Universidade Federal de Jataí

PROFMAT - Mestrado Profissional em Matemática em Rede Nacional

SEQUÊNCIA DIDÁTICA: A MATEMÁTICA POR TRÁS DAS INTELIGÊNCIAS ARTIFICIAIS DO TIPO CHATBOT

Discente: Lucas Meireles Pereira

Orientador: Prof. Dr. Esdras Teixeira Costa

Como realizamos um levantamento acerca dos tópicos e habilidades matemáticas contidas na BNCC, achamos que seria interessante apresentar uma sequência didática, de forma que qualquer professor possa utilizar em sala de aula, caso ache relevante para a aprendizagem dos seus estudantes. Assim como qualquer plano de aula, nos atentamos às informações necessárias para a correta aplicação do tema “A Matemática por trás de Inteligências Artificiais do Tipo ChatBot, tema que explora Redes Neurais Artificiais.

Objetivos Gerais: O objetivo dessa sequência didática consiste em mostrar aos estudantes a presença da matemática em inteligências artificiais simples, uma vez que tal aparato já faz parte de nosso cotidiano, e ao mesmo tempo trabalhar habilidades contidas na BNCC, acerca de tópicos relevantes como funções exponenciais, valor númerico de uma função e funções definidas por mais de uma sentença.

Objetivos específicos:

- Entender o funcionamento básico de inteligências artificiais do tipo Chatbot/Redes Neurais Artificiais a nível do ensino básico.
- Compreender um pouco da matemática por trás de IAs por meio de um exemplo básico.
- Ampliar o conhecimento em tópicos como funções (Exponenciais, definidas por várias sentenças e valor numérico).

- Conhecer o código Python de forma superficial, e realizar testes por tentativa e erro.

Conteúdo: Nesta sequência estarão presentes os conteúdos de função exponencial, funções definidas por várias sentenças, valor numérico de uma função e linguagem de programação (de forma superficial).

Procedimentos metodológicos

Para a aplicação desta sequência, faremos uma separação em seis momentos. Sendo eles:

1. Neste primeiro momento é interessante que o professor retome de forma breve as definições de funções (Exponencial, definidas por várias sentenças e valor numérico) para que os estudantes relembrem esses conteúdos, visto que faz-se a necessidade que já tenham sido estudados. Para que assim possam entender as funções a serem trabalhadas nessa sequência didática:

Função ReLU (Rectified Linear Unit): $f(x) = max(0, x)$

$$f(x) = \begin{cases} 0, se\ x \leq 0 \\ x, \ \ se\ x > 0 \end{cases}$$

Função Sigmoide: Explique como a função sigmoide é usada como função de ativação.

$$f(x) = \frac{1}{1 + e^{-x}}$$

Podemos notar a presença da exponencial, valor numérico (uma vez que os estudantes irão atribuir valores para estas funções) e funções definidas por várias sentenças, como é o caso da função ReLu e de alguns limitantes em nosso código que será devidamente explicado mais à frente.

2. Em um segundo momento, o professor deverá desenvolver uma breve noção de uma Inteligência Artificial e Redes Neurais Artificiais, tal como sua evolução no

decorrer dos anos, os impacto que elas deixam e podem deixar no futuro e funcionamento, correlacionando as funções descritas com o resumo:

A inteligência artificial é uma área relativamente jovem, e possivelmente teria surgido em 1956, na Dartmouth College Conference. Porém, embora seja uma área promissora, passou por momentos de dúvida denominados “Invernos da IA” onde a Inteligência artificial sofreu diversas críticas em relação a alta expectativa da tecnologia e seu baixo retorno, implicando na retirada do suporte financeiro, em especial nas décadas de 1970 e 1990. Todavia, durante a década de 1990, muitas das técnicas computacionais foram desenvolvidas, e com o avanço tecnológico das últimas décadas, foi possível o desenvolvimento de redes mais complexas. Atualmente, atravessamos um novo momento de entusiasmo com relação aos potenciais benefícios que a inteligência artificial pode promover, com inúmeras aplicabilidades nas mais vastas áreas do conhecimento humano.

Embora haja inúmeros avanços, o termo “inteligência artificial” não é bem definido, pois além do fato de que o conceito de inteligência é subjetivo, se trata de uma área muito vasta, com diversas interpretações e conceitos que nem sempre estão alinhados.

É interessante ressaltar que a técnica de Redes Neurais Artificiais (RNAs), surgiu com o intuito de fazer uma analogia entre neurônios biológicos e circuitos eletrônicos, de modo a criar uma representação que simulasse as conexões sinápticas. Essas redes, podem aprender por meio de exemplos com diferentes técnicas de aprendizagem, e posteriormente fazer interpolações do que foi assimilado.

Redes Neurais são compostas por uma camada de entrada, uma camada de saída e pelo menos uma camada intermediária chamada de camada oculta. A camada oculta processa as informações de entrada e as transforma em um formato útil para a camada de saída. Cada uma dessas camadas é formada por neurônios artificiais que realizam cálculos para processar os dados. Na figura abaixo temos um paralelo entre um neurônio artificial e um neurônio biológico.

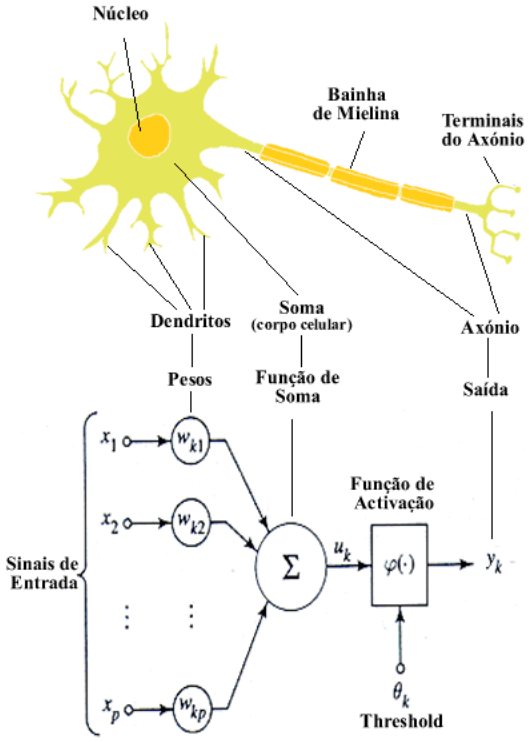


Figura 1 – Neurônio Biológico (acima) Vs Neurônio Artificial (abaixo)
Fonte: (FREITAS, 2020)

De acordo com BARONI e PADILHA (2021), o cérebro humano necessita de treinamento para cada nova tarefa, o neurônio biológico recebe sinais por meio dos diversos dendritos, que serão analisados e encaminhados ao próximo neurônio, ou não (Threshold). A partir daí, são definidos os pesos, ao qual chamamos de memorização.

Já as Redes Neurais Artificiais (RNAs), segundo os autores, são estruturas compostas por camadas de nós, sendo a primeira a camada de entrada, seguida por camadas ocultas e a camada de saída. Os neurônios em uma RNA recebem sinais de entrada, que podem vir dos dados brutos ou de neurônios em camadas anteriores, realizam cálculos e enviam sinais de saída para neurônios mais profundos na rede. Segundo BARONI e PADILHA, (2021), os neurônios podem ter sinapses conectadas a mais de um neurônio na camada anterior, e a partir daí efetuam cálculos e enviam sinais de saída para neurônios mais profundos por meio da sinapse. Cada sinapse tem um peso associado, determinando a importância do neurônio anterior na rede. Assim, o ajuste dos pesos é crucial no treinamento de modelos de aprendizado profundo. Ou seja, após receberem as entradas, os neurônios somam os sinais multiplicados pelos pesos correspondentes e os passam por uma função de ativação, que efetua o cálculo do valor de saída do neurônio, valor que é repassado adiante

para a próxima camada da rede por meio de outra sinapse.

3. Neste momento, o professor apresenta o problema “Festival de Pipas” aos estudantes, explorando e interligando a matemática e redes neurais artificiais. É interessante que os estudantes participem da leitura e de uma interpretação inicial do problema. O professor deve apresentar as variáveis a serem consideradas (velocidade do vento, precipitação de chuva e temperatura) e a função de ativação.

O problema: Festival de Pipas

Considere que a comunidade local de certa cidade quer realizar um festival de pipas. Todavia, necessitam averiguar as condições climáticas para determinar a viabilidade do evento em determinada data. Assim, os organizadores decidiram utilizar uma rede neural simples organizada da seguinte forma: 3 neurônios na camada de entrada (representando probabilidade de chuva, velocidade do vento e temperatura); 3 neurônios na camada oculta (com pesos e bias associados a cada neurônio); função de ativação ReLU $max(tx_m + vx_n + px_x + b, 0)$; um neurônio na camada de saída (output) e função de ativação Sigmoide,

$$\phi(y) = \frac{1}{1 + e^{-y}}$$

Será levado em consideração três variáveis principais: velocidade do vento, temperatura e probabilidade de chuva. Suponha que os resultados de entrada na camada oculta, sejam obtidos por meio das expressões $max(tx_1 + vx_2 + px_3 + b_1, 0)$, $max(tx_4 + vx_5 + px_6 + b_2, 0)$ e $max(tx_7 + vx_8 + px_9 + b_3, 0)$ de tal forma que t, v e p sejam a temperatura (em graus Celsius), velocidade do vento (m/s) e a probabilidade de chuva. Considere ainda que b_1, b_2 e b_3 sejam valores constantes de bias, e x_1 , x_2, x_3, x_4, x_5, x_6, x_7, x_8 e x_9, os respetivos pesos. Suponha ainda, que os resultados de saída dessa camada, sejam obtidos pela função ReLU descrita acima, de tal forma que y seja o maior valor obtido dentre a saída das três funções de entrada. Efetue tentativas atribuindo valores fictícios a cada variável, e escolha pesos de acordo com sua necessidade para cada valor de x e b. Considere:

$output \geq 0,5$ - O Festival de Pipas é viável

$output < 0,5$ - O Festival de Pipas é inviável

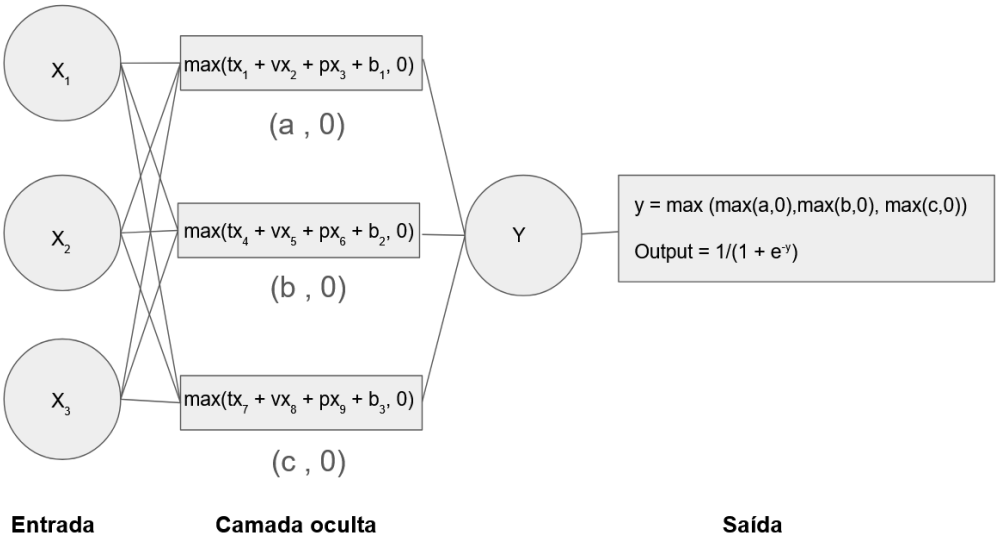


Figura 2 – Modelo de Rede Neural a ser utilizado.
Fonte: Elaborado pelo autor

4. Neste passo teremos a resolução do problema. Assim, os estudantes devem por meio de cálculos de tentativa e erro identificar quando será possível realizar o festival considerando temperatura, probabilidade de chuva e velocidade de vento. É interessante que o professor divida a sala em pequenos grupos, para que possam discutir e testar algumas das possibilidades. Abaixo, temos um exemplo de atribuição de pesos, onde a chuva apresenta valor negativo, uma vez que impede a realização do evento.

Exemplo:

Temperatura: 22.0 °C, Vento = 8 km/h e Chuva=0.2. Considerando

$x_1 = 0,05$, $x_2 = 0,06$, $x_3 = -3,1$ e $b_1 = 0$

$x_4 = 0,04$, $x_5 = 0,07$, $x_6 = -2,8$ e $b_2 = 0$

$x_7 = 0,03$, $x_8 = 0,05$, $x_9 = -2,9$ e $b_3 = 0$

Para $max(tx_1 + vx_2 + px_3 + b_1, 0)$:

max (22 . 0,05 + 8 . 0,06 + 0,2 . (-3,1) + 0 ; 0)

max (1,1 + 0,48 - 0,62 + 0 ; 0)

max (0,96 ; 0) = 0,96

Para max($tx_4 + vx_5 + px_6 + b_2$, 0):

$$max\ (22\,.\,0{,}04\ +\ 8\,.\,0{,}07\ +\ 0{,}2\,.\,(-2{,}8)\ +\ 0\ ;\ 0)$$
$$max\ (0{,}88\ +\ 0{,}56\ -\ 0{,}56\ +\ 0;\ 0)$$
$$max\ (0{,}88\ ;\ 0)\ =\ 0{,}88$$

Para *max*($tx_7 + vx_8 + px_9 + b_3$, 0):

$$max\ (\ 22\,.\,0{,}03\ +\ 8\,.\,0{,}05\ +\ \ 0{,}2\,.\,(-2{,}9)\ +\ 0\ ;\ 0)$$
$$max\ (\ 0{,}66\ +\ 0{,}4\ -\ 0{,}58\ +\ 0;\ 0)$$
$$max\ (0{,}48\ ;\ 0)\ =\ 0{,}48$$

Assim,

$$max\ ((max(0{,}96; 0);\ max\ (0{,}88;\ 0);\ max\ (0{,}48;\ 0))\ =\ 0{,}96$$

Para a função de ativação temos:

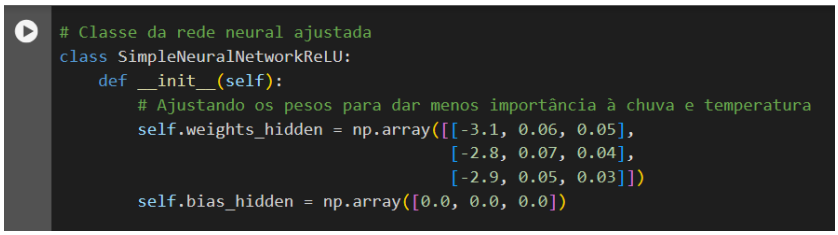
$$output\ =\ \frac{1}{1\ +\ e^{-0{,}96}}$$
$$output\ =\ 0{,}7231$$

5. Neste momento, os alunos devem apresentar alguns dos resultados que encontraram e verificar se os dados encontrados são ou não factíveis com a realização de um Festival de Pipas. Por exemplo, no caso testado anteriormente tivemos $output\ =\ 0{,}7231$. Ou seja, as condições são ideais para a realização do festival de pipas, pois $0{,}7231\ >\ 0{,}5$.

6. Por fim, o professor pode apresentar aos estudantes um exemplo de rede neural em código Python, onde os estudantes podem realizar testes e ajustar os pesos e bias. Deve ficar claro que se trata apenas de um código inicial e que uma rede neural precisa realizar treinamento com uma quantidade significativa de dados para se obter um resultado satisfatório. O código está disponível para acesso em: (https://colab.research.google.com/drive/1YkXcOmXw6wQeYwhWAEgenuKnmFRuJc89?usp=sharing)

Basicamente, o professor deve fornecer o link do Google Colab aos estudantes, ao qual deverá acessar utilizando um computador.

Nas linhas de 15 à 17, estão as matrizes referentes aos pesos e bias, conforme a figura 3, sendo a probabilidade de chuva, velocidade do vento e temperatura respectivamente. Cada linha representa um neurônio, assim podem conter valores diferentes para os pesos. Já na linha 18, seguem as constantes de cada bia. Em nosso exemplo essa linha é representada por valores nulos, porém podem ser modificados de acordo com as necessidades do usuário.



```
# Classe da rede neural ajustada
class SimpleNeuralNetworkReLU:
    def __init__(self):
        # Ajustando os pesos para dar menos importância à chuva e temperatura
        self.weights_hidden = np.array([[-3.1, 0.06, 0.05],
                                        [-2.8, 0.07, 0.04],
                                        [-2.9, 0.05, 0.03]])
        self.bias_hidden = np.array([0.0, 0.0, 0.0])
```

Figura 3: Ajuste de pesos no ambiente Google Colab
Fonte: Elaborado pelo autor

Nas linhas 40 à 61, estão algumas funções (sentenças) para determinar se as condições são ou não ideais para alguns casos, como apresentado na figura 4. Com estas linhas podemos restringir algumas situações indesejáveis como ventos muito fortes, muito fracos, temperaturas muito altas ou baixas, além de limitar a probabilidade máxima para a chuva. Essas condições foram inseridas apenas como sugestões de possibilidades de análise, para que os estudantes e o professor tenham noção de quais condições podem observar. Além disso, é um código bem simples para dar uma previsão exata, sem contar que a rede necessitaria de um treinamento considerável. Porém, essas linhas podem ser excluídas caso o professor ou os estudantes queiram maximizar o uso do código, visto que assim podem verificar mais possibilidades, além do resultado de cada cálculo. Tais linhas podem ser observadas na imagem a seguir e no código mais abaixo.

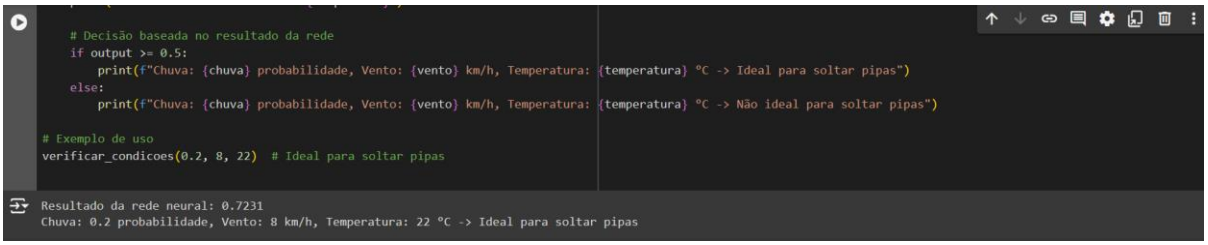
```
# Função para determinar se as condições são ideais para soltar pipa
def verificar_condicoes(chuva, vento, temperatura):
    # Verificar se o vento é superior a 35 km/h
    if vento > 35:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Não ideal para soltar pipas (vento forte)")
        return
    if vento < 8:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Não ideal para soltar pipas (vento fraco)")
        return
    # Verificar se a probabilidade de chuva é superior a 0.5
    if chuva > 0.5:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Não ideal para soltar pipas (alta probabilidade de chuva)")
        return

    # Verificar se a temperatura é abaixo de 15°C ou acima de 30°C
    if temperatura < 15:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Não ideal para soltar pipas (temperatura muito baixa)")
        return

    if temperatura > 30:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Não ideal para soltar pipas (temperatura muito alta)")
        return
```

Figura 4: Ajuste de condições ideais no ambiente Google Colab
Fonte: Elaborado pelo autor

Já nas linhas 71 à 75 (figura 5), a sentença que determina a tomada de decisão da rede, onde $output \geq 5$ equivale a condições ideais para a realização do evento e $output < 5$ não indica condições favoráveis à realização do festival de pipas.



```
    # Decisão baseada no resultado da rede
    if output >= 0.5:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Ideal para soltar pipas")
    else:
        print(f"Chuva: {chuva} probabilidade, Vento: {vento} km/h, Temperatura: {temperatura} °C -> Não ideal para soltar pipas")

# Exemplo de uso
verificar_condicoes(0.2, 8, 22)  # Ideal para soltar pipas
```

```
Resultado da rede neural: 0.7231
Chuva: 0.2 probabilidade, Vento: 8 km/h, Temperatura: 22 °C -> Ideal para soltar pipas
```

Figura 5: Condições de tomada de decisão da rede no ambiente Google Colab
Fonte: Elaborado pelo autor

A linha 78 trás um exemplo de uso, onde tanto os estudante quanto o professor podem realizar testes atribuindo os valores referentes a probabilidade de chuva, velocidade do vento (em km/h) e temperatura (em °C) respectivamente. Note que o exemplo exibido na imagem é o mesmo utilizado anteriormente nos cálculos manuais (0.2 , 8 , 22). Ao substituir esses valores, o usuário deverá clicar em “executar a célula” (simbolizado por um círculo branco com um triângulo interno) localizado na coluna à esquerda do código, ou clicar em alguma área da célula e utilizar o comando Ctrl + Enter. Com essa ação o código será executado e irá determinar as condições para o evento e o resultado numérico obtido pela rede neural. Uma sugestão é que os alunos possam além de testar condições diversas, possam verificar os cálculos manuais por meio do código da rede atribuindo os mesmos pesos e bias utilizados por eles.

Recursos didáticos

Código Python (Google Colab), lousa, televisão, notebook e folhas de rascunho.

Avaliação

A avaliação pode ser dada de forma contínua e por meio de atividades práticas e participativas, envolvendo os cálculos e simulações que os estudantes irão realizar de forma manual e virtual.

Referências Bibliográficas

BARONI, I.; PADILHA, E. J. **A Matemática por Trás das Redes Neurais**. 2021. Centro Universitário Internacional, UNINTER. Acesso em 3 de março de 2024. Disponível em: <https: //repositorio.uninter.com/bitstream/handle/1/1023/ITANIM%c3%81%20BARONI%20-%20RU%202607819.pdf?sequence=1&isAllowed=y>.

COZMAN, F. G.; PLONSKI, G. A.; NERI, H. (Ed.). **Inteligência Artificial: Avanços e Tendências.** São Paulo: IEA e C4AI-USP, 2021. Disponível em: <https://www.livrosabertos.sibi.usp.br/portaldelivrosUSP/catalog/download/650/579/2181?inline=1>.

FREITAS, A. T. **Neurónio - Elemento de Processamento.** 2020. Instituto Superior Técnico, Universidade de Lisboa. Acesso em 03 de março de 2024. Disponível em: <http://web.tecnico.ulisboa.pt/ana.freitas/bioinformatics.ath.cx/bioinformatics.ath.cx/indexb3a3.html?id=110>.

SICHMAN, J. **Inteligência artificial e sociedade: avanços e riscos.** Estudos Avançados, v. 35, n. 101,p.37–50, abril 2021. Disponível em: <https://www.revistas.usp.br/eav/article/view/185024>.

Anexo: Código Python da Rede Neural na íntegra

```
import numpy as np

# Função de ativação ReLU
def relu ( x):
    return np. maximum (0 , x)

# Função de ativação Sigmoide
def sigmoid ( x):
    return 1 / (1 + np. exp (- x))

# Classe da rede neural ajustada
class Simple NeuralNetwork Re LU :
    def __init__ ( self):
        # Ajustando os pesos para dar menos importância à chuva e temperatura
        self. weights_hidden = np. array ([[ -3.1 , 0.06 , 0.05] ,
                                           [-2.8 , 0.07 , 0.04] ,
                                           [-2.9 , 0.05 , 0 .03 ]])
        self. bias_hidden = np. array ([0.0 , 0.0 , 0.0])

    def forward ( self , inputs):
        # Camada oculta
        net_hidden = np. dot( self. weights_hidden , inputs) + self. bias_hidden
        output_hidden = relu ( net_hidden )

        # Pegando o valor má ximo da camada oculta
        max_hidden = np. max ( output_hidden )

        # Aplicando a fun ção sigmoide no valor máximo da camada oculta
        output_final = sigmoid ( max_hidden )

        return output_final

# Função para realizar simulações
def simulate ( chuva , vento , temperatura ):
    nn = Simple NeuralNetwork Re LU ()
    inputs= np. array ([ chuva , vento , temperatura ])
    output= nn. forward ( inputs)
    returnoutput

# Função para determinar se as condições são ideais para soltar pipa
```

```
def verificar_condicoes ( chuva , vento , temperatura ):
    # Verificar se o vento é superior a 35 km/ h
    if vento > 35:
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura :
{temperatura } °C -> Não ideal para soltar pipas ( vento forte )")
        return
    if vento < 8:
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura :
{temperatura } °C -> Não ideal para soltar pipas ( vento fraco )")
        return
    # Verificar se a probabilidade de chuva é superior a 0.5
    if chuva > 0.5:
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura :
{temperatura } °C -> Não ideal para soltar pipas ( alta probabilidade de
chuva )")
        return

    # Verificar se a temperatura é abaixo de 15 °C ou acima de 30 C
    if temperatura < 15:
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura :
{temperatura } °C -> Não ideal para soltar pipas ( temperatura muito baixa
)")
        return

    if temperatura > 30:
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura :
{temperatura } °C -> Não ideal para soltar pipas ( temperatura muito alta
                                                                    )")
        return

    # Executa a rede neural para calcular a probabilidade
    nn = Simple NeuralNetwork Re LU ()
    inputs = np. array ([ chuva , vento , temperatura ])
    output = nn. forward ( inputs)

    # Exibe o resultado da rede neural
    print( f" Resultado da rede neural: { output :.4 f}")

    # Decisão baseada no resultado da rede
    if output >= 0.5:
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura : {
temperatura } °C -> Ideal para soltar pipas")
    else :
        print( f" Chuva : { chuva } probabilidade , Vento : { vento } km/ h, Temperatura : {
temperatura } °C -> Não ideal para soltar pipas")
```

```
76
77 # Exemplo de uso
78 verificar_condicoes (0.2 , 8 , 22)          # Ideal para soltar pipas
```