RELATÓRIO: Redes Neurais

Gabriel Vieira da Cruz

10 de Fevereiro de 2024

1. Redes neurais artificiais

1.2 Neurônio artificial

- Unidade básica de uma rede neural.
- Cada neurônio recebe entradas (dados) que são multiplicadas pelos seus respectivos pesos, e então são usadas como parâmetro para uma função de ativação (define a saída do neurônio).
- Basicamente a aprendizagem da rede é **descobrir os pesos**.
- Os pesos são atualizados utilizando o **backpropagation** (indica como ajustar os pesos) e a **taxa de aprendizagem** (indica a frequência para atualizar os pesos).

1.3 Redes Multicamada

- Para fazer classificações não linearmente separáveis, é necessário fazer uso de camadas escondidas (camada(s) intermediária entre a primeira e a última que visa melhorar a precisão da rede) e funções de ativação mais sofisticadas.
- **Função de soma:** Somatório das entradas multiplicadas pelos seus respectivos pesos
- **Função de ativação:** Resumidamente, transforma as saídas lineares dos neurônios em saídas não lineares.
- Passo a passo do treinamento:
    1. Apresentação da tabela de dados do operador XOR.
    2. Cálculo passo a passo da rede neural para o primeiro registro:
        a. Multiplicação dos pesos pelas entradas.
        b. Aplicação da função soma.
        c. Aplicação da função de ativação na camada escondida.
        d. Aplicação da função soma e de ativação na camada de saída.
        e. Cálculo do erro.
    3. Repetição dos cálculos para os demais registros.
- Otimização dos pesos para reduzir os erros

1.4 Erro

- **Erro simples:** $erro = respostaCorreta - respostaCalculada$
- **Mean Square Error (MSE):** Somatório dos erros (erro simples) ao quadrado, dividido pelo número de elementos.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2$$

- **Root Mean Square Error(RMSE):** Basicamente é o cálculo da raiz quadrada do MSE ($\sqrt{MSE}$).

$$\mathbf{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i - o_i)^2}$$

1.4.1 Gradiente

É feita a derivada parcial do erro em relação ao peso indicando como o erro muda quando o peso é modificado. Os pesos da rede são então atualizados na direção oposta ao gradiente do erro, ou seja, os pesos que contribuem mais para o erro são ajustados mais significativamente.

1.4.2 Delta

O delta determina a direção e a magnitude das atualizações dos pesos. O Delta é usado para calcular o Gradiente e o Gradiente é usado para atualizar os pesos da rede neural.

Passos para o cálculo do Delta:

- Aplicar a função de ativação no valor da soma.
- Calcular a derivada da função de ativação.
- Multiplicar o erro pela derivada da função de ativação para obter o Delta.

Para calcular o Delta em uma **camada escondida** (Delta Escondido), primeiramente é calculado o Delta de saída:

- $Delta\ Saída = Erro \times Derivada\ da\ Função\ de\ Ativação$
- $Delta\ Escondido = Peso \times Derivada\ da\ Função\ de\ Ativação \times Delta\ Saída$

1.4.3 Backpropagation

O backpropagation é um algoritmo essencial para o treinamento de redes neurais que permite o ajuste eficiente dos seus respectivos pesos.

Fórmula genérica para a aplicação do algoritmo:

$peso_{n+1} = (peso_n \times momento) + entrada \times delta \times taxa\ de\ aprendizagem$

Parâmetros:

- **peso:** O peso é o que o algoritmo busca de forma iterativa. O peso atual ($peso_n$) é usado para encontrar o $peso_{n+1}$.
- **momento:** Controla a influência dos pesos anteriores na atualização dos pesos atuais.
  - **Valor alto:** Aumenta a estabilidade do treinamento e acelera a convergência, mas pode reduzir a flexibilidade do algoritmo.

    - **Valor baixo:** Permite um aprendizado mais rápido, mas pode levar a oscilações nos pesos e dificultar a convergência.
- **taxa de aprendizagem:** Controla a magnitude da atualização dos pesos.
    - **Valor alto:** Aprendizado mais rápido, mas pode levar a instabilidade e oscilações nos pesos.
    - **Valor baixo:** Aprendizado mais lento, mas mais estável e preciso.

A mesma fórmula é utilizada tanto da camada oculta para a saída quanto da entrada para a camada oculta, porém há algumas diferenças na função de ativação, cálculo do delta, momento e taxa de aprendizagem.

1.4.4 Bias

O Bias é adicionado nas camadas de entrada e saída com o intuito de modificar o comportamento dos neurônios individualmente, como **valores diferentes mesmo se todas as entradas forem zero** ou **muda a saída com a unidade de bias.**

1.4.5 Gradient Descent

- **Batch Gradient Descent:** Calcula o erro para todos os registros e atualiza os pesos, utilizando o MSE ou o RMSE.
- **Stochastic Gradient Descent:** Calcula o erro para cada registro e atualiza os pesos individualmente (item por item). Ajuda a prevenir mínimos locais e tende a ser mais rápido já que não precisa carregar todos os dados simultaneamente.
- **Mini batch gradient descent:** Escolhe um número de registros para rodar e atualizar os pesos.

Parâmetros:

- **Batch size (tamanho do lote):** Determina o número de registros para atualizar os pesos (Mini Batch Gradient Descent).
- **Epochs (épocas):** Número de atualizações dos pesos.

2. Prática

2.1 Classificação binária

Um modelo de classificação binária é de simples aplicação justamente por conta de sua saída que conterá um valor muito próximo de 0 ou de 1, e também por conta da disponibilidade de ferramentas/funcionalidades que automatizam todo o processo de montagem da rede neural, desde as camadas até seus parâmetros (como a função de ativação, distribuição inicial dos pesos, validação cruzada, etc…). É possível criar seus próprios parâmetros ou personalizar os já pré-existentes nas bibliotecas.

2.2 Classificação Multi-classes

Em um modelo de múltipla classificação, o problema se concentra principalmente no pré-processamento. O conjunto de dados pode apresentar NaN's, atributos desprezíveis para o aprendizado do modelo, atributos categóricos, etc… Uma vez que os dados influenciam diretamente no aprendizado do modelo, é de extrema importância se certificar de tratar os dados com muita cautela.

2.3 Regressão

Na regressão, também há uma certa preocupação no pré-processamento (como todos os modelos), mas sua diferença é, principalmente, a saída, que não busca categorizar/classificar um conjunto de dados, mas sim determinar um novo valor com base nos atributos fornecidos, por conta disso sua função de ativação possui um leque restrito de opções.

Considerações

Durante as aulas vistas eu fiz o código anotando parâmetros e/ou papel das principais funções e métodos utilizados. como o primeiro contato prático foi no módulo de número 4 - Classificação Binária (que é o módulo com maior densidade de conteúdo do curso), os códigos feitos nesse módulo possuem mais anotações que os códigos dos módulos posteriores, uma vez que o conhecimento chave do curso (estrutura de uma rede neural) é adquirido, os outros módulos não possuem uma discrepância tão grande de conteúdo novo entre si.