Eficácia de um Classificador Neural e a Importância Crítica da Normalização de Dados para Convergência

André Malmsteen Oliveira Amorim, Benjamim Isaac Ribeiro Lima, Diego Gabriel Silva Azevedo, Guilherme da Silva Pereira, Letícia Araújo, Manfred Lima Veiga

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Av. Rodrigo Octávio 6200, Coroado I – CEP 69080-900 – Manaus – AM – Brasil

{pereira.guilherme, andre.mlmst, benjamim.lima}@icomp.ufam.br

{diego.gabrielsa, leticia.araujo, manfred.veiga}@icomp.ufam.br

1. Introdução

Este trabalho apresenta a análise de desempenho de um modelo de Rede Neural Artificial (ANN) desenvolvido para a classificação binária de presença ou ausência de doença cardíaca, utilizando o conjunto de dados "Heart Disease" do repositório UCI Machine Learning [Dua and Graff 2019]. O pipeline metodológico compreendeu a limpeza e pré-processamento dos dados, resultando em 302 amostras únicas com um balanceamento de classes favorável (aprox. 54% / 46%), seguido pelo treinamento e avaliação de um classificador neural.

O foco desta análise recai sobre dois aspectos centrais: (1) a justificativa técnica da normalização de dados como uma etapa de pré-processamento indispensável para modelos baseados em gradiente, e (2) a avaliação rigorosa da eficácia e capacidade de generalização do modelo final.

2. Metodologia de Pré-processamento: A Importância da Normalização

A Análise exploratória de dados revelou que a normalização não era uma etapa opcional, mas um pré-requisito para a viabilidade do treinamento.

2.1. O Problema: Disparidade de Escalas dos Atributos

A análise estatística descritiva (via `.describe()`) dos 13 atributos clínicos (features) demonstrou uma vasta disparidade em suas escalas de magnitude. Por exemplo:

- **age** (idade): [29, 77]
- **chol** (colesterol): [126, 564]
- **oldpeak** (depressão ST): [0.0, 6.2]

2.2. Impacto em Redes Neurais Baseadas em Gradiente

O treinamento de ANNs é um problema de otimização onde os pesos sinápticos são ajustados iterativamente para minimizar uma função de custo (loss function). Otimizadores como o Adam (Adaptive Moment Estimation) [Kingma and Ba 2014], uma variação do Gradiente Descendente Estocástico (SGD), são sensíveis à escala dos dados de entrada.

- **Dominância de Atributos:** Sem normalização, atributos com magnitudes maiores (e.g., **chol**) dominariam a função de custo e, consequentemente, o cálculo do gradiente. Isso levaria a atualizações de peso desproporcionais, "ofuscando" a contribuição de atributos com escalas menores, mas potencialmente mais informativos (e.g., **oldpeak**) [LeCun et al. 2012].
- **Convergência Lenta/Instável:** A disparidade de escalas deforma a superfície de custo, criando vales estreitos (narrow ravines). Isso dificulta a convergência do otimizador, que pode oscilar erraticamente ou exigir uma taxa de aprendizado (*learning rate*) excessivamente pequena, tornando o treinamento computacionalmente custoso e instável [Ruder 2016].

2.3. Solução Aplicada: Padronização (Standardization)

A técnica de Padronização (ou z-score normalization) foi implementada usando o `StandardScaler` do Scikit-learn [Pedregosa et al. 2011]. Esta técnica transforma cada atributo x de forma independente, de modo que o conjunto de dados resultante tenha média $\mu = 0$ e desvio padrão $\sigma = 1$. A transformação é dada pela Equação 1:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Isso garante que todos os atributos contribuam de forma equânime para o processo de aprendizado, permitindo uma convergência mais rápida e estável.

2.4. Prevenção de Vazamento de Dados (Data Leakage)

Criticamente, a normalização foi aplicada de forma a evitar o vazamento de dados. O conjunto de dados foi primeiramente dividido em treino (241 amostras) e teste (61 amostras).

- O objeto `StandardScaler` foi ajustado (*fit*) **exclusivamente** aos dados de treino, calculando a média (μ_{treino}) e o desvio padrão (σ_{treino}) deste subconjunto.
- A transformação (*transform*) foi então aplicada tanto a `X_train` quanto a `X_test`, utilizando os parâmetros μ_{treino} e σ_{treino}.

Este procedimento simula o cenário real onde as estatísticas de dados futuros (teste) são desconhecidas, assegurando uma avaliação imparcial (unbiased) da capacidade de generalização do modelo e prevenindo o vazamento de informação do conjunto de teste para o treinamento [Kaufman et al. 2012].

3. Avaliação da Eficácia do Modelo

O modelo foi avaliado em sua capacidade de generalizar o aprendizado para instâncias não vistas (o conjunto de teste).

3.1. Arquitetura e Paradigma de Treinamento

A arquitetura da ANN foi definida como:

- **Camada de Entrada:** 13 neurônios (correspondendo às 13 features).
- **Camada Oculta 1:** 16 neurônios, ativação ReLU.
- **Camada Oculta 2:** 8 neurônios, ativação ReLU.

- **Camada de Saída:** 1 neurônio, ativação Sigmóide (ideal para classificação binária, resultando em uma probabilidade $p \in [0, 1]$).

Para combater o sobreajuste (overfitting), foram empregadas técnicas de regularização L2 (Weight Decay) [Krogh and Hertz 1991] e Dropout [Srivastava et al. 2014]. O treinamento foi otimizado com 'Adam' e monitorado pelo *callback* `EarlyStopping` [Prechelt 2012], observando a perda de validação (`val_loss`) com uma paciência de 11 épocas.

O treinamento foi configurado para 70 épocas, mas foi interrompido automaticamente na Época 31. Crucialmente, a política de `restore_best_weights` foi ativada, restaurando os pesos da Época 20, que apresentou o menor valor de `val_loss` (0.4260) e, portanto, o ponto de melhor generalização.

3.2. Resultados de Desempenho (Conjunto de Teste)

O modelo final (com os pesos da Época 20) alcançou as seguintes métricas no conjunto de teste (N=61):

- **Acurácia Final: 83.61%** (51 acertos / 61 amostras). Dado o balanceamento de classes do dataset, a acurácia é uma métrica significativa para este problema.

3.3. Diagnóstico de Overfitting

- Acurácia de Treino: 86.72%
- Acurácia de Validação (Teste): 83.61%
- **Gap de Generalização: 3.12%**

Uma diferença de apenas 3,12% entre o desempenho no treino e no teste é um indicador excelente de que o modelo não está sobreajustado. Isso sugere um bom *trade-off* entre viés (bias) e variância (variance) [Geman et al. 1992], e que as técnicas de regularização (L2, Dropout, EarlyStopping) foram altamente eficazes.

3.4. Análise de Métricas Clínicas (Matriz de Confusão)

Em contextos de diagnóstico clínico, a acurácia agregada é insuficiente, pois trata todos os erros com o mesmo peso. O erro mais crítico é o Falso Negativo (FN) ou Erro Tipo II, onde um paciente doente é diagnosticado como saudável, resultando em tratamento inadequado [Trevethan 2017].

A análise da Matriz de Confusão (Figura 1) e da Tabela 1 revelou a decomposição dos erros:

- Verdadeiros Positivos (TP): 28
- Verdadeiros Negativos (TN): 23
- Falsos Positivos (FP): 5 (Erro Tipo I)
- Falsos Negativos (FN): 5 (Erro Tipo II)

A métrica de maior relevância clínica é o Recall (Sensibilidade), que quantifica a capacidade do modelo de identificar os casos positivos reais, sendo fundamental em aplicações médicas onde falsos negativos têm consequências graves [Powers 2020, Trevethan 2017].

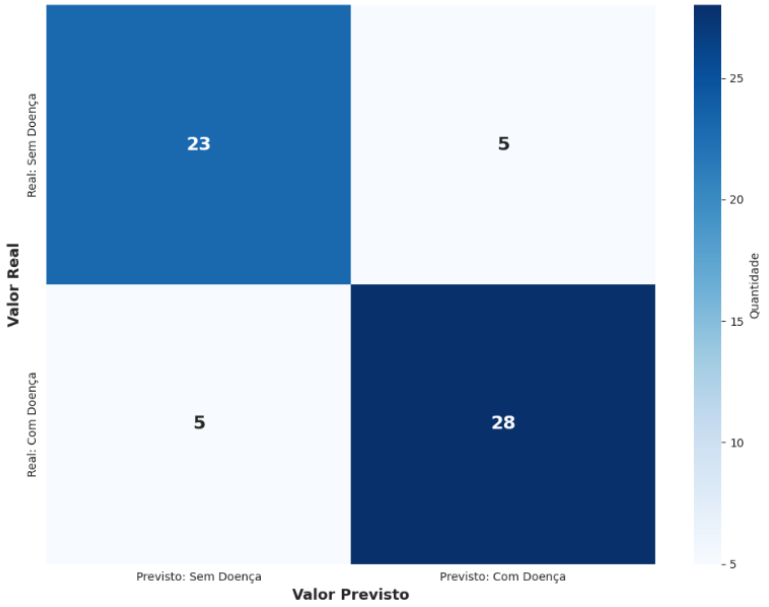

Figura 1. Matriz de Confusão dos resultados no conjunto de teste.

Tabela 1. Resumo das métricas de performance (com bordas).

Métrica	Resultado (Teste)	Interpretação
Acurácia	83.61%	O modelo acertou a classificação de 51 dos 61 pacientes.
Precisão (Classe 1)	84.85%	Das vezes que o modelo previu "Com Doença", ele acertou 84.85% das vezes.
Recall (Classe 1)	84.85%	O modelo identificou corretamente 84.85% de todos os pacientes que realmente tinham a doença.

- **Recall (Sensibilidade): 84.85%**. Calculado como $\frac{TP}{TP+FN} = \frac{28}{28+5}$. Isso significa que o modelo foi capaz de identificar corretamente quase 85% de todos os pacientes que realmente tinham a doença, falhando em 5 casos (FN).
- **Precisão (VPP): 84.85%**. Calculado como $\frac{TP}{TP+FP} = \frac{28}{28+5}$. Das vezes que o modelo previu "doença", ele estava correto em 85% dos casos. (Nota: A coincidência numérica com o Recall é uma particularidade deste resultado específico, pois $FN = FP$).

4. Conclusão

O modelo de Rede Neural Artificial proposto demonstrou ser altamente eficaz para a tarefa de classificação de doença cardíaca, alcançando uma acurácia de 83,6% e, mais importante, um Recall (métrica clinicamente relevante) de 84,85%.

Este desempenho é diretamente atribuído a duas decisões metodológicas chave: (1) A aplicação rigorosa da Padronização (z-score), que foi um pré-requisito não opcional para permitir a convergência estável do otimizador; e (2) O uso combinado de técnicas de regularização (L2, Dropout e Early Stopping), que mitigaram eficazmente o overfitting.

O resultado é um classificador robusto e com alta capacidade de generalização para dados não vistos.

Referências

Dua, D. and Graff, C. (2019). UCI machine learning repository. `http://archive.ics.uci.edu/ml`.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58.

Kaufman, S., Rosset, S., Perlich, C., and Stitelman, O. (2012). Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):1–21.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krogh, A. and Hertz, J. A. (1991). A simple weight decay can improve generalization. *Advances in Neural Information Processing Systems*, 4.

LeCun, Y. A., Bottou, L., Orr, G. B., and Müller, K.-R. (2012). Efficient backprop. *Neural Networks: Tricks of the Trade*, pages 9–48.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Powers, D. M. W. (2020). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.

Prechelt, L. (2012). Early stopping-but when? In Montavon, G., Orr, G. B., and Müller, K.-R., editors, *Neural Networks: Tricks of the Trade*, pages 53–67. Springer, 2nd edition.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Trevethan, R. (2017). Sensitivity, specificity, and predictive values: foundations, pliabilities, and pitfalls in research and practice. *Frontiers in Public Health*, 5:307.