Rede neural do tipo MLP para classificação de dados de câncer de mama

Victor Eduardo Requia

[1]Universidade do Estado de Santa Catarina – UDESC
Joinville – Brasil

victorrequia@gmail.com

Resumo. *Neste trabalho, será desenvolvido um modelo de rede neural do tipo Multilayer Perceptron (MLP) utilizando a biblioteca TensorFlow para definir a arquitetura da rede neural, com o objetivo de classificar dados para análise preditiva de câncer de mama, identificando casos como benignos ou malignos. O modelo será treinado usando um dataset do Breast Cancer Wisconsin, que requer limpeza e normalização devido à presença de dados faltantes. A arquitetura da rede será definida e ajustada em termos de camadas ocultas, neurônios e funções de ativação. O treinamento envolverá o uso do otimizador Adam e a avaliação do modelo incluirá análise de acurácia, assertividade e a identificação de underfitting ou overfitting.*

1. Introdução

Este trabalho concentra-se no desenvolvimento de um modelo de rede neural do tipo Multilayer Perceptron (MLP), utilizando a biblioteca TensorFlow, para a classificação de dados relacionados ao câncer de mama. O objetivo é criar um sistema capaz de diferenciar entre tumores benignos e malignos, uma distinção vital para o diagnóstico e tratamento adequados. Tradicionalmente, a classificação de tumores é realizada por especialistas, um processo que pode ser subjetivo e propenso a erros. A aplicação de redes neurais artificiais, especificamente MLPs, oferece uma abordagem mais objetiva e quantitativa, podendo potencializar a precisão de diagnósticos.

O dataset escolhido para este projeto é o Breast Cancer Wisconsin (Original). Este dataset apresenta desafios, incluindo a necessidade de limpeza e normalização de dados, que são etapas cruciais para garantir a eficácia do modelo de aprendizado de máquina.

Serão exploradas diferentes arquiteturas de rede, taxas de aprendizado e funções de ativação para determinar a configuração mais eficaz. Além disso, a avaliação do modelo incluirá não apenas a acurácia, mas também a assertividade, através de métricas como a curva ROC e a matriz de confusão, e a análise de underfitting ou overfitting.

2. Desenvolvimento

Nesta seção, será discutido os passos do projeto e as ações tomadas para o desenvolvimento dos experimentos.

O primeiro passo para desenvolver o projeto, foi analisar a tabela de dados e as variáveis. Foi concluído que, a maioria dos dados são números flutuantes e as variáveis, após uma pesquisa na área, dizem a respeito das características de núcleos celulares extraídos de imagens de biópsia de mama.

Algumas variáveis incluem, por exemplo:

- ID: Identificador único para cada amostra.
- Diagnóstico: O rótulo da classe, 'M' para maligno e 'B' para benigno.
- Características dos Núcleos Celulares: Incluem medidas como raio médio (radius_mean), textura média (texture_mean), perímetro médio (perimeter_mean), área média (area_mean), suavidade média (smoothness_mean), compactação média (compactness_mean), concavidade média (concavity_mean) e pontos côncavos médios (concave points_mean).

Após a análise da tabela de dados, foi necessário fazer o pré-processamento dos dados. Para o data set em questão, existem dados faltantes, que podem distorcer ou invalidar os resultados do modelo de aprendizado de máquina se não forem tratados adequadamente.

Para realizar a limpeza dos dados, foi feito os seguintes procedimentos:

- Identificação de Dados Faltantes: Foi utilizado o método isnull().mean() para identificar valores faltantes em cada coluna do DataFrame X.
- Remoção de Colunas com Dados Faltantes: Foi escolhido por remover colunas inteiras que contêm dados faltantes, utilizando X.dropna(axis=1, inplace=True).

A normalização é um passo fundamental para garantir que o modelo de rede neural não seja enviesado por variações na escala dos dados. Dados não normalizados podem levar a dificuldades no treinamento do modelo, especialmente em redes neurais, onde a escala dos inputs pode afetar o processo de aprendizado.

O processo de normalização no nosso código é realizado da seguinte forma:

- Aplicação da Função de Normalização: Foi utilizado a função normalize da biblioteca sklearn.preprocessing para normalizar os dados do DataFrame X.
- Criação de um Novo DataFrame Normalizado: Os dados normalizados são então convertidos de volta para um DataFrame do pandas, scaled_df = pd.DataFrame(normalize_df, columns=X.columns).

Essas etapas de limpeza e normalização são essenciais para garantir que o modelo de rede neural seja consistente, podendo afetar o desempenho e a precisão do modelo na classificação de tumores de mama como benignos ou malignos.

Com a base de dados limpa e os dados normalizados, foi feita a divisão dos dados em conjuntos de treinamento e testes. Para isso, foi utilizada a função train_test_split da biblioteca sklearn.model_selection. O test_size de 0.3, o que significa que 30% dos dados são reservados para teste e os 70% restantes para treinamento. Além disso, um random_state é definido para garantir a reprodutibilidade dos resultados.

Para o melhor teste, mostrado na Figura 1,2 e 3.A arquitetura do modelo, foi definida como do tipo sequencial com 3 camadas. A primeira camada é composta por 1 perceptron, a segundo por 3 e a teceria por 1 perceptron. O modelo usa o otimizador Adam com uma taxa de aprendizado de 0.03. O modelo é treinado usando o método fit do modelo. Passamos os conjuntos de dados de treinamento X_train e y_train para este método, juntamente com o número de épocas. No código, o número de épocas é definido como 50 conforme a descrição do trabalho. Durante o treinamento, o modelo utiliza os dados de treinamento para aprender, ajustando seus pesos para minimizar a função

de perda. Os valores de perda, são úteis para monitorar o progresso do treinamento e diagnosticar problemas como overfitting ou underfitting.

3. Resultados

O melhor resultado obtido no experimento, com os parâmetros descritos na Seção de Desenvolvimento, alcançou uma taxa de acurácia no treino de verdadeiro negativo de aproximadamente 82,04%, e 80,74 no verdadeiro positivo. Já no teste, alcançou uma taxa de verdadeiro negativo de 83,90% e de verdadeiro positivo de 80.18%. É importante notar que a curva de acurácia foi parecida nos treinos e nos testes, ressaltando a precisão do modelo sem a base de treinamento. É possível notar também que o padrão da curva ROC foi parecido para o modelo de teste e treino. Isso mostra que o modelo está generalizando bem. Ou seja, está mantendo um desempenho consistente, não apenas com os dados em que foi treinado, mas também com novos dados, indicando que não ocorreu overfitting nem underfitting.

Figura 1. Matriz de confusão: 50 épocas, 70% dos dados para treino, 3 camadas e 5 percéptrons.

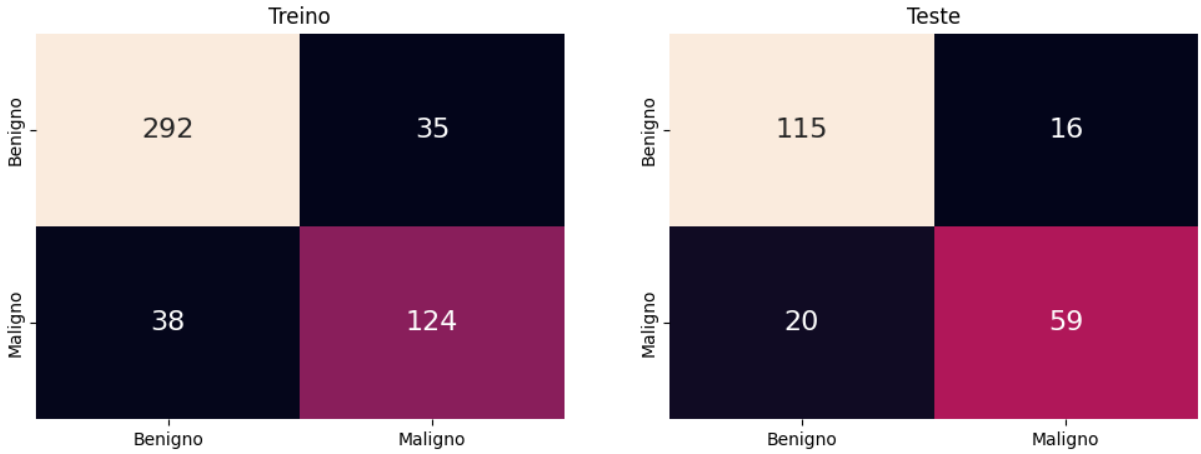


Figura 2. Precisão-Revocação: 50 épocas, 70% dos dados para treino, 3 camadas e 5 perceptrons.

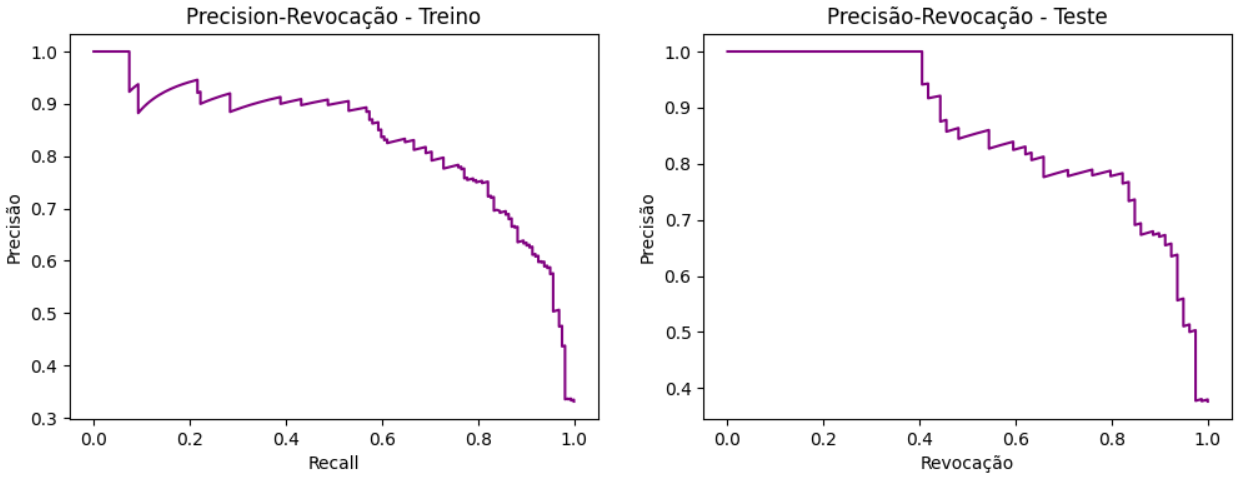
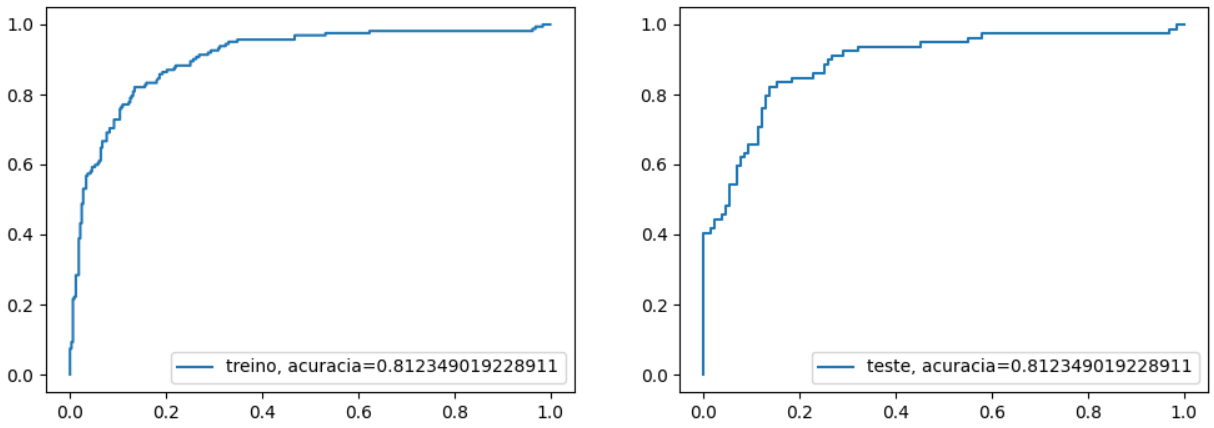


Figura 3. Curva ROC: 50 épocas, 70% dos dados para treino, 3 camadas e 5 perceptrons.



Um dos experimentos realizados, buscou mostrar que aumentar a quantidade de perceptrons e camadas, não necessariamente irá trazer um resultado melhor. As Figuras 4,5 e 6 mostram a mesma rede das figuras 1,2 e 3, porém com 6 camadas, sendo a primeira e última com 1 perceptron e as outras com 5. O resultado mostra que a acurácia diminuiu em relação ao teste anterior, ficando em treino 89% para verdadeiro negativo e 71% para verdadeiro positivo e em teste 87% para verdadeiro negativo e 72% para verdadeiro positivo.

Figura 4. Matriz de confusão: 50 épocas, 70% dos dados para treino, 6 camadas e 22 perceptrons.

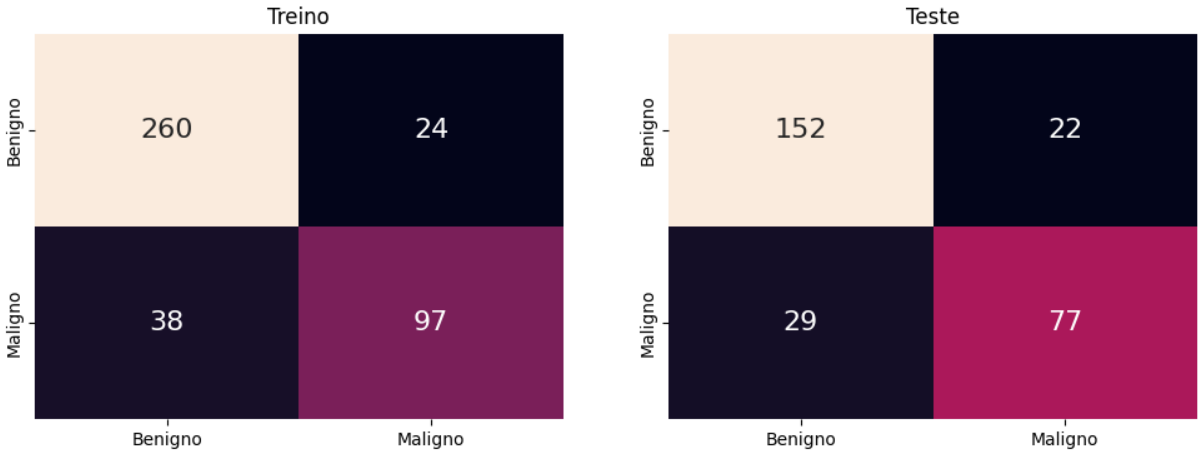


Figura 5. Precisão-Revocação: 50 épocas, 70% dos dados para treino, 6 camadas e 22 perceptrons.

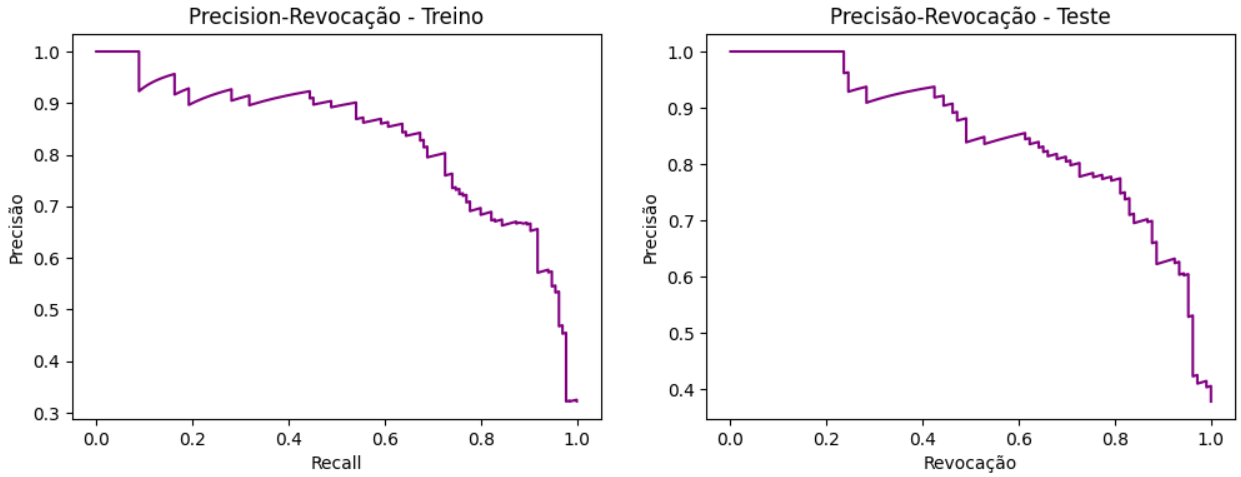
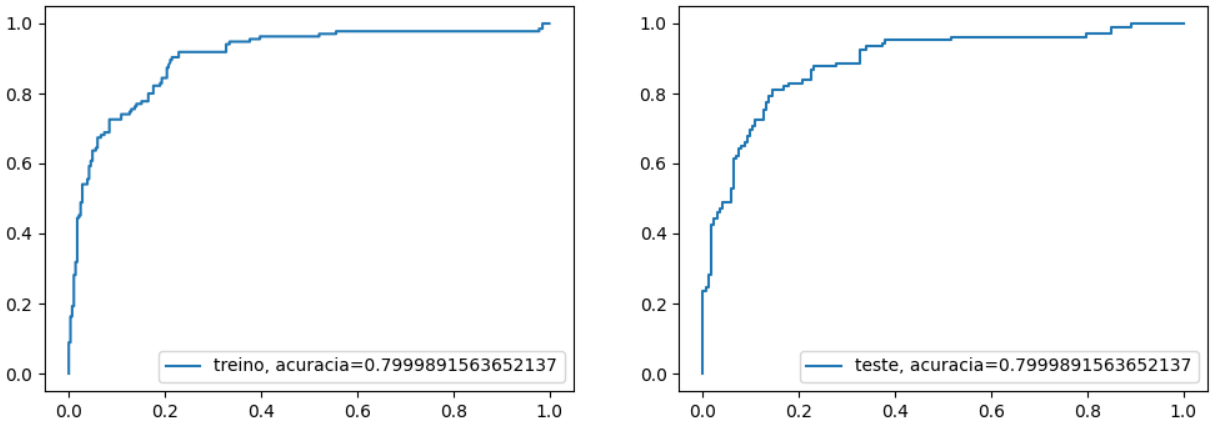


Figura 6. Curva ROC: 50 épocas, 70% dos dados para treino, 6 camadas e 22 perceptrons.



Outro teste realizado foi em relação à curva de aprendizado da rede neural. Foram utilizados os mesmos parâmetros da Figura 1,2 e 3, porém com a taxa de aprendizado de 0.7. Com isso, pode-se constatar com as figuras 7,8 e 9, que a taxa de aprendizado muito alta, afetar negativamente o desempenho da rede. A taxa de aprendizado é um dos parâmetros mais críticos em redes neurais, pois determina o tamanho dos passos que o algoritmo de otimização dá para alcançar o mínimo da função de perda.

Figura 7. Matriz de confusão: 50 épocas, 70% dos dados para treino, 3 camadas e 5 perceptrons.

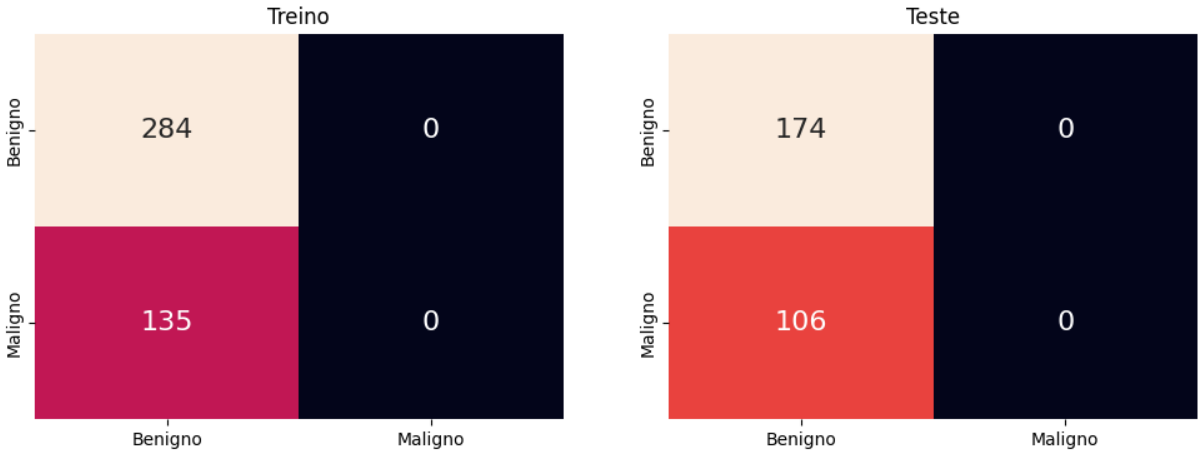


Figura 8. Precisão-Revocação: 50 épocas, 70% dos dados para treino, 3 camadas e 5 perceptrons.

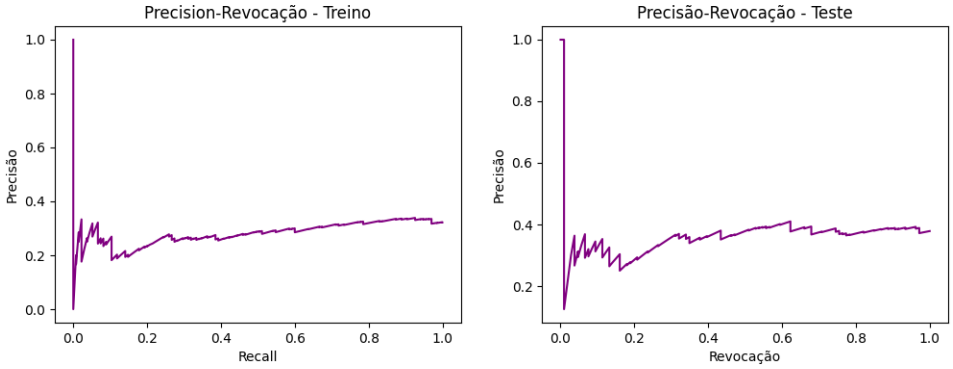
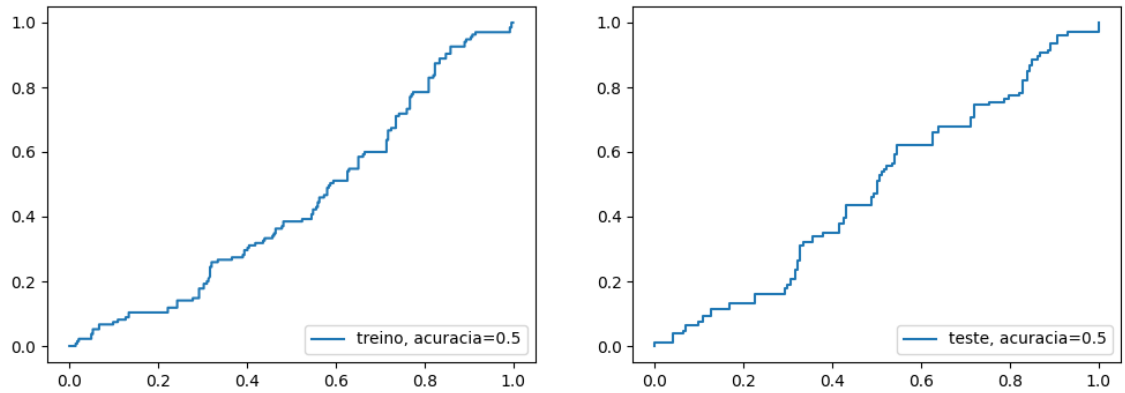


Figura 9. Curva ROC: 50 épocas, 70% dos dados para treino, 3 camadas e 5 perceptrons.



Também foi feito o teste com uma taxa muito baixa de dados para treinamento. Neste teste, os parâmetros utilizados foram os mesmos das Figuras 1,2 e 3, mudando somente a quantidade de dados para o treinamento da rede, deixando em 30% dos dados para treinamento. Os resultados podem ser vistos na figura 10,11 e 12. Foi possível observar que, apesar de poucos dados para treinamento (30%), mesmo assim teve uma boa acurácia de 75%, porém menor em comparação com 70% dos dados para treinamento.

Figura 10. Matriz de confusão: 50 épocas, 30% dos dados para treino, 3 camadas e 5 perceptrons.

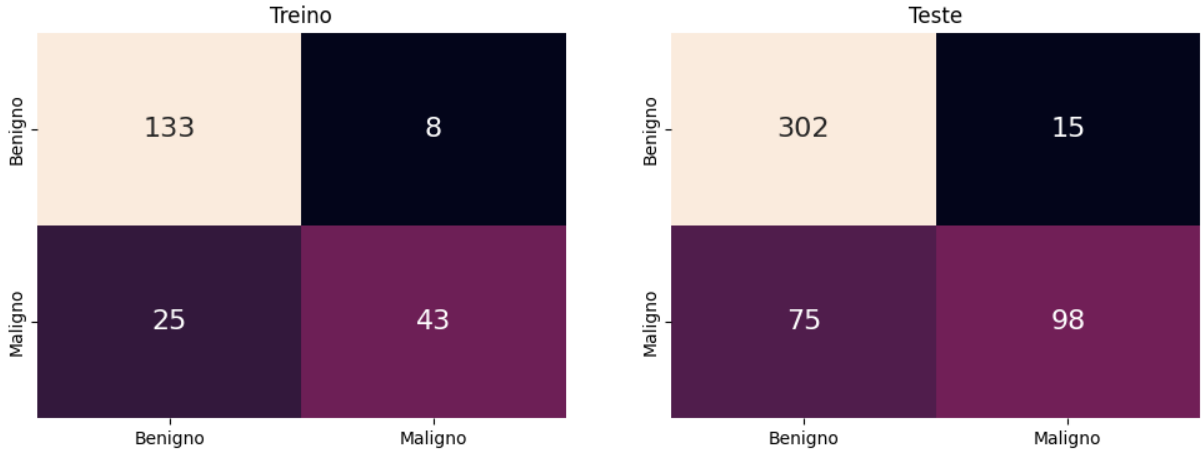


Figura 11. Precisão-Revocação: 50 épocas, 30% dos dados para treino, 3 camadas e 5 perceptrons.

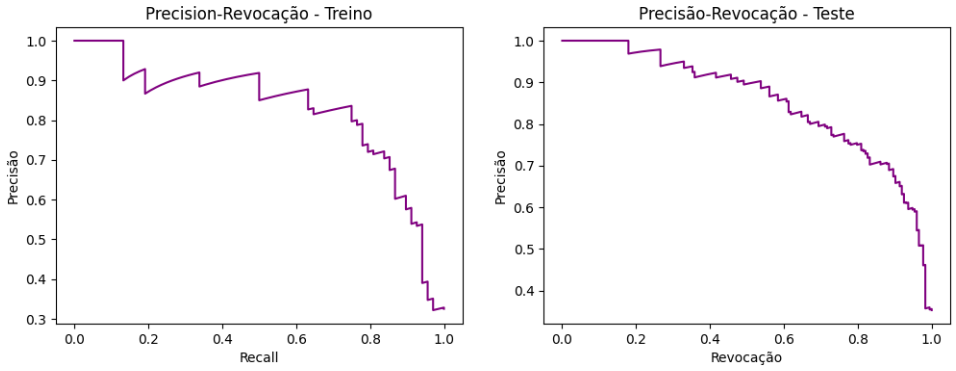
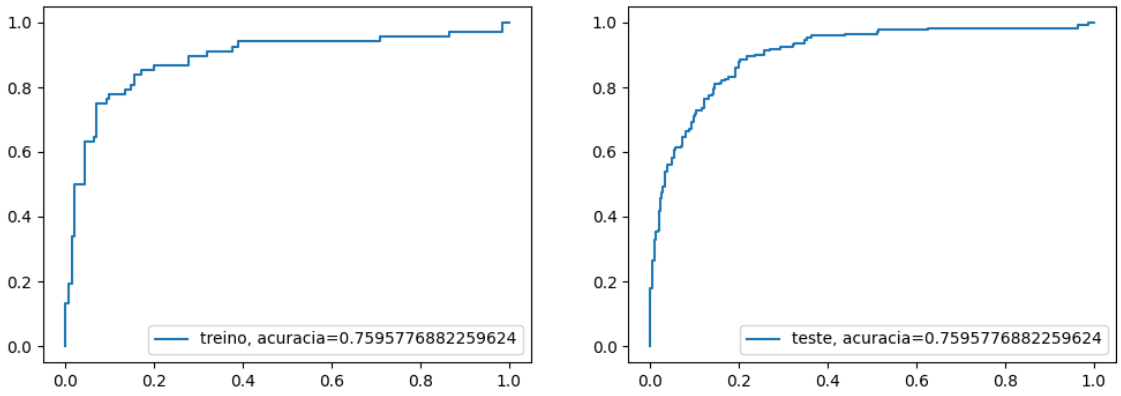


Figura 12. Curva ROC: 50 épocas, 30% dos dados para treino, 3 camadas e 5 perceptrons.



4. Conclusão

O modelo de rede neural desenvolvido, atendeu as expectativas e teve uma boa taxa de precisão não somente nos dados treinados, mas também nos dados que não fizeram parte do treinamento para fornecer diagnóstico de câncer de mama. Pode-se notar que o ajuste de parâmetros é muito sensível, podendo alterar consideravelmente o resultado alterando minimamente os valores dos parâmetros.

No melhor resultado, foi possível obter uma acurácia de 83,90% para verdadeiro negativo e 80.18% para verdadeiro positivo, utilizando 70% dos dados da base para treinamento. Essa acurácia mostra que os modelos de redes neurais para diagnóstico de câncer de mama, podem ajudar ao médica a conceder um diagnóstico para o paciente.

Foi possível concluir, que a quantidade de camadas e perceptrons devem ser bem ajustados e não necessariamente quanto mais camadas e mais perceptrons, melhor será o resultado. Também foi observado que a velocidade de aprendizado da durante o treinamento, pode afetar diretamente a eficácia dos resultados, uma curva de aprendizagem muito rápida pode acarretar vários problemas como falha de convergência, instabilidade, overfitting. Outro aspecto que se pode concluir com o trabalho, é que, aumentando muito a porcentagem da quantidade de dados de para treino, pode diminuir a eficácia da rede.