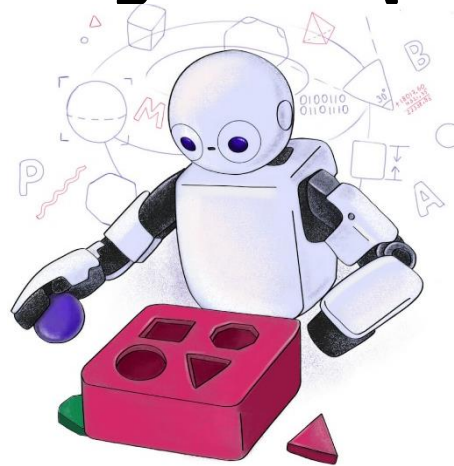


# T320 - Introdução ao Aprendizado de Máquina II: *Classificação (Parte IV)*



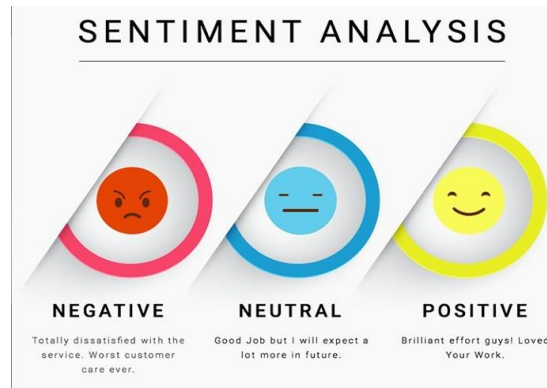
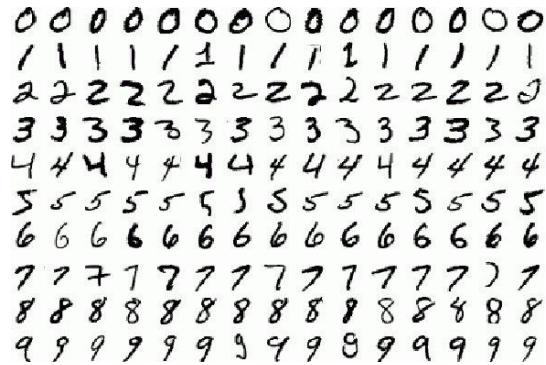
***Inatel***

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 encontrar uma solução para o problema de classificação usando 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 apenas 2 classes, 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***.

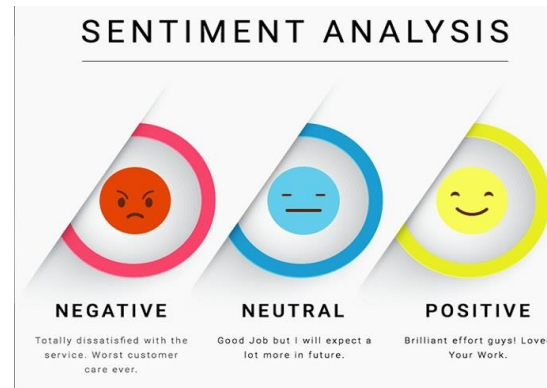
# Casos multi-classe



- Até agora, nós vimos como classificar utilizando ***regressão logística***.
- Nesse caso, os dados pertencem a apenas 2 classes (i.e.,  $Q = 2$ ).
- Porém, e quando o problema possuir 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.

# Casos multi-classe

0000000000000000  
1111111111111111  
2222222222222222  
3333333333333333  
4444444444444444  
5555555555555555  
6666666666666666  
7777777777777777  
8888888888888888  
9999999999999999



- Quando  $Q > 2$ , chamamos o problema de classificação **multi-classes**.
- 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, treina-se **um classificador binário** para cada classe  $q$ , para prever a probabilidade  $P(C_q | \mathbf{x}(i); \mathbf{a})$ ,  $\forall q \in \{1, \dots, Q\}$ , onde  $C_q$  é a classe positiva e as demais formam a classe negativa.
- Em outras palavras, treina-se  $Q$  **classificadores binários**, onde para cada classificador, a classe positiva é a  $q$ -ésima classe e a classe negativa é a junção de todas as outras,  $Q - 1$ , classes.
- Fazendo isso, nós **transformamos um problema com  $Q$  classes em  $Q$  problemas binários**.
- Cada um dos  $Q$  classificadores binários é representado pela função hipótese  $h_a^q(\mathbf{x}(i))$ .

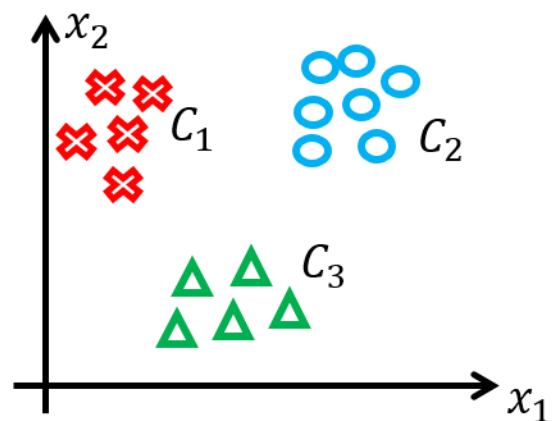
# Um-Contra-o-Resto

- Portanto, o  $q$ -ésimo *classificador* deve indicar a classe positiva caso o exemplo pertença à  $q$ -ésima classe, ou à classe negativa caso o exemplo pertença a qualquer uma das outras  $Q - 1$  classes.
- Após o treinamento, para cada exemplo de entrada,  $\mathbf{x}(i)$ , realiza-se  $Q$  previsões e escolhe-se a classe que maximize  $h_a^q(\mathbf{x}(i))$

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

- A ***vantagem*** desta abordagem é que treina-se apenas  $Q$  ***classificadores***.
- Uma ***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*** e a ***possibilidade de classes desbalanceadas***.

# Um-Contra-o-Resto



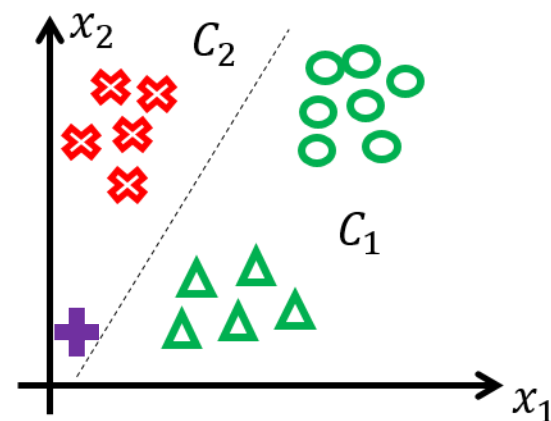
$$Q = 3$$

Passamos a ter **3**  
*classificadores binários.*

$C_1(+)$

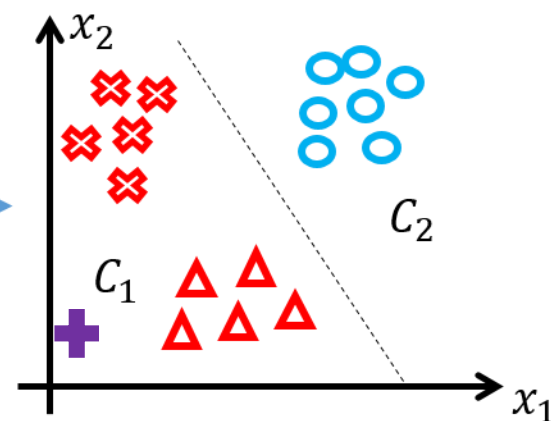
$C_2(+)$

$C_3(+)$

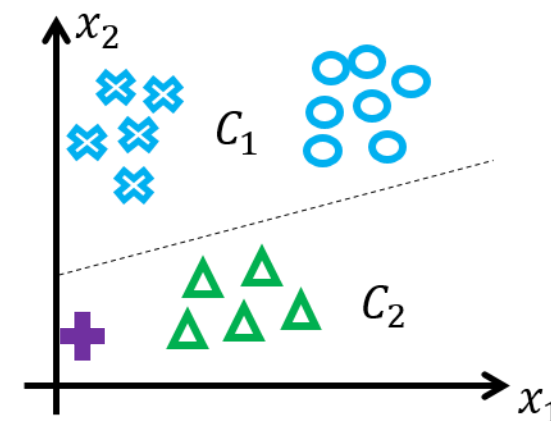


$$h_a^1(\mathbf{x}(i)) = 0.55$$

A qual classe pertence?



$$h_a^2(\mathbf{x}(i)) = 0.05$$



$$h_a^3(\mathbf{x}(i)) = 0.85$$

----- Fronteira de decisão

Exemplo de validação

# Um-Contra-Um

- Nesta abordagem, treina-se  $Q(Q - 1)/2$  **classificadores binários**.
- Cada **classificador** é treinado para classificar os exemplos pertencentes a **cada um dos possíveis pares de classes**.
  - Por exemplo, 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  $C_3/C_4$ .
- **Transformamos um problema com  $Q$  classes em  $Q(Q - 1)/2$  problemas binários.**
- No final, cada exemplo é classificado conforme o **voto majoritário** entre os **classificadores**.
- Ou seja, a classe que receber mais votos é a classe atribuída ao exemplo.



# Um-Contra-Um

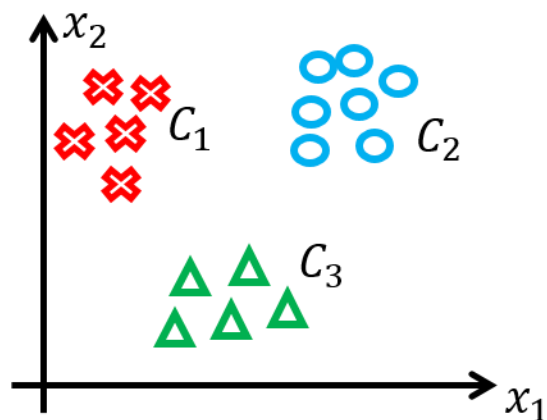
- A principal **vantagem** dessa abordagem é que cada classificador precisa ser **treinado apenas com as duas classes** que ele deve distinguir, portanto, a **chance de desbalanceamento é menor**.
- Além disso, o **tempo de treinamento de cada classificador também é menor**, pois treina-se cada um deles com pares de classes.
- A **desvantagem** é que, por exemplo, se  $Q = 10$ , temos que treinar 45 classificadores.
- Consequentemente, o **tempo total de treinamento pode ser alto**.

# Um-Contra-Um

$$Q = 3$$

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

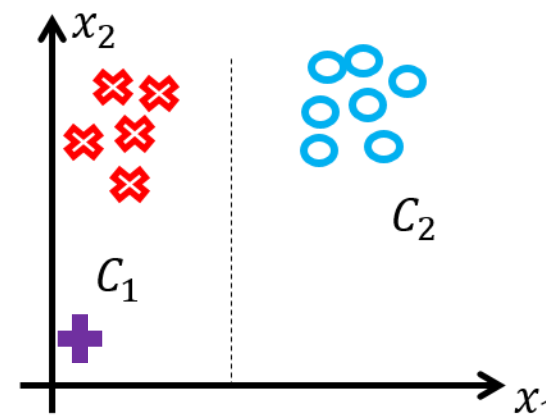
Passamos a ter **3**  
*classificadores binários.*



$C_2(+)/C_1(-)$

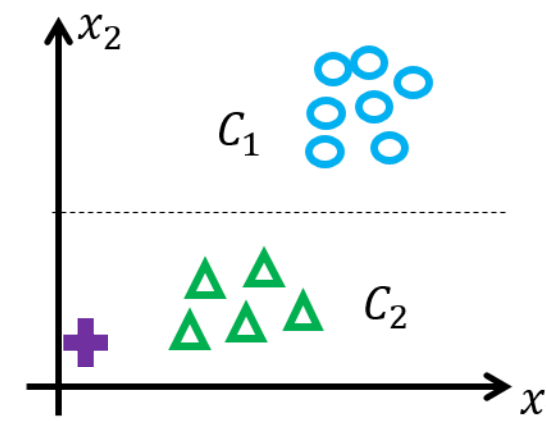
$C_3(+)/C_2(-)$

$C_1(+)/C_3(-)$

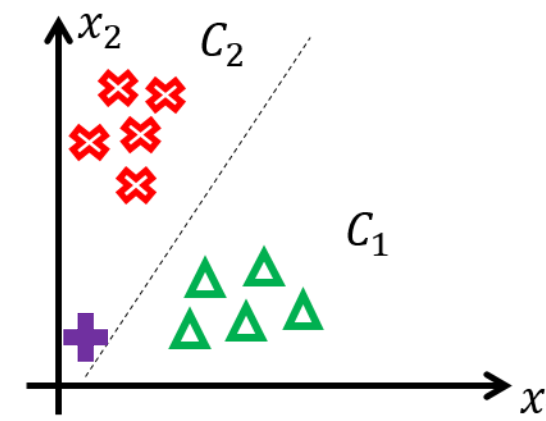


$$h_a^1(\mathbf{x}(i)) = 0.05 \text{ (voto em } C_1\text{)}$$

A qual classe pertence? **+**



$$h_a^2(\mathbf{x}(i)) = 0.85 \text{ (voto em } C_3\text{)}$$



$$h_a^3(\mathbf{x}(i)) = 0.55 \text{ (voto em } C_1\text{)}$$

----- Fronteira de decisão

**+** Exemplo de validação

# Regressão softmax

- Também conhecida como ***regressão logística multinomial***.
  - Pois as saídas deste regressor podem ser interpretadas como as ***probabilidades de uma variável categoricamente distribuída*** (as classes) ***dado um conjunto de variáveis*** (atributos e pesos).
  - Em outras palavras, a regressão ***softmax estima a probabilidade de um exemplo de entrada pertencer a cada uma das  $Q$  possíveis classes***.
- É uma generalização do regressor logístico para problemas com ***múltiplas classes***, em geral,  $Q > 2$ .
- A ideia é treinar um ***único*** classificador com  $Q$  saídas, onde cada saída representa a ***probabilidade*** de um exemplo pertencer a uma das  $Q$  classes.
  - Por exemplo, para um problema com 4 classes, teremos um único classificador, mas com 4 saídas.

# Regressão softmax

- 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 ao mesmo tempo.
- Para termos um ***único*** classificador, o ***regressor softmax possui uma função hipótese de classificação***,  $h_a^q(\mathbf{x}(i))$ , e, conseqüentemente, ***uma função discriminante***,  $g_q(\mathbf{x}(i))$ , ***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**,

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

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

- O somatório de termos exponenciais no denominador **normaliza** o valor da  $q$ -ésima saída de tal forma que o somatório das  $Q$  saídas seja igual a 1.
- Cada função discriminante tem seu próprio vetor de pesos,  $\mathbf{a}_q$ .

# Regressão softmax

- Assim como com o regressor logístico, podemos usar equações de **hiperplanos** ou **polinomiais** como **funções discriminantes**.
- 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, onde **a soma destas  $Q$  probabilidades deve ser igual a 1**

$$P(C_0 | \mathbf{x}(i), \mathbf{a}_0) + P(C_1 | \mathbf{x}(i), \mathbf{a}_1) + \cdots + P(C_{Q-1} | \mathbf{x}(i), \mathbf{a}_{Q-1}) = 1.$$

- 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**.

# Regressão softmax

- 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

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

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** das  $Q$  **funções hipótese** 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**.
- Essa função também é conhecida como **função da entropia cruzada**.

# Regressão softmax

- Usando-se a codificação **one-hot**, a equação de erro médio pode ser reescrita 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  $\mathbf{y}(i) = [1\{y(i) == 0\}, \dots, 1\{y(i) == Q - 1\}]^T \in \mathbb{R}^{Q \times 1}$  é um vetor coluna utilizando a codificação **one-hot** e  $\mathbf{h}_a(\mathbf{x}(i)) \in \mathbb{R}^{Q \times 1}$  é um vetor coluna com as saídas das  $Q$  **funções hipóteses de classificação**

$$\begin{aligned} \mathbf{h}_a(\mathbf{x}(i)) &= \left[ h_a^0(\mathbf{x}(i)) \quad \dots \quad h_a^{Q-1}(\mathbf{x}(i)) \right]^T \\ &= \left[ P(C_0 \mid \mathbf{x}(i), \mathbf{a}_0) \quad \dots \quad P(C_{Q-1} \mid \mathbf{x}(i), \mathbf{a}_{Q-1}) \right]^T. \end{aligned}$$

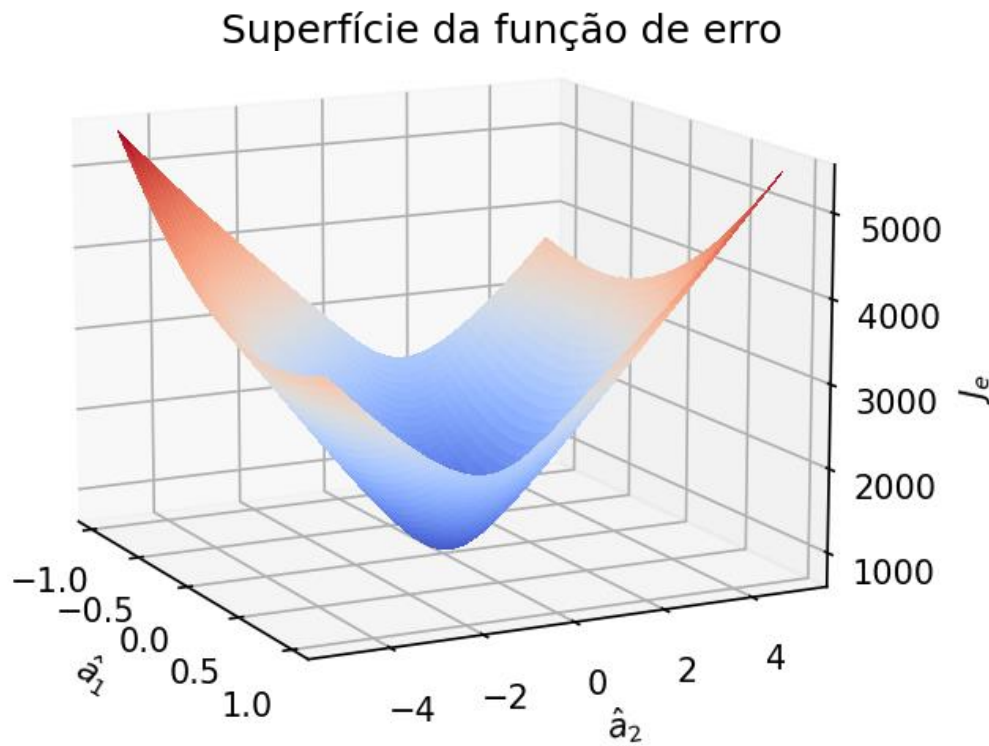


# Regressão softmax

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

- Notem que quando existem apenas duas classes ( $Q = 2$ ), a ***função de erro médio*** é equivalente à ***função de erro médio*** do ***regressor logístico***.
- Ou seja, mesmo tendo sido pensado para o caso onde  $Q > 2$ , o regressor softmax pode ser usado quando  $Q = 2$ .
- Entretanto, existe um problema com essa função de erro médio.
- A ***função de erro médio não é linear*** e, portanto, ***não existe uma forma fechada*** para encontrarmos os pesos.

# Regressão softmax



- Porém, ela é **convexa** e **contínua**, portanto, é **garantido que o algoritmo do gradiente descendente encontre o mínimo global**.
- Sendo assim, usaremos o algoritmo do **gradiente descendente** para encontrar os **pesos** das  $Q$  funções discriminantes que **minimizam a função de erro médio**.


# Regressão softmax

- A atualização dos **pesos** da  $q$ -ésima função discriminante,  $g_q(\mathbf{x}(i))$ , é dada por

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

- Considerando o **hiperplano** como **função discriminante**, a derivada da **função de erro médio**,  $J_e(\mathbf{A})$ , com respeito ao vetor de pesos,  $\mathbf{a}_q$ , é dada por

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

Forma matricial 

onde  $\mathbf{y}_q$  e  $\hat{\mathbf{y}}_q$  são vetores coluna ambos com dimensão  $N \times 1$  contendo os  $q$ -ésimos elementos dos vetores *one-hot* e de saída do regressor softmax para cada um dos  $N$  vetores de atributo, respectivamente.

# Regressão softmax

$$\frac{\partial J_e(\mathbf{A})}{\partial \mathbf{a}_q} = -\frac{1}{N} \mathbf{X}^T (\mathbf{y}_q - \hat{\mathbf{y}}_q).$$

- Notem que essa expressão do vetor gradiente é **idêntica** àquela obtida para o caso da **regressão logística**.
- O vetor gradiente pode ser reescrito levando em consideração todos os  $Q$  vetores de peso:

$$\frac{\partial J_e(\mathbf{A})}{\partial \mathbf{A}} = -\frac{1}{N} \mathbf{X}^T (\mathbf{Y} - \mathbf{H}_A),$$

onde  $\mathbf{Y}$  e  $\mathbf{H}_A$  são matrizes com dimensão  $N \times Q$  contendo em cada linha os vetores  $\mathbf{y}(i)^T$  e  $\mathbf{h}_a(\mathbf{x}(i))^T$  para todos os  $N$  exemplos, respectivamente.

# Regressão softmax

$$\frac{\partial J_e(A)}{\partial A} = -\frac{1}{N} X^T (Y - H_A)$$

- Como aconteceu com a regressão logística, notem que o **vetor gradiente** da **função de erro** *depende do formato da função discriminante* adotada.
- Entretanto, como vimos antes, esta *dependência afeta apenas a matriz de atributos, X*.
- Portanto, *para formatos* de **função discriminante** diferentes de um hiperplano, basta *criar a matriz de atributos, X, seguindo o formato da função*.

# Regressão softmax

- 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 sempre será um valor dentro do intervalo  $[0, 1]$ ;
  - $\sum_{q=0}^{Q-1} h_a^q(\mathbf{x}(i)) = \sum_{q=0}^{Q-1} P(C_q | \mathbf{x}(i); \mathbf{a}_q) = 1$ , ou seja, o somatório das **probabilidades condicionais** de todas as  $Q$  classes é igual a 1.

- Estas duas propriedades fazem com que o vetor

$$\mathbf{h}_a(\mathbf{x}(i)) = [h_a^0(\mathbf{x}(i)) \quad \cdots \quad h_a^{Q-1}(\mathbf{x}(i))]^T \in \mathbb{R}^{Q \times 1},$$

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

# Regressão softmax

- Após o treinamento, durante a **fase de predição**, o classificador atribui ao exemplo de entrada,  $\mathbf{x}(i)$ , a classe,  $q$ , com a **maior probabilidade estimada**, ou seja, atribui a classe,  $q$ , da função hipótese com maior valor:

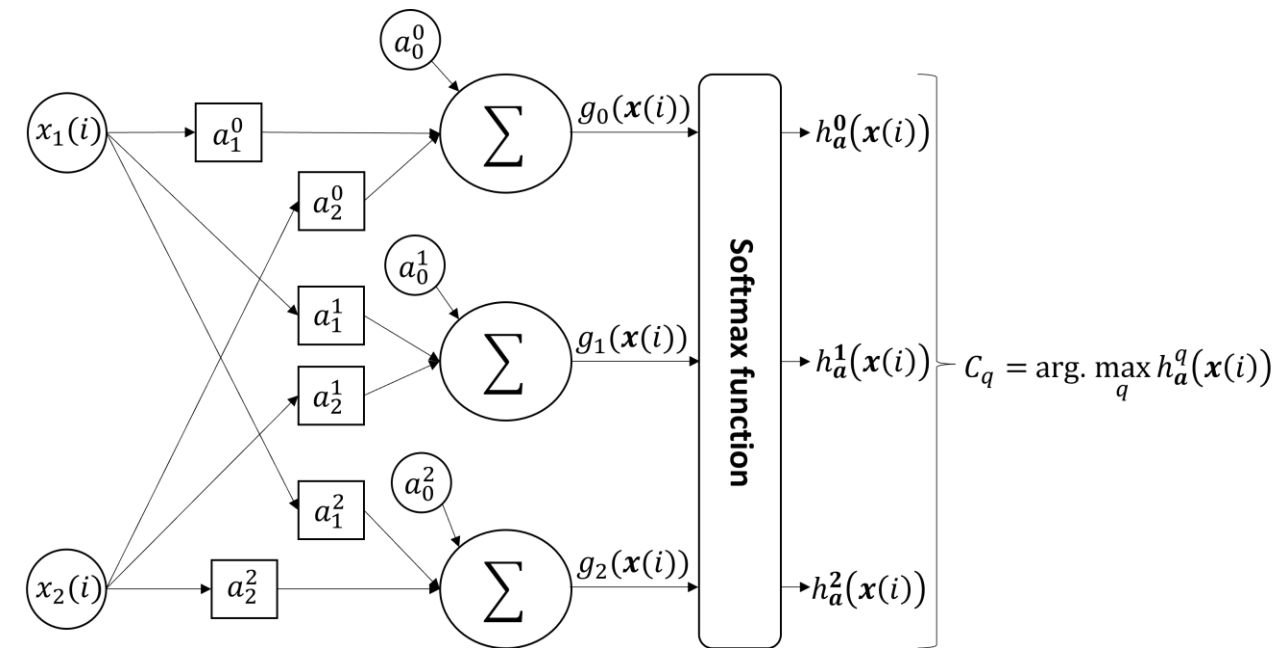
$$C_q = \arg. \max_q h_a^q(\mathbf{x}(i)) = \arg. \max_q P(C_q \mid \mathbf{x}(i); \mathbf{a}_q).$$

- Uma outra forma de reescrevermos o problema de maximização acima é lembrarmos que a **classe com maior probabilidade** é simplesmente a **classe com maior valor de função discriminante**,  $g_q(\mathbf{x}(i)) = \mathbf{x}(i)^T \mathbf{a}_q$

$$C_q = \arg. \max_q g_q(\mathbf{x}(i)) = \arg. \max_q \mathbf{x}(i)^T \mathbf{a}_q.$$

- **OBS.:** A **normalização** é fundamental para garantir que as saídas do modelo sejam **interpretáveis como probabilidades** e para **garantir um treinamento estável**.

# Regressão softmax



- A arquitetura de um **regressor softmax** para um problema com três classes (i.e.,  $Q = 3$ ) e dois atributos ( $x_1$  e  $x_2$ ) é mostrada ao lado.
- A ideia por trás da **regressão softmax** é bastante simples:
  - dado um exemplo de entrada,  $x(i)$ ,
  - o regressor softmax calcula uma “**pontuação**”, para cada classe  $q$ , i.e., as saídas das  $Q$  funções discriminantes  $g_q(x(i)) = x(i)^T a_q, \forall q$ .
  - e, 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](#).

Obrigado!