FUNDAMENTOS DE REDES NEURAIS PROFUNDAS

Palavras-Chave: Redes Neurais, Redes Profundas, Aprendizado de Máquina

Autores(as):
MIGUEL ROCHA MORENO, FEEC/UNICAMP
Prof. ROMIS ATTUX (orientador), FEEC/UNICAMP

INTRODUÇÃO:

O projeto que será apresentado tem como pano de fundo o avanço, nas últimas décadas, de um campo promissor da Inteligência Artificial (IA), que é o Aprendizado de Máquina (ML). Inicialmente concebido como uma especialidade da IA, o ML concentra-se em desenvolver máquinas e algoritmos capazes de aprender e exprimir resultados a partir de um conjunto de dados [Haykin, 2008]. O aprendizado profundo (DL), objeto de estudo deste projeto, é uma evolução do ML que se baseia em redes neurais e alcançou avanços significativos nos últimos anos [LeCun et al., 2015]. Esse progresso foi impulsionado pelo aumento da capacidade computacional, disponibilidade de grandes bases de dados e desenvolvimento de ferramentas de análise de dados.

Desse modo, o projeto como foi concebido está focado em redes neurais profundas convolucionais [Goodfellow et al., 2016]. O objetivo é estudar os conceitos essenciais e implementar uma rede complexa utilizando um (ou mais) conjunto(s) de dados específico(s). Essa iniciativa é motivada pelo crescimento acelerado da tecnologia, que demanda profissionais qualificados para trabalhar com redes neurais de ponta, especialmente em contextos de pesquisa e estudos de pós-graduação. A expectativa é que isso contribua para o desenvolvimento da área de IA no Brasil e gere resultados relevantes no futuro.

Inicialmente, as atividades foram divididas em três partes. As duas primeiras enfocaram conceitos e técnicas essenciais de aprendizado de máquina, como Regressão e problemas de Classificação. Na última parte, foi desenvolvida uma rede neural do tipo MLP (perceptron de múltiplas camadas) e uma rede profunda convolucional para lidar com problemas de Classificação.

METODOLOGIA:

Na primeira etapa do estudo, foram abordados os conceitos fundamentais de aprendizado de máquina (ML), com foco nos modelos básicos de regressão e classificação. O material baseou-se nas notas da disciplina de pós-graduação "IA048 - Aprendizado de Máquina" [Boccato e Attux, 2020]. Foram cobertos os paradigmas de aprendizado: supervisionado, não-supervisionado e por reforço, seguindo para os modelos básicos de Regressão e Classificação (como foco no problema binário).

Para o problema de Regressão Linear, a metodologia utilizada foi implementada em Python, fazendo uso de bibliotecas como Pandas, NumPy e Matplotlib para manipulação e visualização dos dados. A Regressão Linear foi aplicada a um conjunto de dados chamado "monthly-sunspots.csv" contendo informações sobre o número mensal médio de manchas solares [Boccato e Attux, 2020].

A Regressão Linear foi realizada em seis passos sequenciais. Primeiramente, os dados foram coletados e preparados para a análise, incluindo a divisão dos conjuntos de treinamento, validação e teste. Em seguida, foi implementada uma função para carregar os dados de treinamento, criando sequências de tamanho específico para as entradas e suas respectivas saídas esperadas.

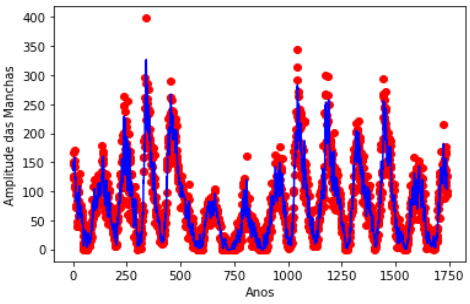

Figura 1 – Regressão das manchas solares em azul, comparado com os dados reais em vermelho – fonte: código próprio

No terceiro passo, foi realizada a implementação da Regressão Linear em si, calculando os pesos da combinação linear por meio do método dos mínimos quadrados. Esse processo possibilitou a criação de um modelo ajustado aos dados de treinamento.

Posteriormente, a Regressão Linear foi utilizada para realizar a predição das saídas em relação à matriz de design "x" e aos pesos obtidos. A métrica de erro médio quadrático (MSE) foi aplicada no quarto passo para avaliar a precisão do modelo, comparando as predições com os valores reais.

O quinto passo envolveu a avaliação de diferentes valores de K, onde o MSE foi calculado para diferentes tamanhos de sequências de dados. Essa análise permitiu compreender a influência do tamanho das sequências no desempenho do modelo.

Por fim, os resultados foram visualizados por meio de gráficos que mostravam as predições em relação aos valores reais. Além disso, um gráfico do MSE em função do valor de K foi plotado, auxiliando na análise dos resultados, exibidos respectivamente nas figuras 1 e 2.

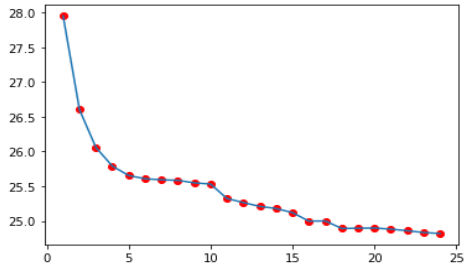

Figura 2 - Gráfico da métrica do Erro Quadrático Médio (MSE) obtido para o modelo de regressão - fonte: código próprio

A classificação binária foi conduzida por meio do algoritmo de Regressão Logística em oito passos sequenciais. Inicialmente, o conjunto de dados "dados_voz_genero.csv" foi carregado e dividido em atributos (X) e classes (y) [Boccato e Attux, 2020]. Para tornar os atributos numéricos, realizou-se um pré-processamento removendo pontos das colunas. Em seguida, o dataset foi separado em conjuntos de treinamento e validação usando a função "train_test_split" da biblioteca Scikit-learn, possibilitando o treinamento do modelo em parte dos dados e a avaliação do desempenho em dados não vistos.

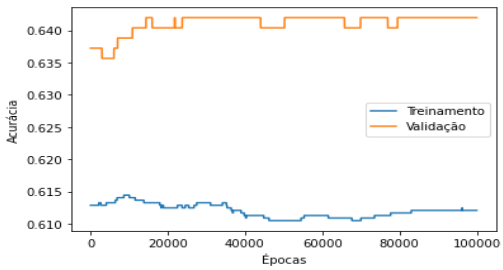

Figura 3 - Acurácia dos conjuntos de Treino e Validação - fonte: código próprio

Posteriormente, os atributos do conjunto de treinamento e validação foram convertidos para tipo numérico (float), e valores faltantes no conjunto de validação foram preenchidos com a média dos valores presentes no conjunto de treinamento. Adicionalmente, a normalização dos dados foi realizada para garantir que os atributos estivessem na mesma escala, contribuindo para a convergência do modelo.

Em seguida, a implementação da Regressão Logística prosseguiu, definindo funções de ativação "sigmoid" e "cost_function". O algoritmo de gradiente descendente foi empregado pela função "gradiente_descendente", permitindo que o modelo fosse ajustado aos dados de treinamento em direção ao gradiente descendente do erro, com o objetivo de otimizar a função de custo.

O modelo foi treinado após adicionar a coluna de 1's aos conjuntos de treinamento e validação para considerar o termo de interceptação na Regressão Logística. O vetor de parâmetros foi inicializado com zeros, e o algoritmo de gradiente descendente foi utilizado para ajustar iterativamente os parâmetros, otimizando a função de custo.

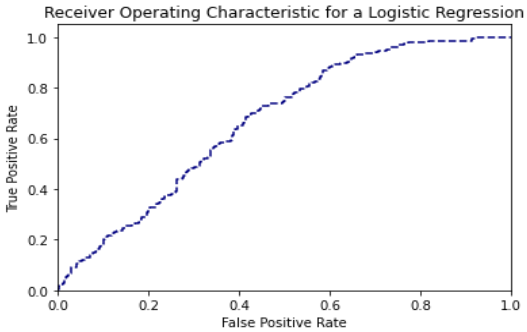

Figura 4 - Curva ROC do modelo criado - fonte: código próprio

Com o modelo treinado, os erros de treinamento e validação foram registrados para acompanhar o desempenho do modelo ao longo das iterações. Além disso, as acurácias de treinamento e validação, exibidas na figura 3, foram calculadas para avaliar a capacidade do modelo em classificar corretamente as instâncias. Nota-se que o modelo melhora com o aumento do número de épocas de treinamento, atingindo uma espécie de platô com números mais elevados.

A busca em grade foi realizada para otimizar os hiperparâmetros do modelo de Regressão Logística. Diferentes combinações de *solvers*, *penalties* e parâmetros de regularização foram exploradas utilizando a biblioteca Scikit-learn para encontrar a melhor combinação de hiperparâmetros com base no conjunto de treinamento.

Por fim, o modelo final, com os melhores hiperparâmetros, foi avaliado em relação ao conjunto de validação. Métricas de desempenho, como acurácia, matriz de confusão, relatório de classificação e a curva ROC - exibida na figura 4, em que fica evidente o perfil típico -, foram calculadas para analisar a capacidade do modelo de classificar corretamente as instâncias entre as classes positiva e negativa, proporcionando uma avaliação completa do algoritmo de Classificação Binária utilizando a Regressão Logística.

Já na última etapa, para o problema de rede MLP, foi realizado o carregamento do conjunto de dados contendo informações sobre diabetes em mulheres. Os atributos foram separados das classes, permitindo a compreensão dos dados e a visualização de sua distribuição por meio de histogramas. Essa análise inicial proporcionou insights importantes sobre a natureza dos dados e suas características.

Para garantir a robustez do modelo de classificação, foi necessário equilibrar as classes presentes no conjunto de dados. Para isso, os pesos das classes foram calculados e posteriormente utilizados no ajuste do algoritmo de classificação. Essa etapa foi essencial para evitar um viés em direção à classe majoritária e melhorar o desempenho do modelo em relação às classes minoritárias.

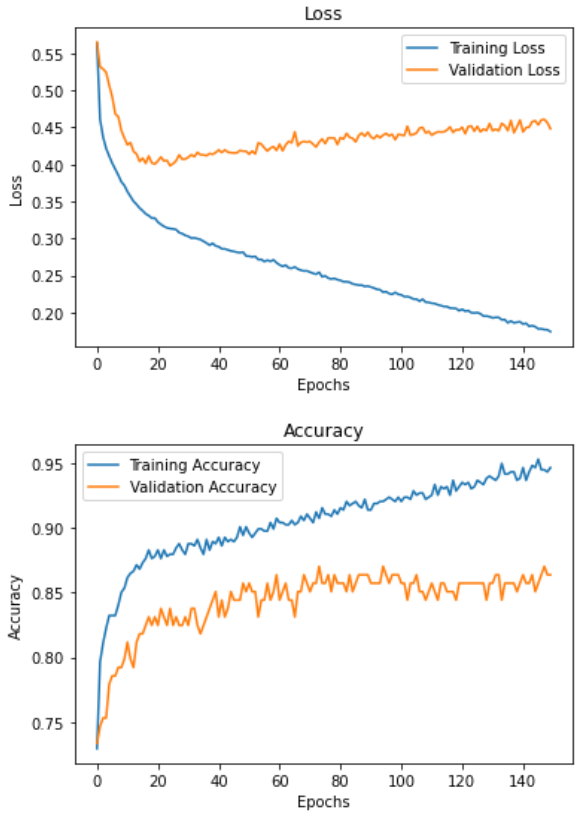

Figura 5 – Métricas de treinamento e validação do histórico - fonte: código próprio

A fim de preparar os dados para a aplicação das redes neurais artificiais, os atributos foram pré-processados. Os valores não numéricos foram convertidos, permitindo a utilização dessas informações na etapa de treinamento do modelo. Além disso, valores faltantes nos atributos foram tratados, sendo substituídos pela média dos valores presentes no conjunto de treinamento.

Com o objetivo de garantir que todos os atributos possuam a mesma escala, os dados foram normalizados. Essa etapa é fundamental para evitar que um atributo com maior amplitude domine o aprendizado do modelo em relação a outros atributos, garantindo, assim, um melhor desempenho e convergência durante o treinamento.

Uma MLP foi escolhida para a classificação dos dados. O modelo foi configurado com variação no número de neurônios na camada intermediária, possibilitando a busca pela melhor configuração que resultasse na maior acurácia no conjunto de treinamento. Para esse fim, um loop foi utilizado para avaliar o desempenho do modelo em diferentes cenários. A figura 5 mostra dois gráficos: a evolução das curvas de perda e de acurácia, respectivamente, deixando claro um possível *overfitting* à partir da 20ª iteração.

Com base na análise das métricas de desempenho, especialmente a acurácia no conjunto de validação, foi possível identificar o número ótimo de neurônios na camada intermediária. Essa configuração proporcionou o melhor resultado no conjunto de validação, garantindo maior precisão na classificação.

Por meio das curvas de perda e acurácia, foi possível acompanhar a evolução do modelo ao longo das épocas de treinamento. Essas curvas forneceram insights sobre o desempenho do modelo em diferentes estágios e auxiliaram na identificação de possíveis problemas, como overfitting ou underfitting.

O modelo com a melhor acurácia no conjunto de validação foi salvo para uso futuro. Essa abordagem garante a reutilização do modelo otimizado em tarefas futuras, economizando recursos computacionais e tempo de treinamento.

RESULTADOS E DISCUSSÃO:

No decorrer deste trabalho, exploramos conceitos essenciais em Inteligência Artificial e Aprendizado de Máquina, aplicando técnicas como Regressão Linear, Regressão Logística e Classificação Binária em diferentes conjuntos de dados. Posteriormente, desenvolvemos uma rede neural MLP para tarefas de classificação. Agora, nosso foco é construir uma Rede Neural Profunda Convolucional (CNN) para reconhecimento de dígitos MNIST, que é uma aplicação clássica de visão computacional. O objetivo é projetar e implementar uma arquitetura poderosa, capaz de identificar corretamente os dígitos escritos à mão, utilizando camadas convolucionais, de pooling e totalmente conectadas.

Com base na base teórica e prática adquirida até o momento, estamos preparados para enfrentar esse desafio. O treinamento e a otimização da CNN serão etapas cruciais para ajustar os parâmetros da rede e obter resultados precisos e robustos. Além disso, nossa experiência com outras técnicas de Aprendizado de Máquina permitirá explorar métodos de avaliação, como validação cruzada e curvas de aprendizado, para evitar problemas de overfitting.

Em suma, este trabalho possui um caráter formativo, culminando na construção de uma Rede Neural Profunda Convolucional voltada ao reconhecimento de dígitos MNIST.

CONCLUSÕES:

Ao longo deste trabalho de iniciação científica, exploramos conceitos fundamentais em Inteligência Artificial e Aprendizado de Máquina, aplicamos técnicas de Regressão Linear, Regressão Logística e Classificação Binária, desenvolvemos uma rede neural MLP para tarefas de classificação e nos dedicamos à construção de uma Rede Neural Profunda Convolucional para o reconhecimento de dígitos MNIST.

O processo nos proporcionou uma visão abrangente sobre o poder das redes neurais e sua aplicabilidade a tarefas complexas de visão computacional. Esperamos que a contribuição possa, no futuro, levar a novas pesquisas em redes neurais, impulsionando avanços em Inteligência Artificial.

REFERÊNCIAS

[Boccato e Attux, 2020] L. Boccato, R. Attux, **Notas de Aula do Curso IA048 – Aprendizado de Máquina**, FEEC/UNICAMP, 2020.

[Goodfellow et al., 2016] I. Goodfellow, Y. Bengio, A. Courville, **Deep Learning**, MIT Press, 2016.

[Haykin, 2008] S. Haykin, **Neural Networks and Learning Machines**, Pearson, 2008.

[LeCun et al., 2015] Y. LeCun, Y. Bengio, G. Hinton, **“Deep Learning”**, Nature, Vol. 521, pp. 436 – 444, 2015.