados de treino e teste é gerada de maneira randomizada a cada réplica. Desta maneira, utilizando réplicas, é possível estimar estatisticamente (média e desvio padrão) a real performance da rede frente àqueles dados.

Os resultados obtidos podem ser vistos abaixo.

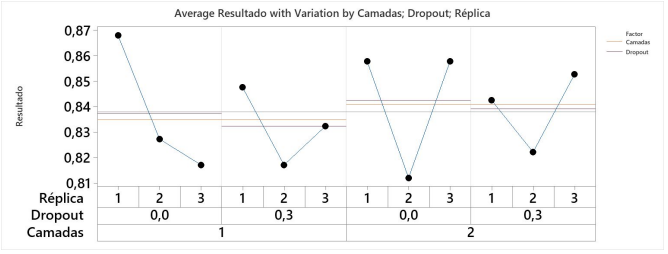

Fig. 9: Gráfico de variabilidade comparando os valores de acuracidade em cada tratamento.

Observa-se que nenhum dos dois fatores melhorou significativamente a performance do classificador, tendo em vista que a amplitude de variação da precisão entre réplicas de uma mesma arquitetura foi maior que o desvio de médias entre arquiteturas diferentes.

A configuração de rede com 1 camada oculta e sem *dropout*, por exemplo, apresentou média de precisão de 83,7%, com amplitude de 5%. A máxima amplitude de variação de precisão entre os quatro modelos testados foi de 1%.

Desta maneira, as quatro arquiteturas de rede estudadas apresentam performance de precisão em torno de 84 %. A partir do tratamento 1 foi gerada a matriz de confusão de uma rede específica, a partir da arquitetura com uma camada oculta e sem uso de *dropout*.

V. CONCLUSÃO

O trabalho realizado demonstra a capacidade e aplicabilidade das ferramentas de *machine learning* para a aplicação em questão, aprovação de crédito bancário.

Considerando-se as etapas de tratamentos de dados e treinamentos dos modelos, eles demonstraram precisão considerável em relação à aprovação de crédito.

Ambas os modelos foram capazes de lidar com o grande volume de dados de entrada e com uma quantidade significativa de variáveis independentes e atingir bons resultados de classificação.

O modelo de regressão logística apresentou precisão média de 86%, enquanto as redes neurais apresentaram precisão média de 83,7%.

Desta maneira, demonstra-se que as ferramentas são robustas, uma vez que seu uso possibilitaria a instituições financeiras concessão de crédito com uma boa previsibilidade de adimplência de seus clientes.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] A. M. Turing. Computing machinery and intelligence. Mind, 59(236):433–460, 1950.

[2] B. Widrow and M. A. Lehr. 30 years of adaptive neural networks: perceptron, madeline, and backpropagation. Proceedings of the IEEE, 78(9):1415–1442, Sep. 1990

[3] R. Lippmann. An introduction to computing with neural nets. IEEE ASSP Magazine, 4(2):4–22, Apr 1987.

[4] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133, Dec 1943.

[5] D.O. Hebb. The Organization of Behavior: A Neuropsychological Theory. A Wiley book in clinical psychology. Wiley, 1949.

[6] F. Rosenblatt. Perceptron simulation experiments. Proceedings of the IRE, 48(3):301–309, March 1960.

[7] IBM. 704 data processing system. IBM Archives.

[8] John C. Hay, Ben E. Lynch, and David R. Smith. Mark i perceptron operators' manual (project para). Report from Cornell Aeronautical Laboratory.

[9] Bernard Widrow. An adaptive "adaline"neuron using chemical "memistors". Report from Stanford Electronics Laboratories.

[10] Marvin Minsky and Seymour Papert. Perceptrons: An Introduction to Computational Geometry. MIT Press, Cambridge, MA, USA, 1969.

[11] P. J. Werbos. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. PhD thesis, Harvard University, 1974.

[12] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. Nature, 323:533–, October 1986.

[13] Janardan Misra and Indranil Saha. Artificial neural networks in hardware: A survey of two decades of progress. Neurocomputing, 74:239–255, 12 2010

[14] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. The MIT Press, 2016.

[15] Revista Ciências Exatas e Naturais, Vol.12 , no 2, Jul/Dez 2010 <https://core.ac.uk/download/pdf/230455828.pdf>

[16] Revista Contabilidade Vista & Revista, ISSN 0103-734X, Universidade Federal de Minas Gerais, 101 Belo Horizonte, v. 24, n. 4, p. 96-123, out./dez. 2013 <https://www.redalyc.org/pdf/1970/197033497006.pdf

CAMPOS, L.S., leonardo.souzacampos@hotmail.com, Tel. +55 16 99754-4292, RIBEIRO, T.B., terciobr05@hotmail.com, Tel. +55 34 99943-4377,JUNIOR. U.P., uemersonpinheirojunior@gmail.com Tel. +55 34 99887-5158; PAPA. J.P., papa@fc.unesp.br, Tel. +55 14 3103-6079