# Introdução ao aprendizado de máquina 5

Aula 5- Redes Neurais

# Aplicações de redes neurais

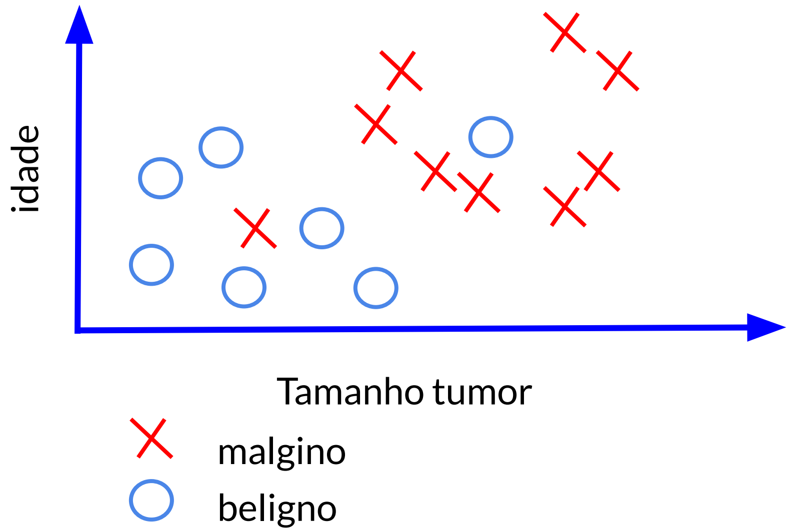
1. Classificação de imagens
2. Reconhecimento de dígitos
3. Perguntas sobre imagens
4. Reconhecimento de fala
5. Modelos de processamento de linguagem natural
6. Carros autônomos
7. Modelos de previsão de séries temporais

# Por que regressão logística não basta?

- Para treinar uma regressão logística com N características e todas combinações  de segunda ordem, precisamos treinar $O(n^2)$ parâmetros

- Uma foto colorida de 50 pixels tem 50x50x3= 7500 parâmetros iniciais

- Essa abordagem fica computacionalmente complicada muito rapidamente



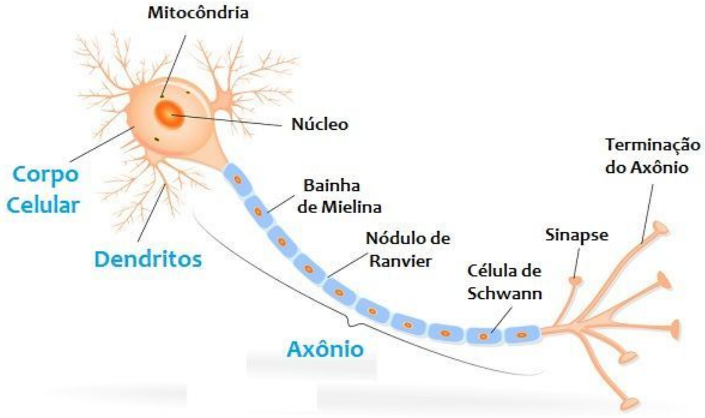
# Um paralelo com biologia

# Origens de redes neurais

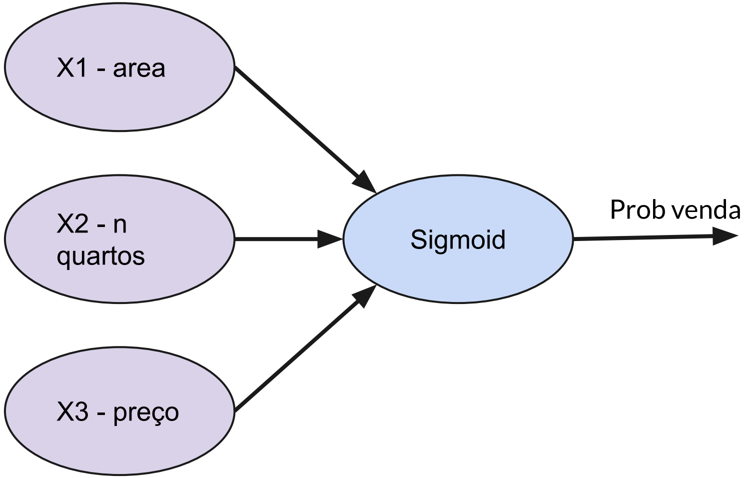
- Redes neurais foram inspiradas pela estrutura do cérebro humano
- Um neurônio humano é uma de aprendizado geral
- Foi vastamente usada nos anos 80 e no começo da década de 90, mas ao término da década de 90 caiu em desuso
- Melhores computadores e mais dado levarem ao renascimento de redes neurais.
- Atualmente, a maioria das técnicas de fronteira usa redes neurais

# Representação de um Neurônio

Neurônio

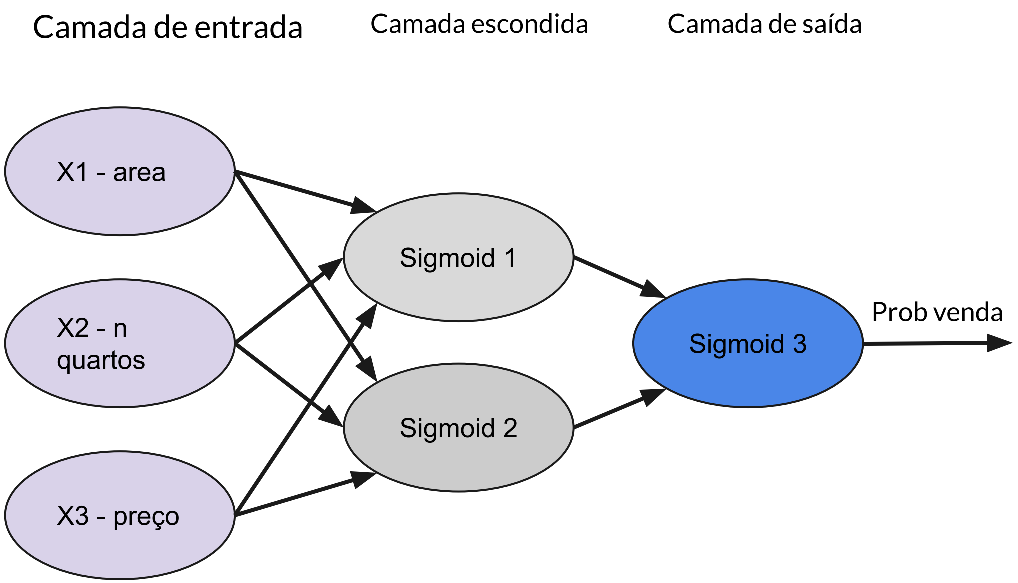



Regressão logística



# Um Exemplo

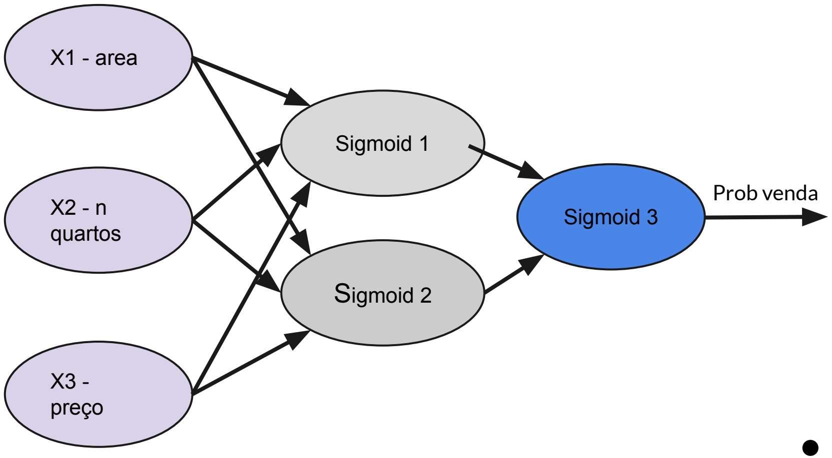
# Exemplo de redes neurais




- Chamados o resultado de cada camada $a_i$
- $\theta_{01}$ é o parâmetro associado ao input 0 na sigmoid 1
- Quantos parâmetros cada sigmoid terá?

# Matematica das redes neurais

Camada de entrada Camada escondida Camada de saída

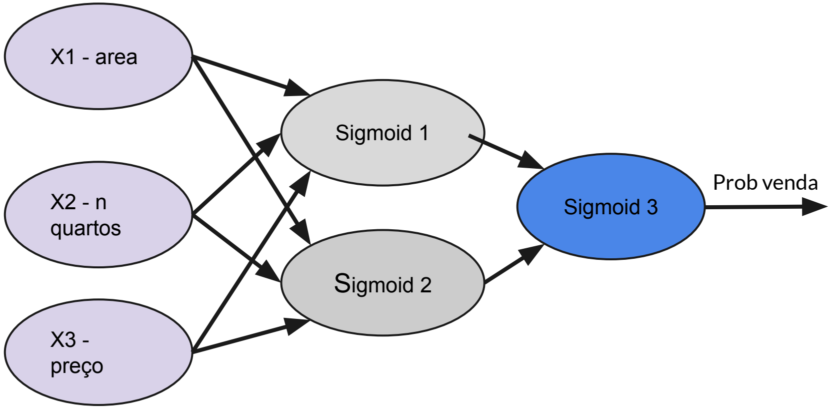



$$a_1 = \Omega\left(\theta_{10} + \theta_{11} x_1 + \theta_{12} x_2 + \theta_{13} x_3\right)$$
$$a_2 = \Omega\left(\theta_{20} + \theta_{21} x_1 + \theta_{22} x_2 + \theta_{23} x_3\right)$$
$$a_3 = \Omega\left(\theta_{30} + \theta_{31} a_1 + \theta_{32} a_2\right)$$
$$\hat{y} = a_3$$

- Precisamos iniciar os parâmetros aleatoriamente para quebrar simetria!!

# Propagação para frente e para trás

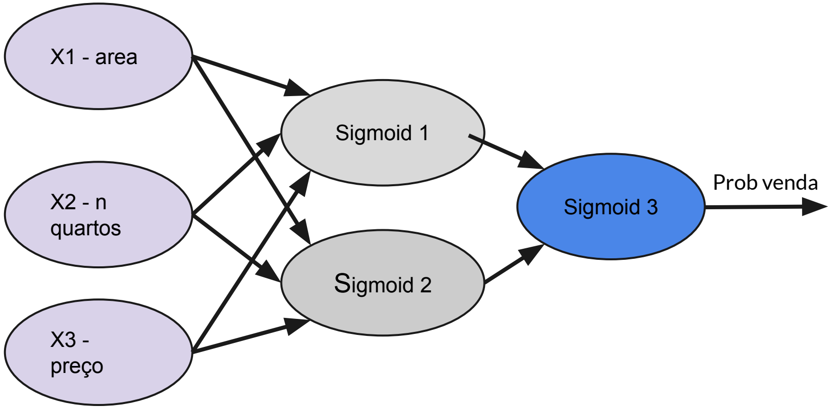
# Propagação para frente




$$a_1 = \Omega\,(\theta_{10} + \theta_{11}\,x_1 + \theta_{12}\,x_2 + \theta_{13}\,x_3)$$
$$a_2 = \Omega\,(\theta_{20} + \theta_{21}\,x_1 + \theta_{22}\,x_2 + \theta_{23}\,x_3)$$
$$a_3 = \Omega\,(\theta_{30} + \theta_{31}\,a_1 + \theta_{32}\,a_2)$$
$$\hat{y} = a_3$$

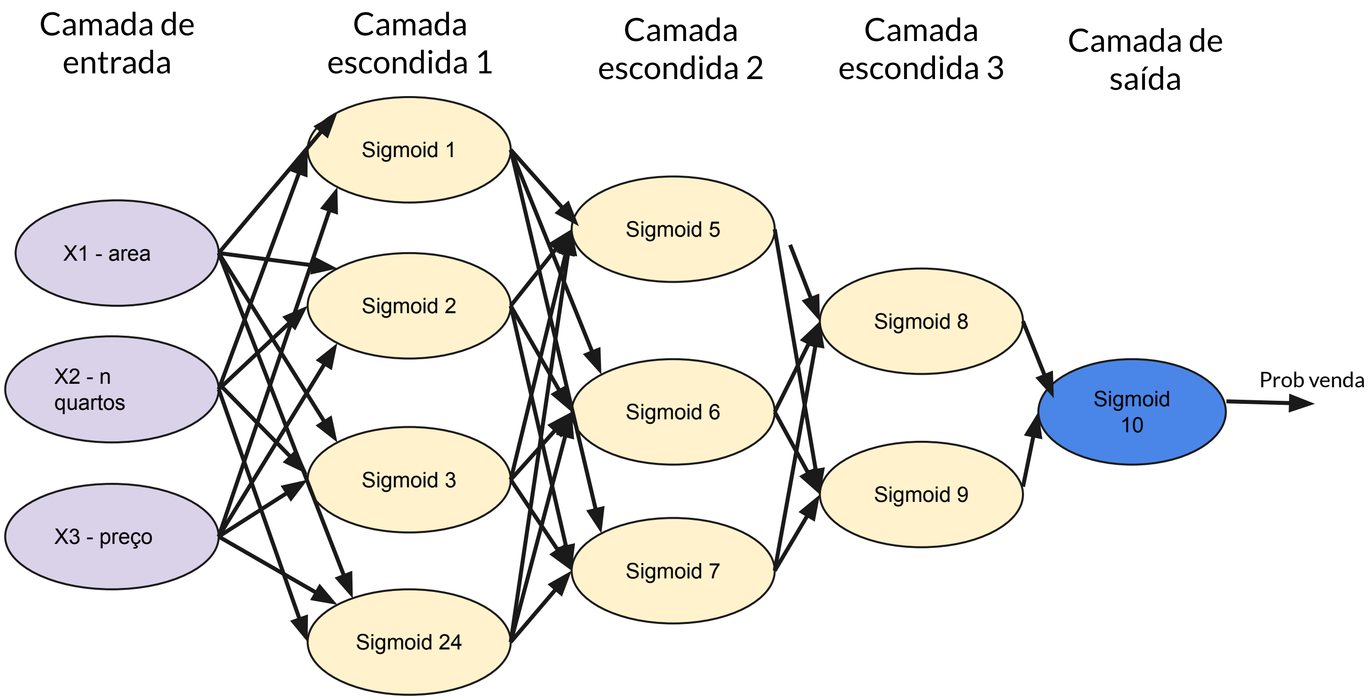
$$A = \Omega(\Theta_{12}\,X)$$
$$\hat{Y} = \Omega(\Theta_3\,A)$$

# Propagação para trás




$$a_1 = \Omega\left(\theta_{10} + \theta_{11} x_1 + \theta_{12} x_2 + \theta_{13} x_3\right)$$
$$a_2 = \Omega\left(\theta_{20} + \theta_{21} x_1 + \theta_{22} x_2 + \theta_{23} x_3\right)$$
$$a_3 = \Omega\left(\theta_{30} + \theta_{31} a_1 + \theta_{32} a_2\right)$$
$$\hat{y} = a_3$$

# Redes mais complicadas



# Resolvendo com método do gradiente

# Calculando os gradientes

$$a_1 = \Omega\left(\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2\right)$$
$$a_2 = \Omega\left(\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2\right)$$
$$a_3 = \Omega\left(\theta_{30} + \theta_{31}a_1 + \theta_{32}a_2\right)$$
$$\hat{y} = a_3$$
$$\Omega(z) = \frac{1}{1+e^{-z}}$$
$$J(\theta) = -\sum\left(y_i \log(\hat{y}) + (1 - y_i)\log(1 - \hat{y})\right)$$

# Calculando os gradientes

$$a_1 = \Omega\left(\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2\right)$$
$$a_2 = \Omega\left(\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2\right)$$
$$a_3 = \Omega\left(\theta_{30} + \theta_{31}a_1 + \theta_{32}a_2\right)$$
$$\hat{y} = a_3$$
$$\Omega(z) = \frac{1}{1+e^{-z}}$$
$$J(\theta) = -\sum\left(y_i \log(\hat{y}) + (1 - y_i)\log(1 - \hat{y})\right)$$

# Teorema da aproximação universal

# Teorema da aproximação universal

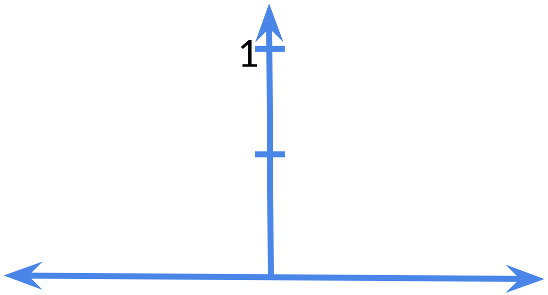
**Teorema:** "Uma rede neural com uma simples camada escondida pode aproximar arbitrariamente bem, qualquer função contínua"

Esse resultado é poderoso, mas não diz nada sobre o número de neurônios nas camadas os como calcular os parâmetros.

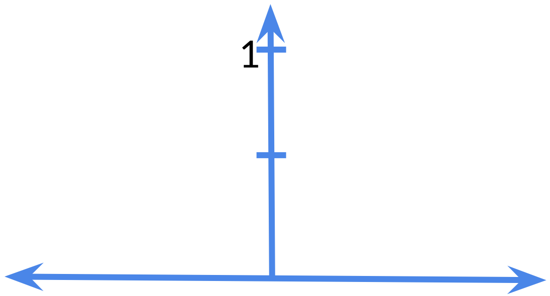
Isso quer dizer que as redes neurais de fronteira têm apenas uma camada? Não

# Outras funções de ativação

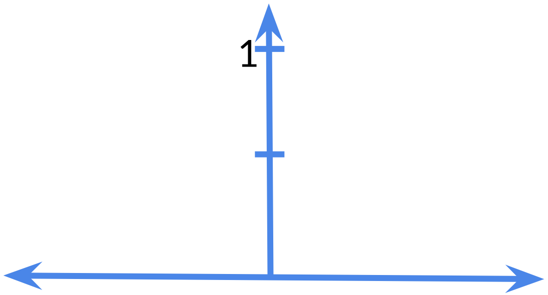
Relu




Tangente hiperbólica




Função degrau



# Interpretação de um neurônio

# Interpretação de parâmetros em outros modelos

$$\text{Regressão:}\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

$$\text{Logistica:}\hat{y} = \Omega\left(\gamma_0 + \gamma_1 x_1 + \gamma_2 x_2\right)$$

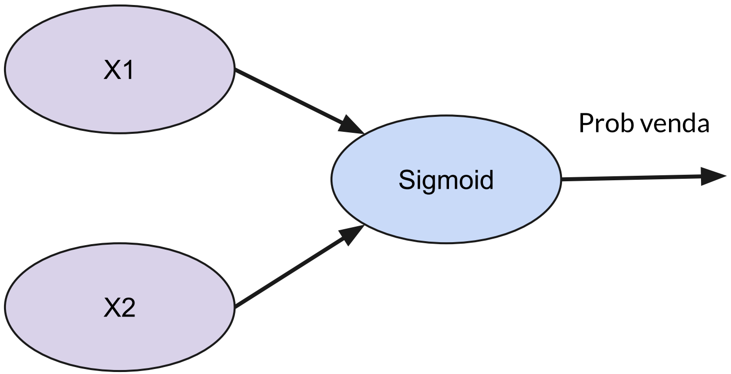
Em redes neurais a interpretação é ainda mais difícil

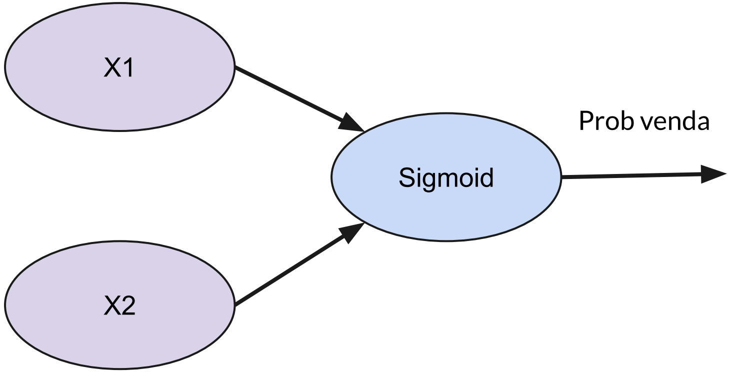
# Exemplo: “e” e “ou” redes

$$x = 1 \text{ or } x = 0$$

$$y = \Omega\left(-50 + 30x_1 + 30x_2\right)$$

$$y = \Omega\left(-20 + 30x_1 + 30x_2\right)$$



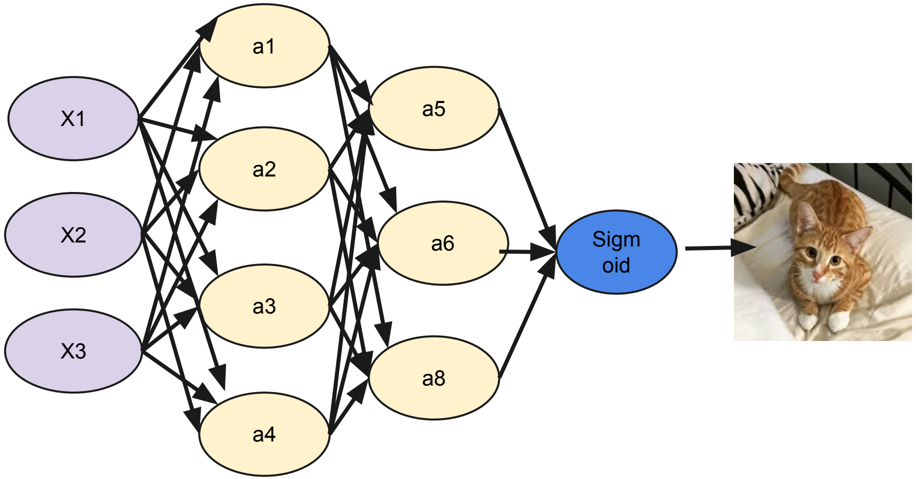



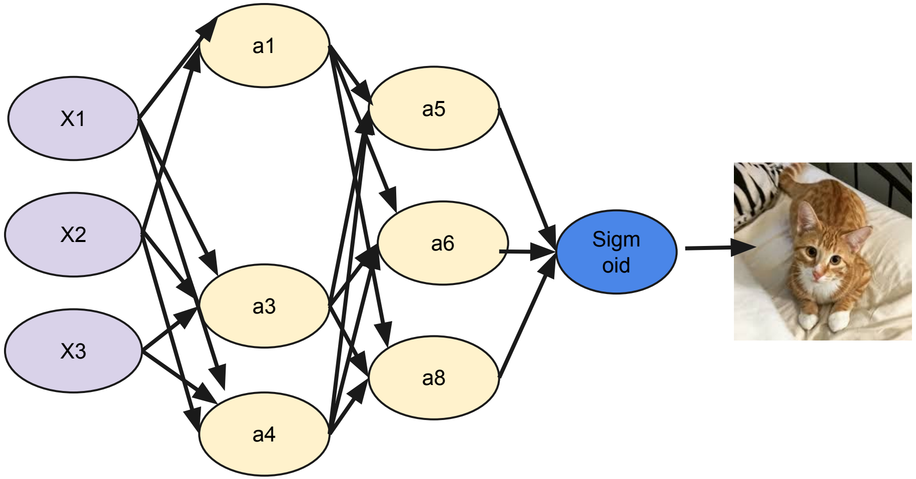
# O que um neurônio vê?

**Edges** (layer conv2d0) **Textures** (layer mixed3a) **Patterns** (layer mixed4a) **Parts** (layers mixed4b & mixed4c) **Objects** (layers mixed4d & mixed4e)

# Dropout (abandono)

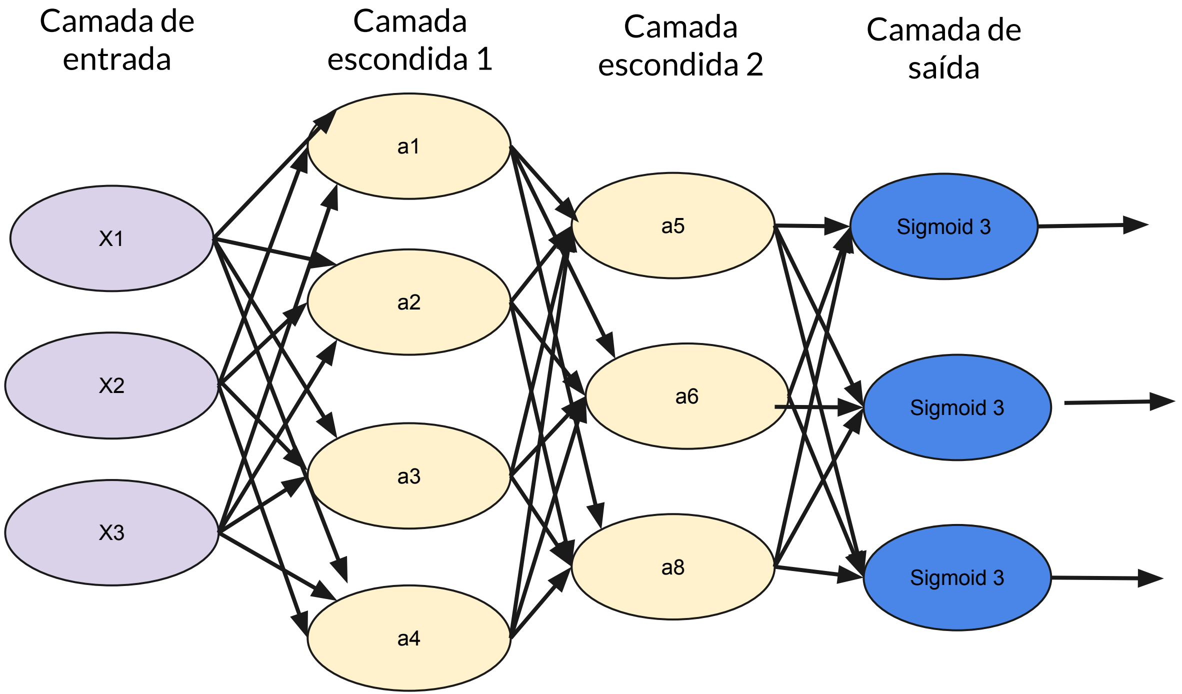
# Regularização: dropout

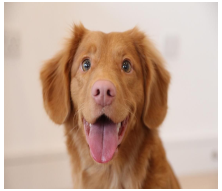


# Multiplas classes e reutilizando o aprendizado

# Exemplos com múltiplas classes



# Transferir o aprendizado