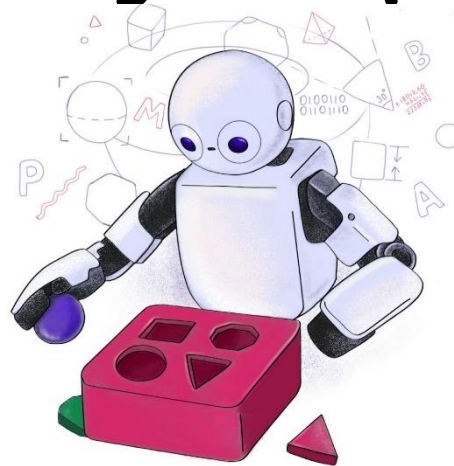


# T320 - Introdução ao Aprendizado de Máquina II: *Classificação (Parte IV)*



**Inatel**

Felipe Augusto Pereira de Figueiredo  
felipe.figueiredo@inatel.br

# Recapitulando

- Anteriormente, aprendemos uma nova ***função de limiar***, chamada de ***função logística***, com a qual foi possível se encontrar uma solução com o algoritmo do ***gradiente descendente***.
- Classificadores que utilizam a ***função logística*** como ***função de limiar*** são conhecidos como ***regressores logísticos*** e são utilizados em problemas de ***classificação binária***, ou seja, problemas com 2 classes apenas, após a discretização do valor de saída.
- Na sequência, veremos como lidar com problemas de classificação que envolvem mais de 2 classes, também chamados de ***classificação multi-classes***.

# Casos multi-classe

- Até agora, nós vimos como classificar utilizando ***regressão logística*** quando os dados pertencem a apenas 2 classes (i.e.,  $Q = 2$ ), mas e quando o problema possui mais de 2 classes (i.e.,  $Q > 2$ )? Por exemplo
  - Reconhecimento de dígitos escritos à mão: 10 dígitos.
  - Classificação de texto: Esportes, Economia, Política, Entretenimento, etc.
  - Classificação de sentimentos: Neutro, Positivo, Negativo.
- Existem algumas abordagens para a ***classificação multi-classe***:
  - Um-Contra-o-Resto
  - Um-Contra-Um
  - Regressão Softmax
- As duas primeiras podem ser aplicadas a qualquer tipo de ***classificador binário*** e não apenas ao ***regressor logístico***.
- A terceira abordagem é uma generalização do ***classificador logístico*** para problemas multi-classe.

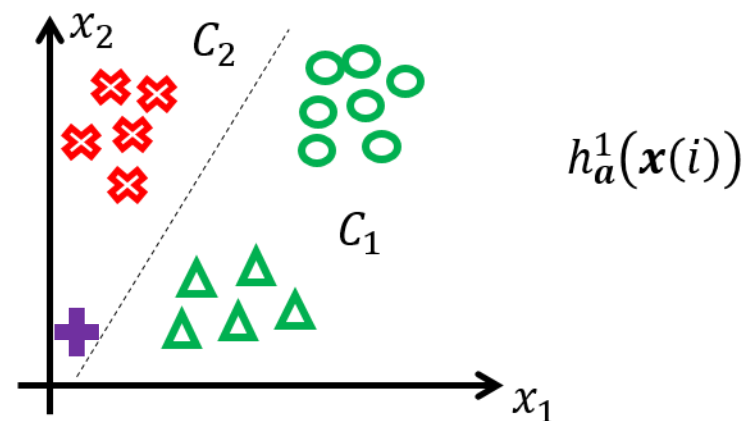
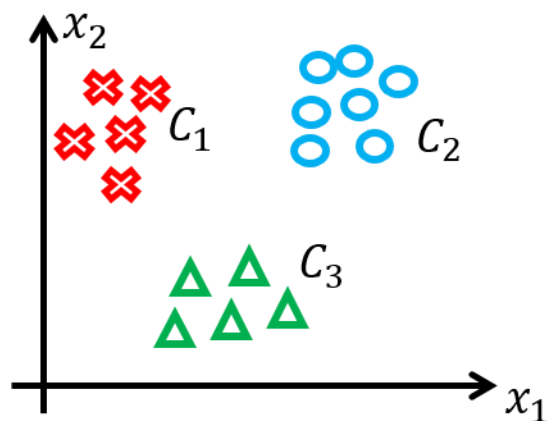
# Um-Contra-o-Resto

- Nesta abordagem, nós treinamos um **classificador binário** (e.g., **regressor logístico**), representado pela função hipótese,  $h_a^q(\mathbf{x}(i))$ , para cada classe  $q$  para predizer a probabilidade de  $\hat{y} = q$ , ou seja,  $P(\hat{y} = q | \mathbf{x}(i); \mathbf{a})$ .
- Em outras palavras, cria-se  $Q$  **classificadores binários**, onde para cada classificador, a classe positiva  $C_2 = q$  e a classe negativa  $C_1$  é a junção de todas as outras  $Q - 1$  classes.
- Portanto, o **classificador** deve indicar a classe positiva caso o exemplo pertença à classe  $q$ , e a classe negativa caso o exemplo pertença a qualquer outra classe.
- Para cada novo exemplo de entrada,  $\mathbf{x}(i)$ , realiza-se as predições e escolhe-se a classe que maximize

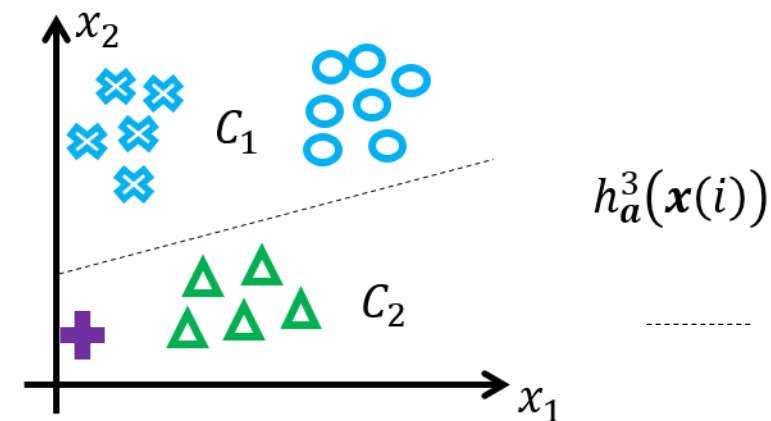
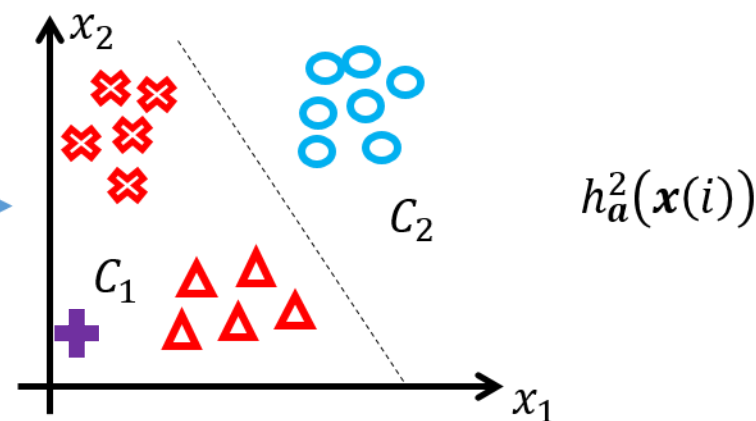
$$C_q = \arg \max_q h_a^q(\mathbf{x}(i)).$$

- A vantagem desta abordagem é que se treina apenas  $Q$  **classificadores**.
- A desvantagem é que cada **classificador binário** precisa ser treinado com um conjunto negativo que é  $Q-1$  vezes maior, o que pode aumentar o tempo de treinamento.

# Um-Contra-o-Resto



*A qual classe  
pertence  $\text{+}$  ?*



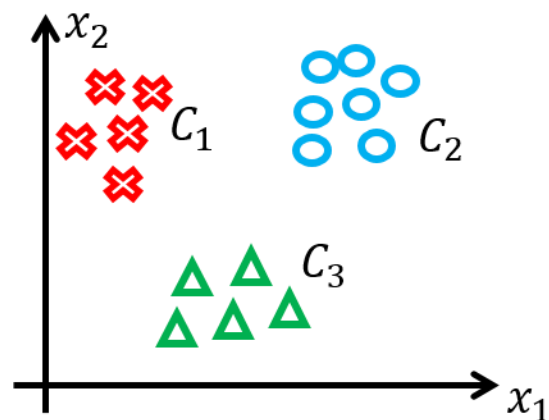
----- Fronteira de decisão

Exemplo de validação

# Um-Contra-Um

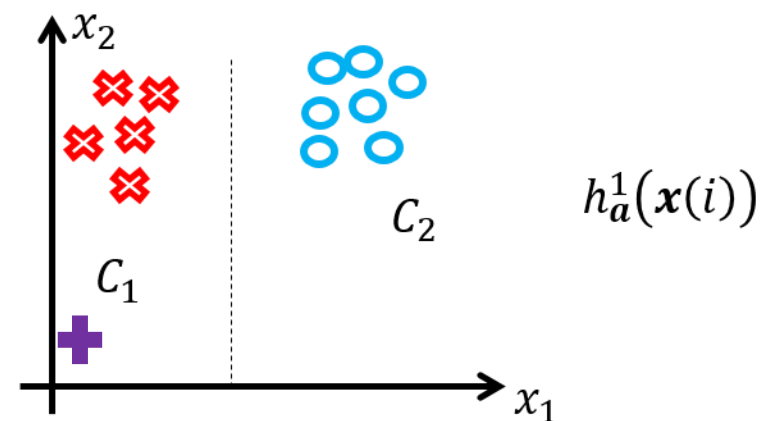
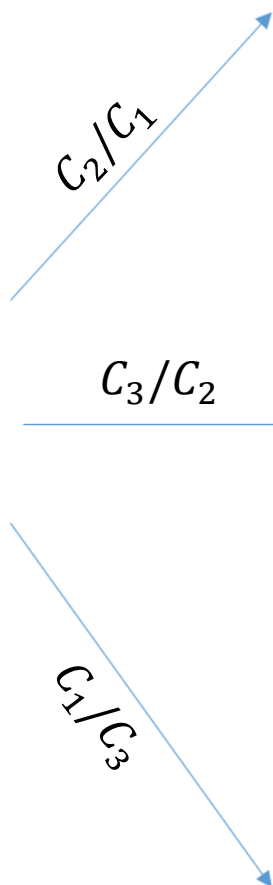
- Nesta abordagem, treina-se  $Q(Q - 1)/2$  **classificadores binários**.
- Cada **classificador** é construído para fazer a distinção entre exemplos pertencentes a cada um dos possíveis **pares** de classes.
  - Se  $Q = 4$ , então treina-se 6 **classificadores** para classificar entre  $C_1/C_2$ ,  $C_1/C_3$ ,  $C_1/C_4$ ,  $C_2/C_3$ ,  $C_2/C_4$ , e  $C_3/C_4$ .
- No final, cada exemplo é classificado conforme o **voto majoritário** entre os **classificadores**.
- A principal vantagem da abordagem **Um-Contra-Um** é que cada **classificador** precisa ser treinado apenas com as duas classes que ele deve distinguir.
- A desvantagem é que, por exemplo, se  $Q = 10$ , temos que treinar 45 **classificadores**.

# Um-Contra-Um

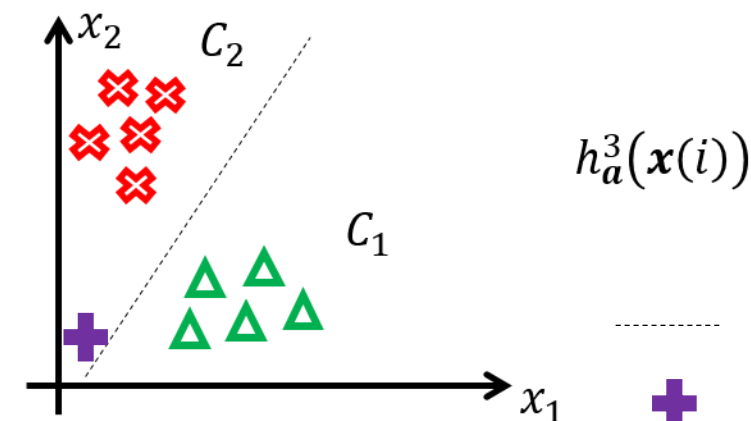
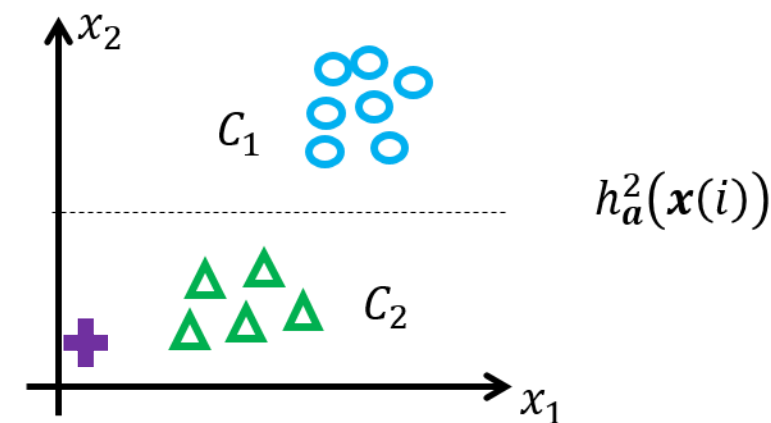


$$Q = 3$$
$$\frac{Q(Q-1)}{2} = 3$$

[Exemplo: ClassificationOfFourClassesWithOvAandOvO.ipynb](#)



*A qual classe  
pertence + ?*



----- Fronteira de decisão  
+ Exemplo de validação

# Regressão Softmax

- Também conhecida como ***regressão logística multinomial***.
  - Pois as saídas do regressor podem ser interpretadas como as probabilidades de uma variável categoricamente distribuída (as classes) dado um conjunto de variáveis (atributos e pesos).
- É uma generalização do regressor logístico para problemas com ***múltiplas classes***.
- A ideia é treinar um ***único*** classificador com  $Q > 2$  saídas, onde cada saída representa a ***probabilidade*** de um exemplo pertencer a uma das  $Q$  classes.
  - Por exemplo, para um problema com 4 classes, teríamos um único classificador, mas com 4 saídas.
- Prediz ***apenas uma classe por classificação***, portanto, ele deve ser usado apenas com ***classes mutuamente exclusivas*** como por exemplo diferentes tipos de plantas, dígitos, carros, etc.
  - ***Classes mutuamente exclusivas***: exemplos pertencem a apenas uma das  $Q$  classes.
  - Já notícias e animais, por exemplo, podem pertencer a várias a várias classes.
- Para termos um ***único*** classificador, o ***regressor softmax*** possui uma ***função hipótese de classificação***,  $h_a^q(x(i))$ , e uma ***função discriminante***,  $g_q(x(i))$ , para cada classe  $q$ .



# Regressão Softmax

- A **função hipótese de classificação** associada à  $q$ -ésima classe,  $h_a^q(\mathbf{x}(i))$ , é obtida passando-se a **função discriminante** da  $q$ -ésima classe,  $g_q(\mathbf{x}(i))$ , através da **função softmax**,

Cada função discriminante tem seu próprio vetor de pesos.

$$P(C_q | \mathbf{x}(i), \mathbf{a}_q) = h_a^q(\mathbf{x}(i)) = \frac{e^{g_q(\mathbf{x}(i))}}{\sum_{j=1}^Q e^{g_j(\mathbf{x}(i))}} = \frac{e^{\mathbf{x}(i)^T \mathbf{a}_q}}{\sum_{j=1}^Q e^{\mathbf{x}(i)^T \mathbf{a}_j}} \in \mathbb{R} [0,1],$$

O somatório de termos exponenciais normaliza o valor da  $q$ -ésima saída de tal forma que o somatório das  $Q$  saídas seja igual a 1.

onde  $\mathbf{a}_q = [a_0^q \ a_1^q \ \dots \ a_K^q]^T \in \mathbb{R}^{K+1 \times 1}$  é o **vetor de pesos** da  $q$ -ésima **função discriminante**,  $g_q(\mathbf{x}(i))$ , e  $i$  indica o número da amostra.

- Assim como com o regressor logístico, podemos usar equações de **hiperplanos** ou **polinomiais** como **funções discriminantes**.
- Assim, a **função softmax** estende a ideia do **regressor logístico** ao mundo multi-classes.
- Ou seja, a **função softmax** atribui uma **probabilidade condicional**,  $P(C_q | \mathbf{x}(i), \mathbf{a}_q)$ , a cada classe,  $q$ , em um problema com múltiplas classes ( $Q > 2$ ), onde a soma destas  $Q$  probabilidades deve ser igual a 1

$$P(C_1 | \mathbf{x}(i), \mathbf{a}_1) + P(C_2 | \mathbf{x}(i), \mathbf{a}_2) + \dots + P(C_Q | \mathbf{x}(i), \mathbf{a}_Q) = 1.$$

# Regressão Softmax

- Portanto, o objetivo é encontrar um **modelo** (i.e., os **pesos** das  $Q$  funções hipótese) que atribua uma alta probabilidade para a classe alvo e consequentemente uma baixa probabilidade para as demais classes.
- Assim como fizemos anteriormente, precisamos definir uma **função de erro** e **minimizá-la** para encontrarmos os **pesos** das  $Q$  **funções hipótese** do classificador softmax.

- A **função de erro médio** para a **regressão softmax** é dada por

$$J_e(\mathbf{A}) = -\frac{1}{N} \sum_{i=0}^{N-1} \sum_{q=1}^Q 1\{y(i) + 1 == q\} \log(h_a^q(\mathbf{x}(i))),$$

O erro tende a 0 quando  $h_a^q(\mathbf{x})$  tende a 1, caso contrário, o erro aumenta.

onde  $1\{\cdot\}$  é a **função indicadora**, de modo que  $1\{\text{uma condição verdadeira}\} = 1$  e  $1\{\text{uma condição falsa}\} = 0$ ,  $\mathbf{A} \in \mathbb{R}^{K+1 \times Q}$  é a matriz com os **pesos** para todas as **funções hipótese** das  $Q$  classes e  $y(i)$  é o  $i$ -ésimo valor esperado.

- A matriz  $\mathbf{A}$  contém em suas colunas os vetores de pesos,  $\mathbf{a}_q$ , de cada uma das  $Q$  **funções discriminantes**.

# Regressão Softmax

- Usando-se a codificação **one-hot**, a equação anterior pode ser re-escrita como

$$J_e(\mathbf{A}) = -\frac{1}{N} \sum_{i=0}^{N-1} \mathbf{y}(i)^T \log \left( \mathbf{h}_a(\mathbf{x}(i)) \right),$$

onde  $\mathbf{y}(i) = [1\{y(i) + 1 == 1\}, \dots, 1\{y(i) + 1 == Q\}]^T \in \mathbb{R}^{Q \times 1}$  é o vetor utilizando a codificação **one-hot** e  $\mathbf{h}_a(\mathbf{x}(i)) \in \mathbb{R}^{Q \times 1}$  é o vetor com as saídas das  $Q$  **funções hipóteses**

$$\begin{aligned} \mathbf{h}_a(\mathbf{x}(i)) &= [h_a^1(\mathbf{x}(i)), \dots, h_a^Q(\mathbf{x}(i))]^T \\ &= [P(C_1 \mid \mathbf{x}(i), \mathbf{a}_1), \dots, P(C_Q \mid \mathbf{x}(i), \mathbf{a}_Q)]^T. \end{aligned}$$

- Notem que quando existem apenas duas classes ( $Q = 2$ ), a **função de erro** acima é equivalente à **função de erro** do **regressor logístico**.
- Ou seja, mesmo tendo sido pensado para caso onde  $Q > 2$ , o regressor softmax pode ser usado quando  $Q = 2$ .
- A **função de erro médio não é linear** e, portanto, **não existe uma forma fechada** para encontrarmos os pesos. Porém, ela é **convexa** e, portanto, é garantido que o algoritmo do **gradiente descendente** encontre o mínimo global.

# Regressão Softmax

- Sendo assim, usamos o algoritmo do ***gradiente descendente*** para encontrar os ***pesos*** das  $Q$  funções discriminantes que ***minimizam*** a ***função de erro médio***.

- A atualização iterativa dos ***pesos*** da  $q$ -ésima classe,  $C_q$ , é dada por

$$\mathbf{a}_q = \mathbf{a}_q - \alpha \frac{\partial J_e(A)}{\partial \mathbf{a}_q}, \forall q$$

- Considerando uma ***função discriminante linear*** (i.e., ***hiperplano***), a derivada da ***função de erro médio***,  $J_e(A)$ , com respeito a cada vetor de pesos,  $\mathbf{a}_q$ , tem uma expressão idêntica àquela obtida para a ***regressão logística***:

Para outros formatos de função discriminante,  
basta alterarmos o formato da matriz  $X$ .

Forma  
matricial

$$\frac{\partial J_e(A)}{\partial \mathbf{a}_q} = -\frac{1}{N} \sum_{i=0}^{N-1} [1\{y(i) + 1 == q\} - h_a^q(\mathbf{x}(i))] \mathbf{x}(i)^T = -\frac{1}{N} \mathbf{X}^T (\mathbf{y}_q - \hat{\mathbf{y}}_q).$$

# Regressão Softmax

## Observações

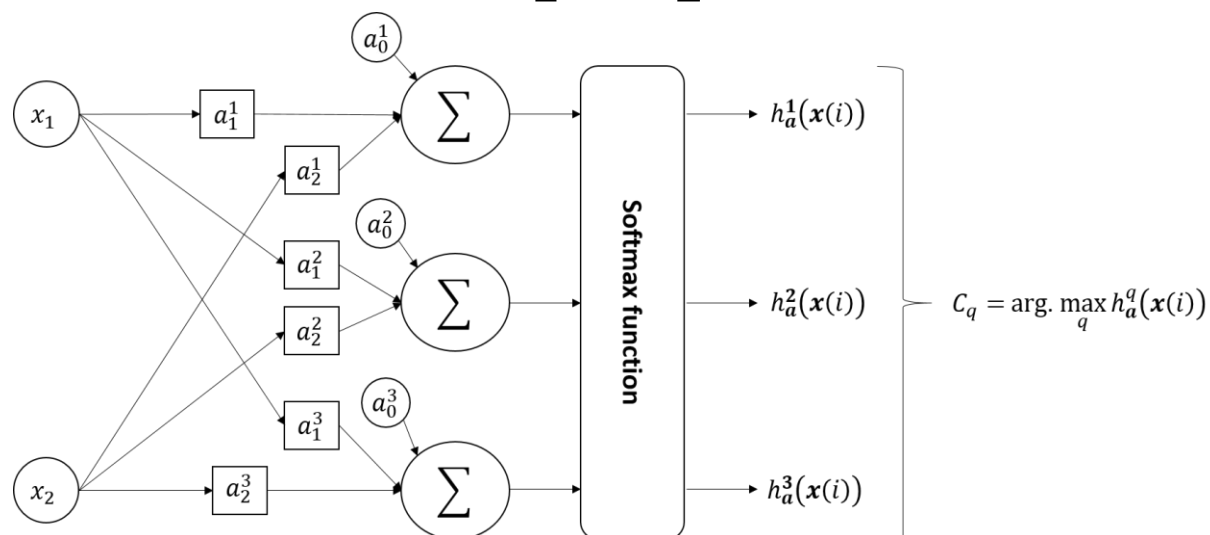
- O **vetor gradiente** da **função de erro** depende da **função discriminante** adotada.
- Entretanto, como vimos antes, esta dependência afeta apenas a **matriz de atributos**,  $X$ .
- O regressor softmax apresenta duas propriedades:
  - $0 \leq h_a^q(\mathbf{x}(i)) \leq 1$ , ou seja, a saída da  $q$ -ésima função hipótese de classificação sempre será um valor dentro do intervalo  $[0, 1]$ ;
  - $\sum_{q=1}^Q h_a^q(\mathbf{x}(i)) = \sum_{q=1}^Q P(C_q | \mathbf{x}(i), \mathbf{a}_q) = 1$ , ou seja, o somatório das **probabilidades condicionais** de todas as  $Q$  classes é igual a 1.
- Estas duas propriedades fazem com que o vetor

$$\mathbf{h}_a(\mathbf{x}(i)) = [h_a^1(\mathbf{x}(i)), \dots, h_a^Q(\mathbf{x}(i))]^T \in \mathbb{R}^{Q \times 1},$$

contendo todas as saídas do regressor softmax atenda os requisitos de uma **função massa de probabilidade multinomial**.

# Regressão Softmax

- Após o treinamento, o classificador atribui ao exemplo de entrada,  $\mathbf{x}(i)$ , a classe,  $q$ , com a **maior probabilidade estimada**, que é simplesmente a classe com maior valor para  $g_q(\mathbf{x}(i)) = \mathbf{x}(i)^T \mathbf{a}_q$   
$$C_q = \arg. \max_q h_a^q(\mathbf{x}(i)) = \arg. \max_q P(C_q \mid \mathbf{x}(i), \mathbf{a}_q) = \arg. \max_q \mathbf{x}(i)^T \mathbf{a}_q .$$
- A arquitetura de um **regressor softmax** para três classes (i.e.,  $Q = 3$ ) e dois atributos ( $x_1$  e  $x_2$ ) é mostrada abaixo.



A ideia por trás da **regressão softmax** é bastante simples: dado um exemplo de entrada  $\mathbf{x}$ , o regressor softmax primeiro calcula uma “**pontuação**”,  $g_q(\mathbf{x}) = \mathbf{x}^T \mathbf{a}_q$ , para cada classe  $q$ , em seguida, estima a probabilidade de cada classe aplicando a função softmax às “**pontuações**”.

# Tarefas

- **Quiz:** “*T320 - Quiz - Classificação (Parte IV)*” que se encontra no MS Teams.
- **Exercício Prático:** Laboratório #4.
  - Pode ser acessado através do link acima (Google Colab) ou no GitHub.
  - Se atentem aos prazos de entrega.
  - [Instruções para resolução e entrega dos laboratórios](#).
  - **Laboratórios podem ser resolvidos em grupo, mas as entregas devem ser individuais.**

Obrigado!