Lista de Exercícios – Rede Neural de Camada Simples

Tema: Aplicações regionais do Amazonas
Pré-requisitos: Python básico, funções, listas, loops, lógica condicional, funções de ativação (`relu`, `sigmoid`, `degrau`, `tanh`)

Exercício 1 – Previsão de Cheia com Neurônio Simples

No interior do Amazonas, ribeirinhos monitoram a cheia dos rios todos os anos. Suponha que a entrada seja a **quantidade de chuvas** nos últimos 3 dias (em mm).

Implemente um neurônio com **pesos definidos manualmente**, somatório ponderado e a **função de ativação ReLU**, para prever um **índice de risco de cheia**.

Requisitos:

- Entrada: 3 valores (chuvas dos últimos 3 dias)
- Pesos: `[0.3, 0.5, 0.2]`
- Use função `relu(x)` definida por você
- Saída: índice de risco de cheia (quanto maior, maior o risco)

```
# Exemplo de entrada: [120, 150, 80]
```

Exercício 2 – Classificação de Qualidade da Água com Função Degrau

Ribeirinhos costumam avaliar a qualidade da água observando sinais visuais simples. Crie um **neurônio que classifica a água como própria (1) ou imprópria (0)** com base em 3 entradas:

- Turbidez (0 a 10)
- Odor (0 a 10)
- Presença de lixo visível (0 a 1)

Use pesos definidos e uma **função de ativação degrau** para classificar.

Requisitos:

- Pesos: `[0.6, 0.3, 0.9]`
- Limite de decisão: `>= 0.7` → água imprópria
- Use a função degrau `step(x)`
- Entrada: valores reais entre 0 e 1 (normalizados)

```
# Exemplo: [0.5, 0.4, 1]
```

Exercício 3 – Previsão de Atividade de Peixes com Sigmoid

No Amazonas, pescadores artesanais utilizam conhecimento empírico sobre **temperatura da água**, **nível do rio**, e **horário do dia** para prever o melhor momento para pesca.

Crie um neurônio simples que calcule a **probabilidade de alta atividade de peixes** usando a **função sigmoid** como ativação.

Requisitos:

- Entrada: temperatura (°C), nível do rio (m), hora do dia (0-23)
- Pesos: `[0.4, 0.6, -0.1]`
- Use função `sigmoid(x)`
- Saída: probabilidade (0 a 1)

```
# Exemplo: [29, 8, 6]
```

Exercício 4 – Monitoramento de Temperatura de Plantação com Tanh

Na região do Alto Solimões, agricultores cultivam mandioca e monitoram a temperatura do solo para saber se a plantação está sob risco de estresse térmico.

Crie um neurônio simples com ativação `tanh`, que recebe:

- Temperatura atual do solo (em °C)
- Umidade relativa (em %)
- Hora do dia

Requisitos:

- Pesos: `[0.3, 0.2, -0.4]`
- Use função `tanh(x)`
- A saída pode variar de -1 a 1:
    - próximo de -1: muito frio
    - próximo de 1: muito quente
    - próximo de 0: ideal

```
# Exemplo: [35, 70, 14]
```

Observações:

- Todos os neurônios devem ser implementados **manualmente** (sem frameworks).
- Incentive os alunos a criarem suas próprias **funções de ativação** em Python.
- Estimule o uso de `input()` para que testem com seus próprios dados locais.

Gabarito Comentado – Exercícios de Rede Neural de Camada Simples

---

Exercício 1 – Previsão de Cheia com ReLU

```
import numpy as np

def relu(x):
    return np.maximum(0, x)

entradas = np.array([120, 150, 80])  # chuvas em mm
pesos = np.array([0.3, 0.5, 0.2])

soma = np.dot(entradas, pesos)
print("Soma ponderada:", soma)

saida = relu(soma)
```

```
print("Índice de risco de cheia:", saida)

# Interpretação da saída
if saida < 100:
    print("Situação normal: baixo risco de cheia.")
elif saida < 150:
    print("Atenção: risco moderado de cheia.")
else:
    print("Alerta: alto risco de cheia! População deve ser
avisada.")
```

Explicação:

- O código simula um **único neurônio** que soma os valores de entrada multiplicados pelos pesos.
- A função `relu(x)` garante que a saída seja **0 ou positiva**, pois não faz sentido ter risco de cheia negativo.
- O resultado final é um número que representa o **nível de risco de cheia**.

Exercício 2 – Classificação de Qualidade da Água com Função Degrau

```
import numpy as np

def degrau(x):
    return 1 if x >= 0.7 else 0

entradas = np.array([0.5, 0.4, 1.0])  # turbidez, odor, lixo visível
pesos = np.array([0.6, 0.3, 0.9])

soma = np.dot(entradas, pesos)
print("Soma ponderada:", soma)

saida = degrau(soma)
print("Classificação (1 = imprópria, 0 = própria):", saida)

# Interpretação
if saida == 1:
    print("Água imprópria para consumo. Necessário tratamento ou
buscar nova fonte.")
else:
```

```
    print("Água própria para consumo imediato.")
```

Explicação:

- A função `degrau(x)` retorna 1 se o valor for maior ou igual a 0.7, indicando **água imprópria**.

- A saída é binária (0 ou 1), usando o limiar `0.7` como critério.
- O modelo classifica automaticamente com base em critérios simples e **simula um sistema de triagem automatizada local**.

- Os pesos refletem a **importância de cada fator** na decisão.

Exercício 3 – Previsão de Atividade de Peixes com Sigmoid

```
import numpy as np

def sigmoid(x):
    return 1 / (1 + np.exp(-x))

entradas = np.array([29, 8, 6])  # temperatura, nível do rio, hora
pesos = np.array([0.4, 0.6, -0.1])

soma = np.dot(entradas, pesos)
print("Soma ponderada:", soma)

saida = sigmoid(soma)
print("Probabilidade de alta atividade de peixes:", saida)

# Interpretação da saída
if saida > 0.75:
    print("Excelente momento para pesca! Alta atividade prevista.")
elif saida > 0.5:
    print("Bom momento para pesca.")
elif saida > 0.3:
    print("Atividade de peixes moderada. Pesca possível, mas menos
eficiente.")
else:
    print("Baixa atividade de peixes. Melhor aguardar outro
horário.")
```

Explicação:

- O `np.exp()` é usado para o cálculo do `e^(-x)` da função sigmoid.
- A saída pode ser interpretada como uma **chance entre 0 e 1**.
-
- Se a saída estiver próxima de 1, é **um bom momento para pesca**.
- A **hora do dia** tem peso negativo, indicando que horas mais altas podem ser menos propícias.

Exercício 4 – Monitoramento da Temperatura com `tanh`

```
import numpy as np

# Função tanh
def tanh(x):
    return np.tanh(x)

# Entradas: temperatura do solo, umidade, hora do dia
entradas = np.array([35, 70, 14])

# Pesos
pesos = np.array([0.3, 0.2, -0.4])

# Soma ponderada
soma = np.dot(entradas, pesos)
print("Soma ponderada:", soma)

# Aplicação da tanh
saida = tanh(soma)
print("Índice térmico:", saida)

# Interpretação da saída
if saida < -0.5:
    print("Alerta: solo muito frio!")
elif saida > 0.5:
    print("Alerta: solo muito quente!")
else:
    print("Temperatura do solo ideal.")
```

Explicação:

- A `np.tanh(x)` retorna valores entre -1 e 1.

- A interpretação final mostra **como um modelo simples pode gerar alertas automatizados.**
- Valores extremos indicam **condições não ideais** para a mandioca.

- O uso do `if` ajuda os alunos a praticar **interpretação da saída** e **ações automatizadas**.