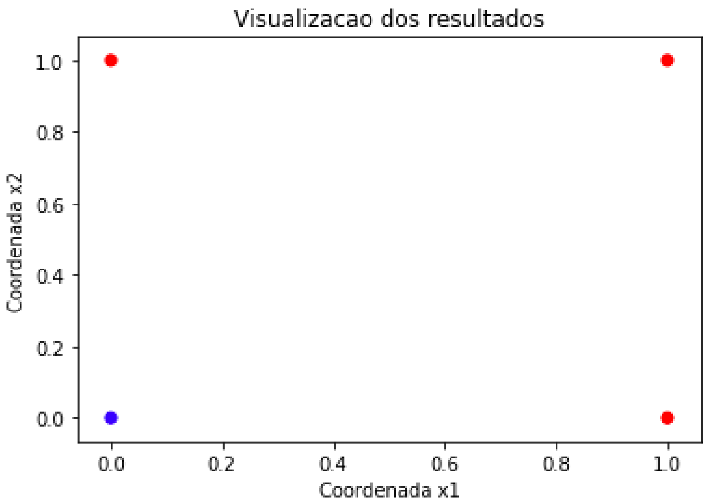
Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Data: 26 de Setembro de 2018
Professora: Elloá B. Guedes
Disciplina: Redes Neurais Artificiais

ATIVIDADE PRÁTICA

Para consolidar esta atividade, os alunos devem se organizar em duplas balanceadas e produzir um `Jupyter notebook` para praticar o aprendizado de neurônios *perceptron*. Estes neurônios são conceitualmente simples e individualmente capazes de resolver problemas linearmente separáveis. O problema linearmente separável a ser considerado é o da porta lógica *NAND*, caracterizada pelas seguintes entradas e saídas:

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0



A partir de uma inspeção visual é possível notar que este problema é linearmente separável. Nos resta, então, mostrar estes exemplos a um neurônio e deixar que o mesmo ajuste os pesos até o aprendizado da fronteira de decisão adequada.

1 Conjunto de Treinamento

Neste exemplo, o conjunto de treinamento é composto pelos atributos preditores x_1 e x_2 e atributo alvo y. No caso do neurônio *perceptron*, vamos acrescentar à entrada um *bias* igual a -1. De maneira breve, o *bias* atua auxiliando na formulação matemática, auxiliando a determinar a fronteira de decisão para o problema em questão. Desta maneira, teremos o seguinte conjunto de treinamento.

```
# conjunto de treinamento com bias e conversao para numpy.array
X = [[-1,0,0],[-1,0,1],[-1,1,0],[-1,1,1]]
for i in range(len(X)):
    X[i] = np.asarray(X[i])

Y = [1,0,0,0]
```

2 Neurônio Perceptron

Neste problema, o neurônio *perceptron* deve possuir vetor W com 3 pesos inicializados aleatoriamente $\sim U(-0.5, 0.5)$. Os pesos atuarão sobre a entrada produzindo uma combinação linear, que passará pela função de ativação. A função de ativação a ser considerada será a função degrau com limiar igual a zero.

$$f(u = \sum_{i=1}^{3} x_i \cdot w_i) = \begin{cases} 1, & u \geq 0. \\ 0, & c.c \end{cases} \tag{1}$$

O neurônio em questão deve examinar os exemplos de entrada e ajustar os pesos caso esteja produzindo uma solução diferente da desejada. Este processo é iterativo e continua até que, no caso de um problema linearmente separável, não haja mais erros de classificação. Este ajuste de pesos respeita à seguinte equação:

$$w(n+1) = w(n) + \eta \cdot (y_i - \hat{y}_i) \cdot x_i, \tag{2}$$

em que y_i é a saída correta para a entrada x_i, $\hat{y}_i$ é a saída produzida pelo neurônio para a entrada x_i e η é a taxa de aprendizado.

Assim, implemente o aprendizado de um neurônio perceptron para o conjunto de treinamento X e rótulos Y relativos à função $NAND$. Seu algoritmo deve ilustrar claramente:

1. O vetor de pesos a cada época;
2. A quantidade de exemplos corretos e errados a cada época;
3. A quantidade de épocas até a convergência;

4. A quantidade de vezes em que houve ajuste no vetor de pesos.

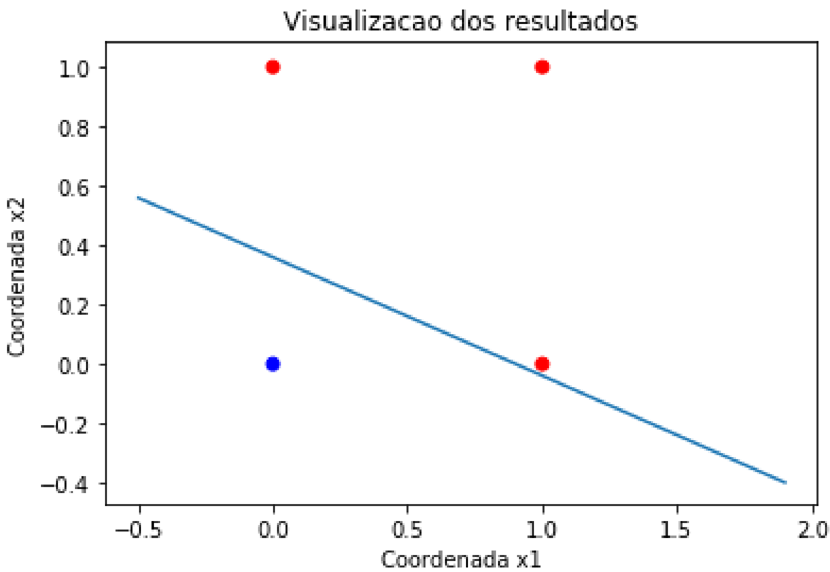
Para favorecer a prática dos conceitos relativos à biblioteca `numpy`, todo o processamento necessário deve, sempre que possível, ser implementado com funções de manipulação de matrizes disponíveis neste pacote.

3 Visualizando a Fronteira de Decisão

Após o processo de aprendizado, tem-se que o neurônio aprendeu os valores corretos do vetor de pesos capaz de separar adequadamente as classes deste exemplo. Para visualizar a fronteira de decisão a partir deste vetor de 3 pesos, construa uma reta da seguinte forma:

$$y = W[0]/W[2] - (W[1]/W[2]) * x, \tag{3}$$

em que os pesos identificados são utilizados para obter a forma canônica da reta característica da fronteira de decisão. Denote as classes em cores distintas, para facilitar a visualização dos resultados. O exemplo a seguir ilustra a construção desejada com resultados obtidos a partir do treinamento para aprendizado da função OR.



4 Funções de Ativação

Para complementar o entendimento para as próximas etapas da disciplina, apresente também gráficos para as seguintes funções de ativação, considerando o intervalo $[-1.5, 1.5]$:

1. Degrau (com $\theta = 0.5$);
2. Sigmoidal, com 3 valores distintos de suavidade e centro da curva no valor 0;
3. Tangente Hiperbólica;
4. ReLU.