Redes Neurais Artificais - Exercício Adaline

Marcio R. A. Souza Filho - 2015104105

Agosto 2020

1 Problema e Metodologia

Para resolver os exercícios 1 e 2, foi feita a implementação do Adaline e seu treinamento seguindo exatamente o pseudo algoritmo apresentado nos materiais do professor Antônio P. Braga (p. 41). Por conta disso, nas explicações estou supondo que possuo uma função $TreinaAdaline$, que é implementada seguindo o algoritmo acima.

Como os exercícios 1 e 2, são em essência o mesmo problema, vou explicar de forma geral a resolução do problema que se aplica aos dois casos.

Os problemas nos exercícios 1 e 2 consistem em, utilizando o Adaline, encontrar uma função $f(x) = y$, onde $x = [x_1, x_2, \ldots, x_n]$, que seja uma combinação linear dos termos de x adicionado de um termo independente, relacionado a uma possível translação da combinação linear, ou seja, $f(x) = w \cdot x + C$, na qual $w = [w_1, w_2, \ldots, w_n]$ é o vetor de pesos e C uma constante. Para facilitar as operações entre vetores, foi considerado $x = [x_1, x_2, \ldots, x_n, 1]$ e $w = [w_1, w_2, \ldots, w_n, C]$.

Essa função deve possuir um erro quadrático médio em relação aos dados do problema, menor do que uma tolerância definida.

Como o objetivo é aproximar uma função que é combinação linear da entrada, a função de ativação do Adaline escolhida foi a função identidade:

$$f(u) = u$$

Para o treinamento do Adaline é preciso definir os parâmetros: número máximo de épocas ($nEpocas$) do treinamento e tolerância (tol) do erro quadrático médio em relação às dados de treinamento. Por conta do caráter didático desse exercício, os valores dos parâmetros foram definidos de forma que o algoritmo pudesse ser rapidamente executado a cada alteração. Os valores utilizados foram:

$$nEpocas = 100$$

$$tol = 10e - 12$$

Além disso, para separar os dados de treinamento dos dados de teste, implementei uma função para separar os índices de forma que os dados ficassem distribuídos por toda extensão de x.

Abaixo está o pseudo algoritmo utilizado para a solução dos exercícios. Considerei que os dados dos arquivos já foram lidos e disponíveis no início do algoritmo.

```
nEpocas = 100;
tol = 1e-14;
percentualTreino = 0.7;
tempoAmostras = dados de tempo;
xAmostras = dados de X;
Adiciona uma coluna com valor 1 ao final de cada vetor de xAmostras;
yAmostras = dados de Y;
xDim = (dimensão dos vetores de xAmostras) + 1;
w = [0]*nDim //Vetor de dimensão xDim com valor 0 para cada elemento;
[indicesTreino, indicesTeste] = separaIndices(nAmostras, percentualTreino);
xTreino = xAmostras[indicesTreino];
yTreino = yAmostras[indicesTreino];
xTeste = xAmostras[indicesTeste];
yTeste = yAmostras[indicesTeste];
TreinaAdaline(xTreino, yTreino, nEpocas, tol, w);
yTeste = AvaliaAdaline(xTeste);
erroQuadMedio = media( (yTeste - y)**2);
print(erroQuadMedio);
print(w);
yAprox = AvaliaAdaline(xAmostras);
plot(tempoAmostras, yAprox);
```

2 Respostas

As repostas consistem no erro quadrático médio obtido, os parâmetros do modelo, o gráfico da saída para todos os pontos de entrada e uma discussão sobre o que foi obtido.

2.1 Exercício 1

- Erro quadrático médio: 1.3680736010105402e-16.
- $w = [0.29999999, 0.5]$.
- Analogamente, para a função $f(x) = a * x + b$, seria $a = 0.29999999$ e $b = 0.5$.

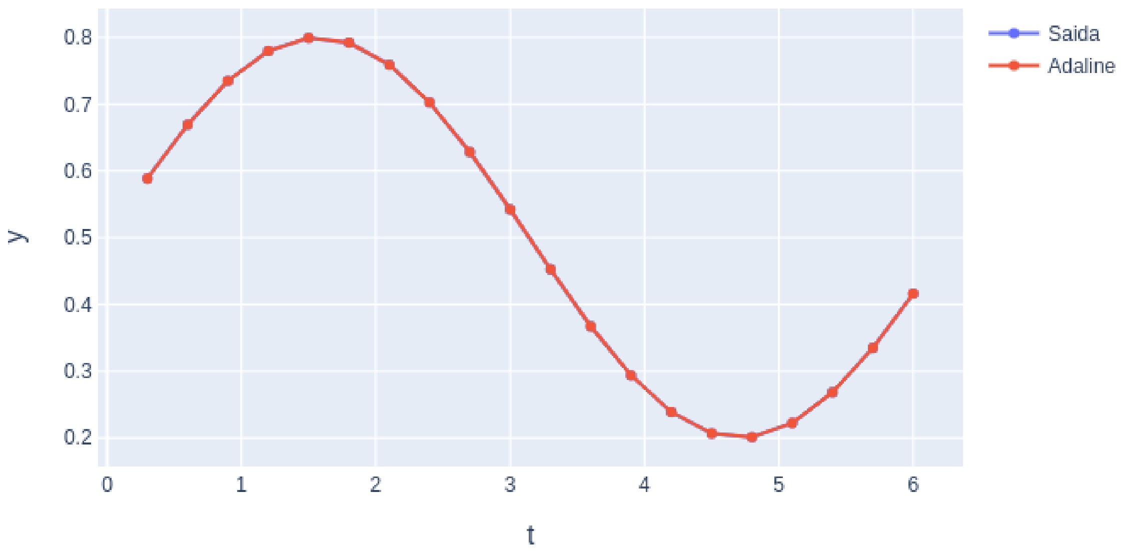

Figura 1: Saída para o Adaline para todos os pontos de entrada.

O resultados mostram claramente que o Adaline, conseguiu aproximar a função de geradora de forma que o erro entre a saída obtida através do modelo apresenta um erro quadrático médio da ordem de 1e-16. É importante lembrar que os dados de entrada e saída utilizados, possuem precisão da ordem de 1e-14. Então nota-se que o modelo se aproximou da função geradora até o limite do que os dados permitiam.

Observando o gráfico de saída para todos os pontos de entrada, fica ainda mais evidente como o modelo se aproximou bastante dos dados do sistema. No gráfico a curva referente ao Adaline superpõe completamente a saída dos dados originais.

2.2 Exercício 2

- Erro quadrático médio: 1.9504451872672573e-16.
- $w = [0.999999992.000000012.999999981.57079633]$.
- Analogamente, para a função $f(x) = a * x_1 + b * x_2 + c * x_3 + d$, seria $a = 0.99999999$, $b = 2.00000001$, $c = 2.99999998$ e $d = 1.57079633$.

A análise que pode ser feita a partir dos resultados é a mesma feita no exercício 1, somada à observação do poder de aproximação do Adaline para uma entrada de dimensão maior que no exercício 1.

O gráfico plotado não possui os pontos de entrada, pois poluiria bastante a visualização.

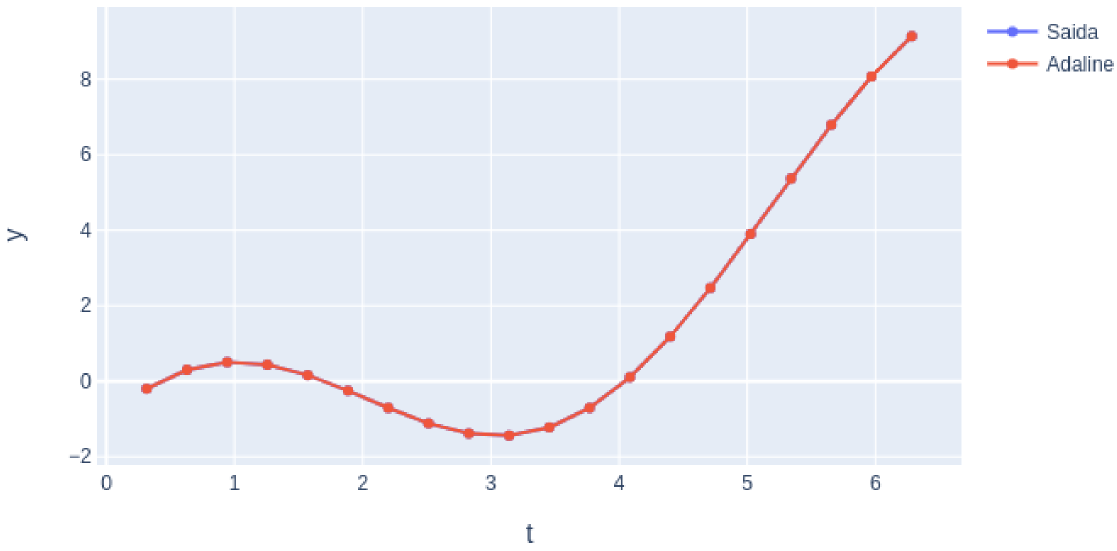

Figura 2: Saída para o Adaline para todos os pontos de entrada.