Inteligência Artificial

Lista 06

Bernard P. Ferreira[1]
[1] Instituto de Ciências Exatas e Informática - Pontifícia Universidade Católica de Minas Gerais (PUC-MG) - Belo Horizonte - MG – Brasil

Questão 01:

Considere as seguintes afirmações sobre redes neurais artificiais:

I. Um perceptron elementar só computa funções linearmente separáveis.

II. Não aceitam valores numéricos como entrada.

III. O "conhecimento" é representado principalmente através do peso das conexões.

São corretas:

C) Apenas I e III

Questão 02:

Considere as funções booleanas abaixo:

I. $p \wedge q$ (conjunção)

II. $p \leftrightarrow q$ (equivalência)

III. $p \rightarrow q$ (implicação)

Quais destas funções podem ser implementadas por um perceptron elementar?

A) Somente I;

Questão 03:

Dado um perceptron simples de duas entradas e um bias, cujos pesos são w1=0,5, w2=0,4 e w0=-3, respectivamente, assinalar a resposta correta:

c) O perceptron realiza a função OR

Questão 04:

Considerando a base de dados (breast-cancer.csv) e utilizando-se o notebook “RNA.ipynb” (faça as adaptações necessárias) que está no CANVAS, pede-se:

1) Faça os pré-processamentos necessários para esta base: conversão de nominal para numérico, identificação de outlier, normalização, balanceamento, eliminação de redundância, dentre outros

2) Avalie o desempenho do modelo a partir de diferentes topologias: números de camadas e neurônios. Use diferentes heurísticas para isso, conforme discutido em sala.

3) Avalie o parâmetro ‘taxa de aprendizado’ e veja sua relação com a quantidade de épocas.

4) Caso julgue necessário, investigue outros hiperparâmetros necessários para a rede neural a fim de melhorar os resultados obtidos. Compare os resultados obtidos.

5) Ajuste os hiperparâmetros automaticamente, usando métodos como Grid Search, CVParameterSelection e MultiSearch.

https://github.com/BernardPaes/Inteligencia-Artificial

- Adam; Relu; 10 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.92	0.81	36
recurrence-events	0.77	0.45	0.57	22
accuracy			0.74	58
macro avg	0.75	0.69	0.69	58
weighted avg	0.75	0.74	0.72	58

- Adam; Relu; 100 layers

	precision	recall	f1-score	support
no-recurrence-events	0.75	0.92	0.82	36
recurrence-events	0.79	0.50	0.61	22
accuracy			0.76	58
macro avg	0.77	0.71	0.72	58
weighted avg	0.76	0.76	0.74	58

- Adam; Relu; 1000 layers

	precision	recall	f1-score	support
no-recurrence-events	0.72	0.86	0.78	36
recurrence-events	0.67	0.45	0.54	22
accuracy			0.71	58
macro avg	0.69	0.66	0.66	58
weighted avg	0.70	0.71	0.69	58

- Adam; Sigmoidal ; 10 layers

	precision	recall	f1-score	support
no-recurrence-events	0.72	0.86	0.78	36
recurrence-events	0.67	0.45	0.54	22
accuracy			0.71	58
macro avg	0.69	0.66	0.66	58
weighted avg	0.70	0.71	0.69	58

- Adam; Sigmoidal; 100 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.92	0.81	36
recurrence-events	0.77	0.45	0.57	22
accuracy			0.74	58
macro avg	0.75	0.69	0.69	58
weighted avg	0.75	0.74	0.72	58

- Adam; Sigmoidal; 1000 layers

	precision	recall	f1-score	support
no-recurrence-events	0.72	0.86	0.78	36
recurrence-events	0.67	0.45	0.54	22
accuracy			0.71	58
macro avg	0.69	0.66	0.66	58
weighted avg	0.70	0.71	0.69	58

- Adam; Tanh; 10 layers

	precision	recall	f1-score	support
no-recurrence-events	0.74	0.97	0.84	36
recurrence-events	0.91	0.45	0.61	22
accuracy			0.78	58
macro avg	0.83	0.71	0.72	58
weighted avg	0.81	0.78	0.75	58

- Adam; Tanh; 100 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.92	0.81	36
recurrence-events	0.77	0.45	0.57	22
accuracy			0.74	58
macro avg	0.75	0.69	0.69	58
weighted avg	0.75	0.74	0.72	58

- Adam; Tanh;1000 layers

	precision	recall	f1-score	support
no-recurrence-events	0.74	0.97	0.84	36
recurrence-events	0.91	0.45	0.61	22
accuracy			0.78	58
macro avg	0.83	0.71	0.72	58
weighted avg	0.81	0.78	0.75	58

- SGD; Relu; 10 layers

	precision	recall	f1-score	support
no-recurrence-events	0.74	0.89	0.81	36
recurrence-events	0.73	0.50	0.59	22
accuracy			0.74	58
macro avg	0.74	0.69	0.70	58
weighted avg	0.74	0.74	0.73	58

- SGD; Relu; 100 layers

	precision	recall	f1-score	support
no-recurrence-events	0.74	0.94	0.83	36
recurrence-events	0.83	0.45	0.59	22
accuracy			0.76	58
macro avg	0.79	0.70	0.71	58
weighted avg	0.77	0.76	0.74	58

- SGD; Relu; 1000 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.92	0.81	36
recurrence-events	0.77	0.45	0.57	22
accuracy			0.74	58
macro avg	0.75	0.69	0.69	58
weighted avg	0.75	0.74	0.72	58

- SGD; Sigmoidal ; 10 layers

	precision	recall	f1-score	support
no-recurrence-events	0.72	0.94	0.82	36
recurrence-events	0.82	0.41	0.55	22
accuracy			0.74	58
macro avg	0.77	0.68	0.68	58
weighted avg	0.76	0.74	0.72	58

- SGD; Sigmoidal; 100 layers

	precision	recall	f1-score	support
no-recurrence-events	0.75	0.92	0.82	36
recurrence-events	0.79	0.50	0.61	22
accuracy			0.76	58
macro avg	0.77	0.71	0.72	58
weighted avg	0.76	0.76	0.74	58

- SGD; Sigmoidal; 1000 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.89	0.80	36
recurrence-events	0.71	0.45	0.56	22
accuracy			0.72	58
macro avg	0.72	0.67	0.68	58
weighted avg	0.72	0.72	0.71	58

- SGD; Tanh; 10 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.89	0.80	36
recurrence-events	0.71	0.45	0.56	22
accuracy			0.72	58
macro avg	0.72	0.67	0.68	58
weighted avg	0.72	0.72	0.71	58

- SGD; Tanh; 100 layers

	precision	recall	f1-score	support
no-recurrence-events	0.77	0.94	0.85	36
recurrence-events	0.86	0.55	0.67	22
accuracy			0.79	58
macro avg	0.81	0.74	0.76	58
weighted avg	0.80	0.79	0.78	58

- SGD; Tanh;1000 layers

	precision	recall	f1-score	support
no-recurrence-events	0.73	0.92	0.81	36
recurrence-events	0.77	0.45	0.57	22
accuracy			0.74	58
macro avg	0.75	0.69	0.69	58
weighted avg	0.75	0.74	0.72	58

Questão 05:

Implemente o algoritmo Perceptron e teste-o com as funções booleanas AND e OR com n entradas.

O usuário deve fornecer a quantidade de entradas (função booleana de 2, 3, 4... n entradas) e a função (AND ou OR) desejadas.

Atenção: É necessário implementar o Perceptron. Ou seja, não será considerado fazer chamadas a funções

prontas.

Você deverá fazer um pequeno texto, contendo:

1) Uma breve explicação do Perceptron

   O algoritmo Perceptron é um modelo de aprendizado de máquina baseado em redes neurais artificiais. Proposto por Frank Rosenblatt em 1957, o Perceptron é considerado um dos primeiros algoritmos de aprendizado supervisionado.

   Sua estrutura consiste em uma única camada de neurônios (ou perceptrons). Cada neurônio recebe entradas ponderadas e produz uma saída binária (0 ou 1). As entradas são multiplicadas pelos pesos associados a cada conexão e somadas. Se a soma ponderada exceder um determinado limiar, o neurônio é ativado (produzindo uma saída de 1); caso contrário, permanece inativo (saída de 0).

   O treinamento do Perceptron envolve ajustar os pesos das conexões para que o modelo possa aprender a classificar corretamente os exemplos de treinamento. Utiliza-se uma regra de atualização baseada no erro entre a saída prevista e a saída real. Se o exemplo for classificado incorretamente, os pesos são atualizados para corrigir o erro. O processo de treinamento continua até que o Perceptron alcance uma taxa de erro aceitável ou atinja um número máximo de iterações.

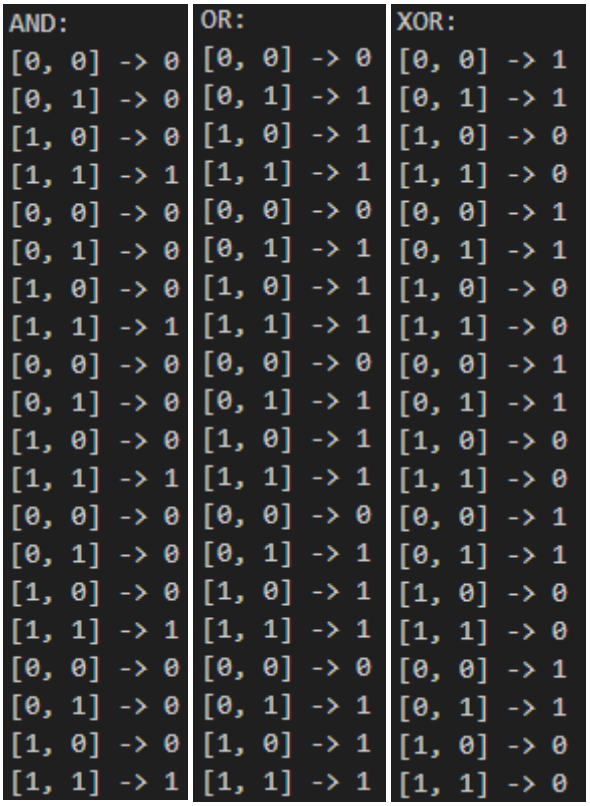
   No entanto, o Perceptron possui limitações. Ele é adequado apenas para problemas linearmente separáveis, o que significa que não pode aprender funções complexas que não podem ser separadas por uma linha reta. Além disso, não lida bem com dados ruidosos ou com sobreposição de classes.

2) O código desenvolvido (favor compartilhar o link para o código)
   https://github.com/BernardPaes/Inteligencia-Artificial

3) Imagens mostrando os resultados obtidos para a função booleana AND

4) Imagens mostrando os resultados obtidos para a função booleana OR

5) Imagens mostrando que o perceptron não resolve o XOR



Questão 06:

Implemente o algoritmo Backpropagation e mostre que ele resolve o XOR, além de AND e OR.

O usuário deve fornecer a quantidade de entradas (função booleana de 2, 3, 4... n entradas) e a função (AND, XOR, OR) desejadas.

Atenção: É necessário implementar o Backpropagation. Ou seja, não será considerado fazer chamadas a funções prontas.

Você deverá fazer um pequeno texto, contendo:

1) Uma breve explicação do Backpropagation
   O backpropagation é um algoritmo crucial no campo do aprendizado de máquina e redes neurais. Ele permite ajustar os pesos e viés (bias) de uma rede neural com base no gradiente da função de erro. Essa técnica é fundamental para treinar redes profundas e resolver tarefas complexas.
   Durante o treinamento, os dados de entrada são alimentados na rede, e a propagação direta (feedforward) ocorre. Os sinais passam pelas camadas da rede, multiplicando-se pelos pesos e aplicando funções de ativação nos neurônios. A saída da rede é comparada com a saída esperada usando uma função de erro.
   O algoritmo calcula o gradiente da função de erro em relação aos pesos da rede. Para cada peso, o gradiente indica como a função de erro muda quando o peso é ajustado. Essa informação é crucial para otimizar os pesos corretamente durante o treinamento.

O gradiente é propagado de volta (da saída para a entrada) através da rede. Os pesos são atualizados iterativamente usando uma taxa de aprendizado multiplicada pelo gradiente.

O backpropagation é superior ao Perceptron tradicional por vários motivos. Ele permite redes neurais profundas, capazes de aprender representações hierárquicas e resolver problemas não linearmente separáveis. Além disso, sua flexibilidade e poder de adaptação tornam-no essencial para a IA moderna. Em resumo, o backpropagation representa um avanço significativo, possibilitando a criação de modelos sofisticados e a solução de desafios complexos

2) O código desenvolvido (favor compartilhar o link para o código)

https://github.com/BernardPaes/Inteligencia-Artificial

3) Imagens mostrando os resultados obtidos para a função booleana AND

4) Imagens mostrando os resultados obtidos para a função booleana OR

6) Imagens mostrando que o Backpropagation resolve o XOR

Resultados AND

```
[[0 0] [[0.00478699]
 [0 1]  [0.03672231]
 [1 0]  [0.03662911]
 [1 1]  [0.94863443]
 [0 0]  [0.00478699]
 [0 1]  [0.03672231]
 [1 0]  [0.03662911]
 [1 1]  [0.94863443]
 [0 0]  [0.00478699]
 [0 1]  [0.03672231]
 [1 0]  [0.03662911]
 [1 1]  [0.94863443]
 [0 0]  [0.00478699]
 [0 1]  [0.03672231]
 [1 0]  [0.03662911]
 [1 1]  [0.94863443]
 [0 0]  [0.00478699]
 [0 1]  [0.03672231]
 [1 0]  [0.03662911]
 [1 1]] [0.94863443]]
```

Resultado OR

```
[0 0] [0.05167885]
[0 1] [0.97103487]
[1 0] [0.97113084]
[1 1] [0.99105006]
[0 0] [0.05167885]
[0 1] [0.97103487]
[1 0] [0.97113084]
[1 1] [0.99105006]
[0 0] [0.05167885]
[0 1] [0.97103487]
[1 0] [0.97113084]
[1 1] [0.99105006]
[0 0] [0.05167885]
[0 1] [0.97103487]
[1 0] [0.97113084]
[1 1] [0.99105006]
[0 0] [0.05167885]
[0 1] [0.97103487]
[1 0] [0.97113084]
[1 1] [0.99105006]
```

Resultados XOR

```
[0 0][0.10124564]
[0 1][0.92688977]
[1 0][0.92005948]
[1 1][0.05932789]
[0 0][0.10124564]
[0 1][0.92688977]
[1 0][0.92005948]
[1 1][0.05932789]
[0 0][0.10124564]
[0 1][0.92688977]
[1 0][0.92005948]
[1 1][0.05932789]
[0 0][0.10124564]
[0 1][0.92688977]
[1 0][0.92005948]
[1 1][0.05932789]
[0 0][0.10124564]
[0 1][0.92688977]
[1 0][0.92005948]
[1 1][0.05932789]
```