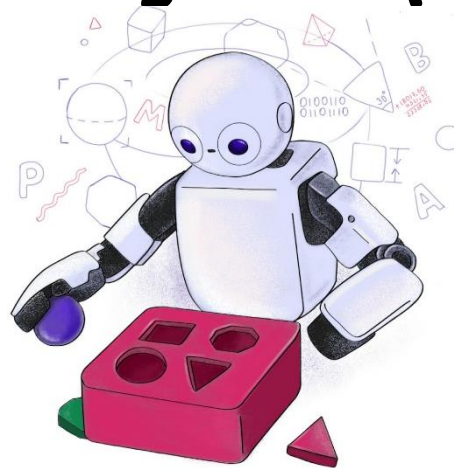


# T320 - Introdução ao Aprendizado de Máquina II: *Classificação (Parte II)*



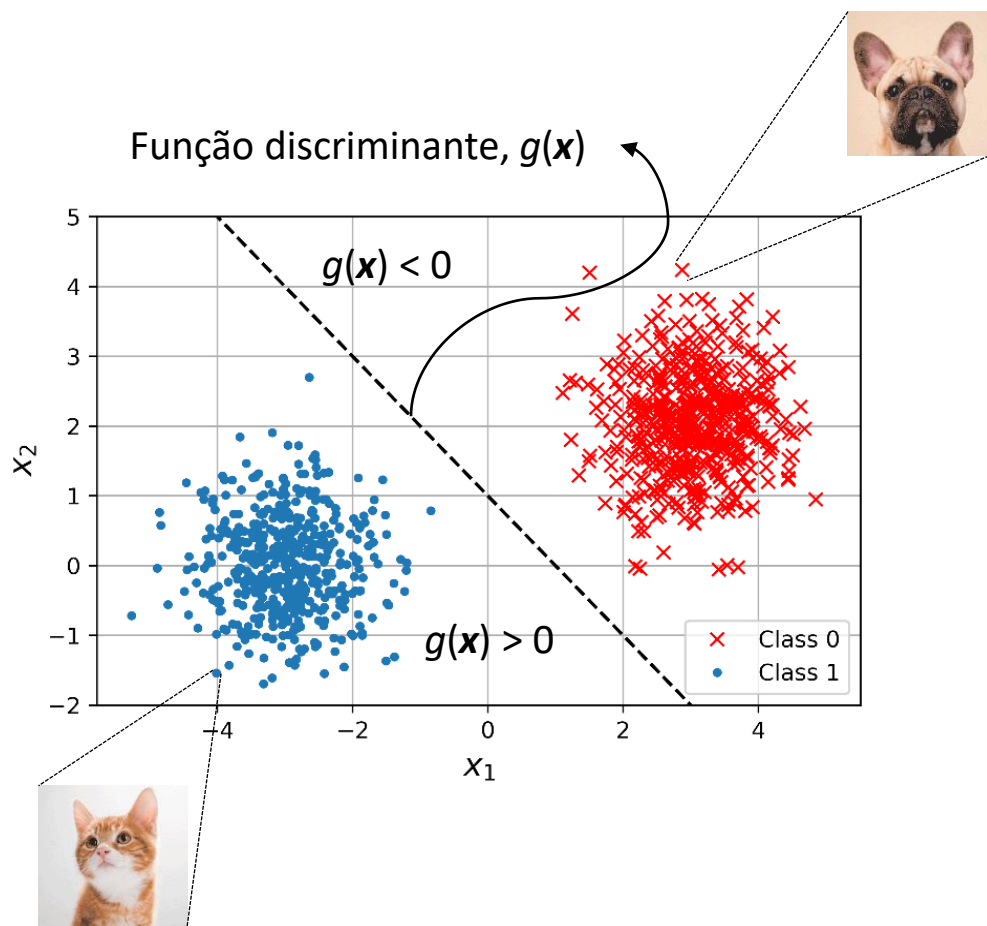
***Inatel***

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

# Recapitulando

- Anteriormente, vimos alguns exemplos de aplicação de algoritmos de ***classificação***:
  - Detecção de spam.
  - Análise de sentimentos.
  - Reconhecimento de objetos, faces, letras/dígitos.
- Definimos o problema da classificação e concluimos que ele também é um problema de ***aprendizado supervisionado***.
- Aprendemos que as classes são separadas através de ***funções discriminantes*** e que o desafio é encontrar funções adequadas e seus respectivos pesos.
- A partir da aula de hoje, começamos a discutir como encontrar os pesos.

# Classificação linear

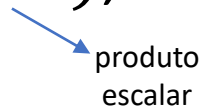


- Como vimos, o objetivo da **classificação** é usar as características (i.e., vetores de atributos,  $x$ ) de, por exemplo, um e-mail ou imagem, para identificar a qual classe ele pertence.
- Um **classificador linear** atinge esse objetivo **tomando uma decisão** (i.e., os **ifs** e **elses**) com base no valor de uma **combinação linear dos atributos em relação aos pesos**, ou seja, na saída de uma **função discriminante linear**.

# Classificação linear

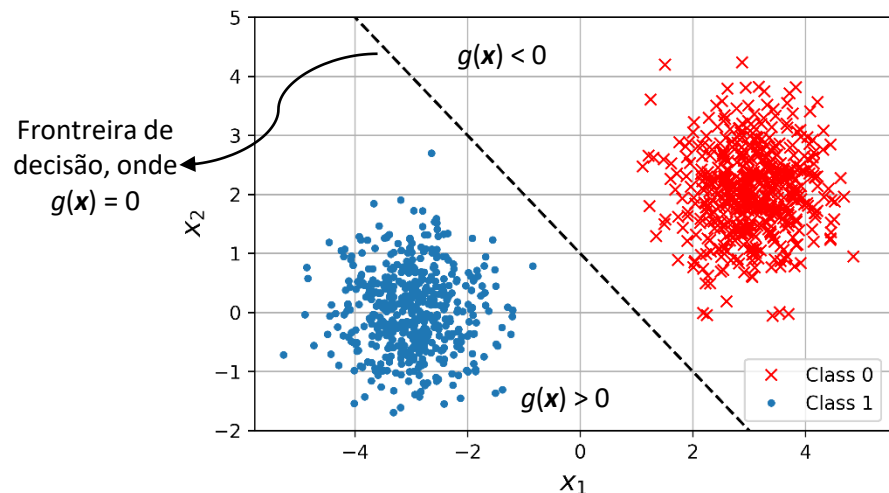
- Portanto, a saída de um **classificador linear** é dada por

$$\hat{y} = h_a(\mathbf{x}) = f(g(\mathbf{x})) = f(a_0 + \cdots + a_K x_K) = f\left(\sum_{k=0}^K a_k x_k\right) = f(\mathbf{a}^T \mathbf{x}),$$

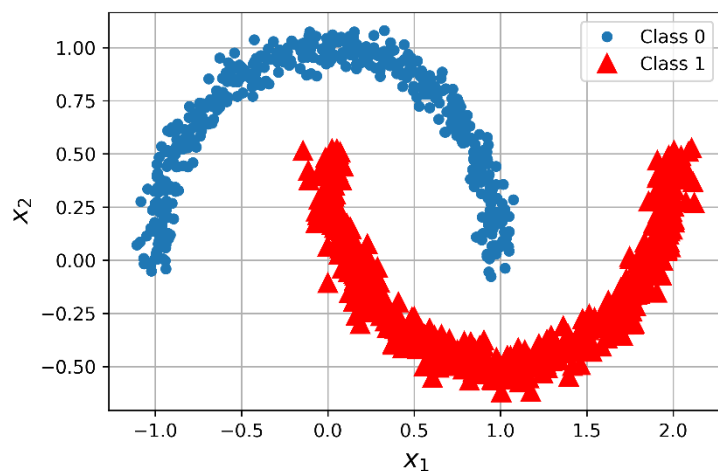
onde  $h_a(\mathbf{x})$  é conhecida como **função hipótese de classificação**,  $\mathbf{x} = [1, x_1, \dots, x_K]^T$  é o vetor de atributos com o primeiro elemento sendo o atributo de *bias*,  $x_0 = 1$ , e  $f(\cdot)$  é uma **função de limiar de decisão**. 

- **Função de limiar de decisão** é uma função que mapeia a saída da **função discriminante linear**,  $g(\mathbf{x})$ , na saída desejada, ou seja, na classe  $C_q$ ,  $q = 1, \dots, Q$ , do objeto.
- Ela é apenas uma formalização matemática para os **ifs** e **elses** que usamos para decidir as classes dos exemplos (i.e., atributos) de entrada.
- Na teoria original dos classificadores lineares, as **funções discriminantes** seguiam equações de hiperplanos:  $\sum_{k=0}^K a_k x_k$ .

# Classificação linear



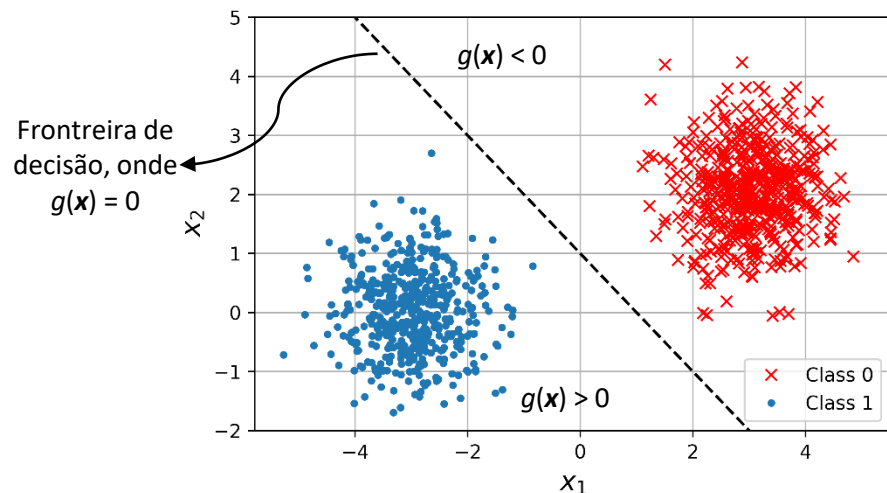
Classes linearmente separáveis.



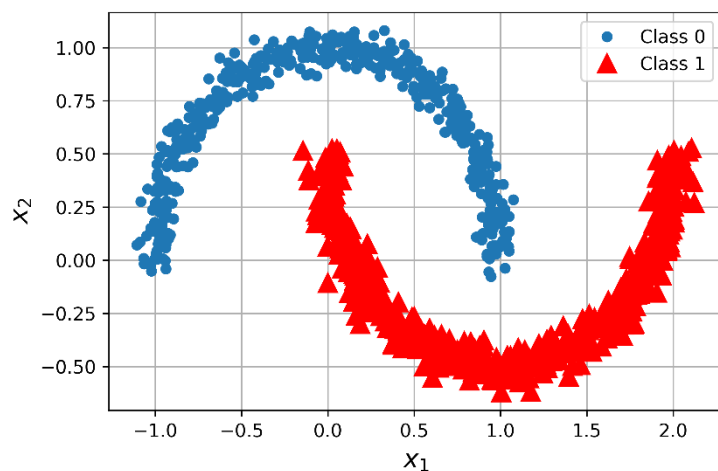
Classes não-linearmente separáveis.

- Dado um **conjunto de treinamento**, a tarefa do **classificador** é a de **aprender** uma **função hipótese de classificação**,  $h_a(x)$ , que receba um exemplo de entrada (e.g.,  $x_1$  e  $x_2$ ) e retorne a classe do exemplo.
- Para que um **classificador linear** funcione corretamente, as classes devem ser **linearmente separáveis**.
- Isso significa que as classes devem ser **suficientemente separadas** umas das outras para garantir que a **superfície de decisão** seja um **hiperplano**.

# Classificação linear



Classes linearmente separáveis.



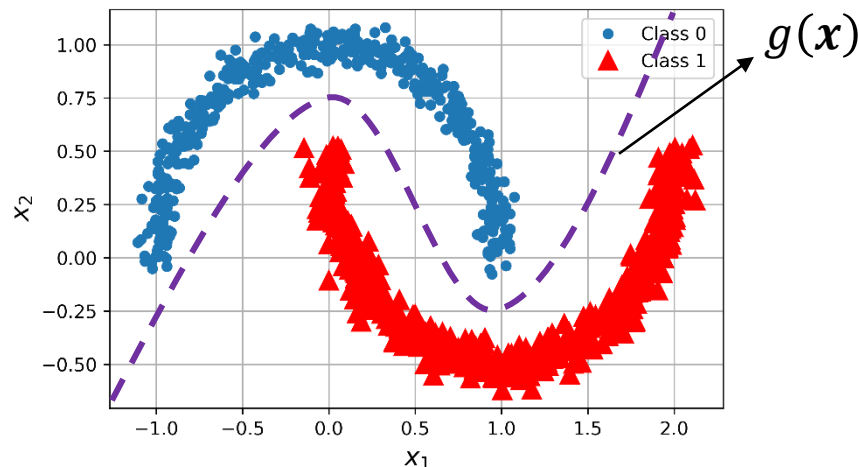
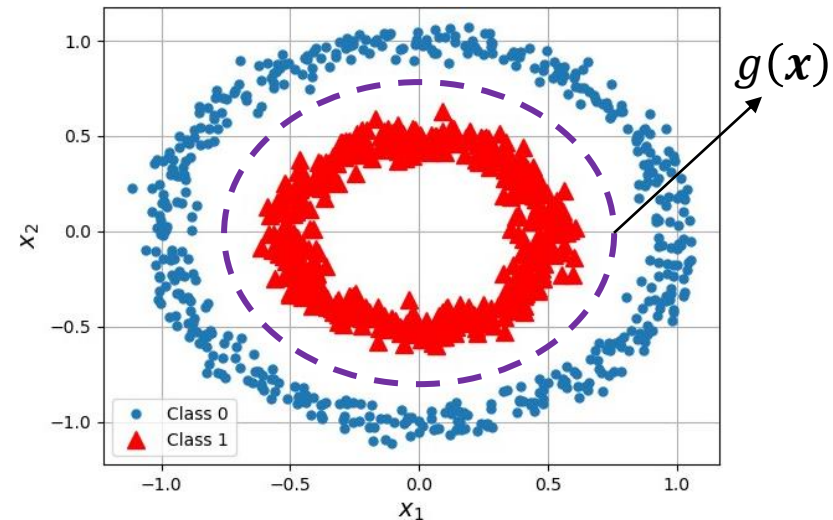
Classes não-linearmente separáveis.

- Classes que podem ser separadas por um **hiperplano** são chamadas de **linearmente separáveis**.
- Na primeira figura, a **fronteira de decisão** é definida por uma **função discriminante** que tem formato de uma **reta**:

$$g(\mathbf{x}) = 1 - x_1 - x_2.$$

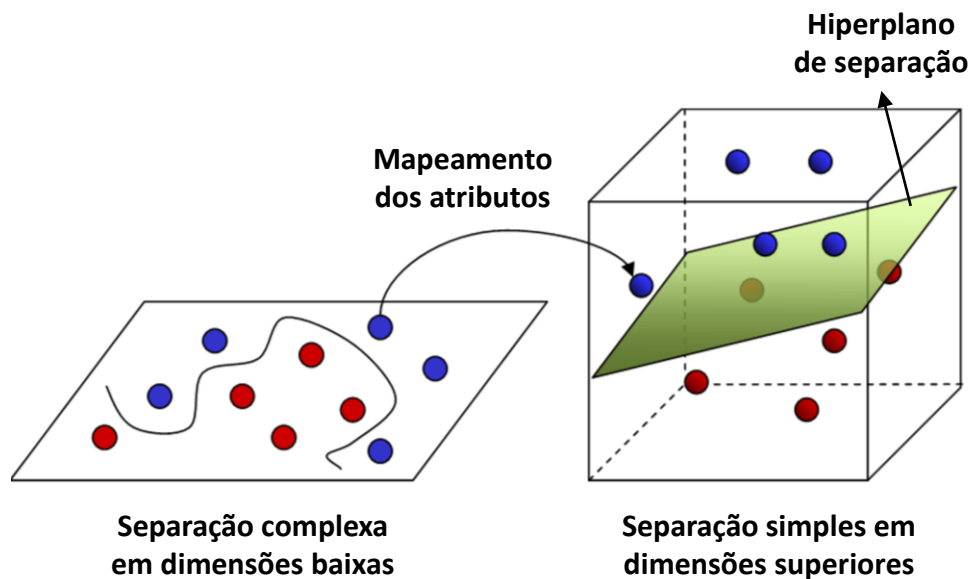
- Na segunda figura, devido à proximidade das classes, não existe um **hiperplano** que as separe.
- Originalmente, **classificação linear** é usada quando as classes podem ser separadas por **superfícies de decisão lineares**.
- Ou seja, as **funções discriminantes** são **hiperplanos**:  $\sum_{k=0}^K a_k x_k$ .

# Classificação não-linear



- Mas e se não pudermos separar as classes com um **hiperplano**, ou seja, se elas não forem **linearmente separáveis**?
- Nestes casos, usamos **funções discriminantes não-lineares**, como, por exemplo, **polinômios**:
  - $g(x) = (x_1 - a)^2 + (x_2 - b)^2 - r^2$ , Círculo centrado em  $(a, b)$  e com raio  $r$ .
  - $g(x) = \frac{(x_1 - a)^2}{c^2} + \frac{(x_2 - b)^2}{d^2} - 1$ , Elipse centrada em  $(a, b)$ , com largura  $2c$  e altura  $2d$ .
  - $g(x) = (x_1 - a)(x_2 - b) - c$ , Hipérbole retangular com eixos paralelos às suas assíntotas.

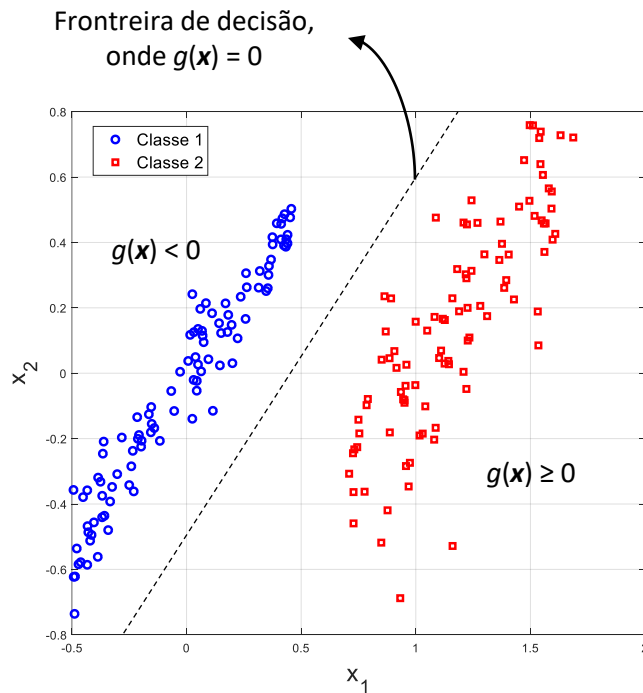
# Classificação não-linear



- As **função discriminantes não-lineares** aplicam **transformações não-lineares aos atributos originais**, levando ao **aumento das dimensões de entrada**.
  - $$g(x) = (x_1 - a)^2 + (x_2 - b)^2 - r^2$$
$$= x_1^2 - 2ax_1 + x_2^2 - 2bx_2 + (a^2 + b^2 - r^2)$$
$$= a_1z_1 + a_2z_2 + a_3z_3 + a_4z_4 + a_0$$
- Essas transformações realizam **mapeamentos não-lineares dos atributos** para um **espaço de dimensão superior** onde as classes possam ser mais facilmente separadas.

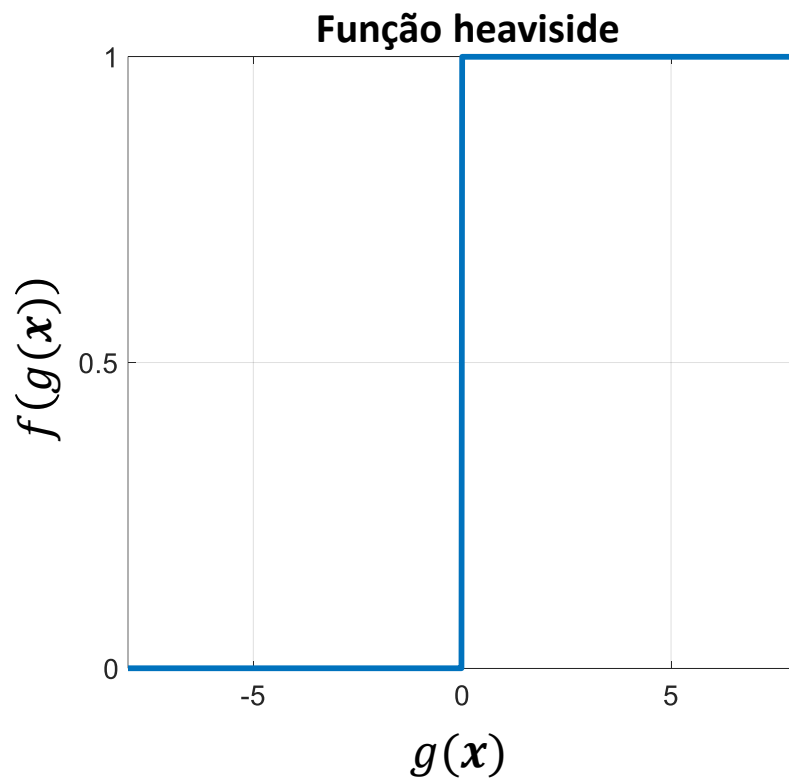


# Função de limiar de decisão



- Para o exemplo ao lado, podemos definir a **função hipótese de classificação** como duas **condições**:
$$\hat{y} = h_a(\mathbf{x}) = \begin{cases} 0, & g(\mathbf{x}) = \mathbf{x}^T \mathbf{a} < 0 \text{ (Classe 1)} \\ 1, & g(\mathbf{x}) = \mathbf{x}^T \mathbf{a} \geq 0 \text{ (Classe 2)} \end{cases} \quad \left. \begin{array}{l} \text{if} \\ \text{else} \end{array} \right\}$$
- Percebam que a saída da **função hipótese de classificação** é **binária**, ou seja, como temos **2 classes**, temos apenas **2 possíveis valores de saída**, 0 ou 1.
- O mapeamento entre o valor da função discriminante,  $g(\mathbf{x})$ , e a saída 0 ou 1 é feito através da **função de limiar de decisão**,  $f(g(\mathbf{x}))$ .
- Como implementar essas **condições** através de uma função matemática?

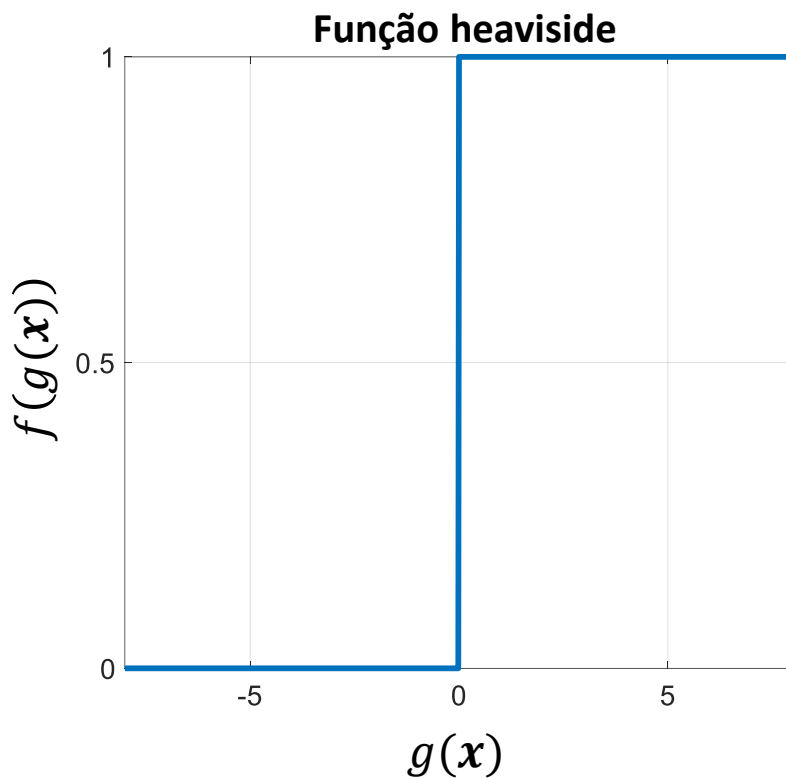
# Função de limiar de decisão rígido



Conhecida também  
como **função heavyside**  
ou **degrau unitário**.

- Uma **função de limiar de decisão** simples que faz o mapeamento do valor de  $g(x)$  em apenas 2 valores de saída é chamada de **função de limiar de decisão rígido**.
- A **função de limiar de decisão rígido** é mostrada na figura ao lado e é definida como
$$f(g(x)) = \begin{cases} 0, & g(x) < 0 \\ 1, & g(x) > 0 \\ \text{Indeterminado}, & g(x) = 0 \end{cases}$$

# Classificação com limiar de decisão rígido



- Agora que a **função hipótese de classificação**,  $h_a(x)$ , tem uma forma matemática bem definida, precisamos pensar em como encontrar os pesos,  $a$ .
- Nós queremos encontrá-los de tal forma que **o erro de classificação seja minimizado**, ou seja, que os exemplos sejam atribuídos corretamente às suas respectivas classes.
- No caso da **regressão linear**, nós fizemos isso de duas maneiras:
  - i. de forma fechada (através da **equação normal**) fazendo a derivada parcial do erro em relação aos pesos igual a zero e resolvendo a equação para os pesos;
  - ii. e através do algoritmo do **gradiente descendente**.
- Entretanto, com a **função de limiar rígido**, **nenhuma das duas abordagens é possível** devido à **derivada** de  $f(g(x))$  ser igual a zero em todos os pontos exceto em  $g(x) = 0$ , onde ela é indeterminada.
- **Portanto, o que podemos fazer?**

# Classificação com limiar de decisão rígido

- Uma possível abordagem para o problema da aprendizagem quando utilizamos o **limiar de decisão rígido** é utilizar uma **regra intuitiva** de atualização dos **pesos** que **converge para uma solução dado que exista uma função discriminante adequada e que as classes não se sobreponham**.
- Essa **regra intuitiva de atualização dos pesos** é dada pela seguinte equação

$$\mathbf{a} = \mathbf{a} + \alpha \left( y(i) - h_{\mathbf{a}}(\mathbf{x}(i)) \right) \mathbf{x}(i), \forall i,$$

onde  $\alpha$  é o passo de aprendizagem, o qual é sempre maior do que zero.

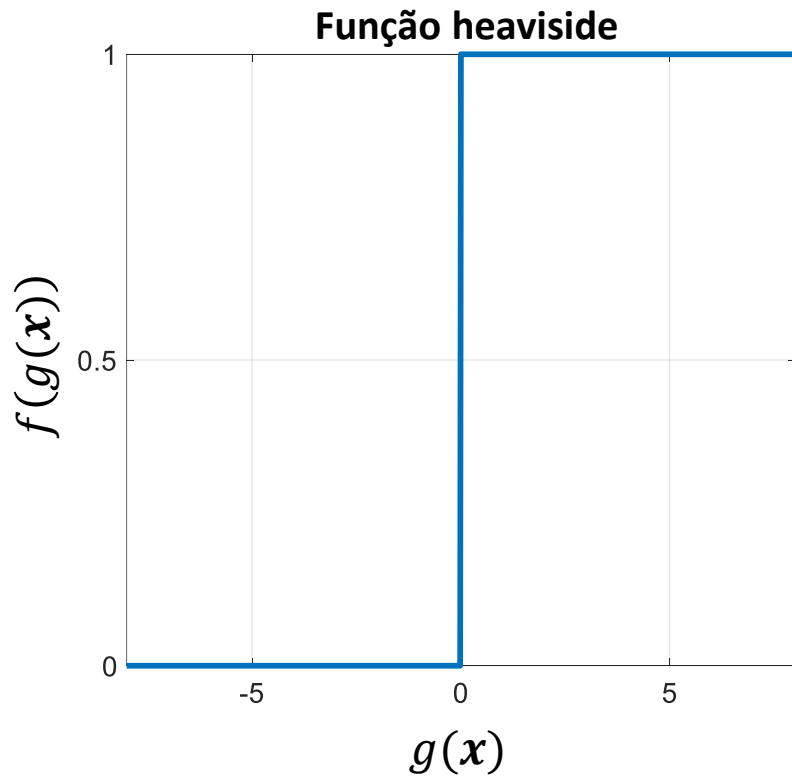
- A regra é essencialmente idêntica à regra de atualização para a **regressão linear** quando utilizamos o **gradiente descendente estocástico**.

# Classificação com limiar de decisão rígido

$$\mathbf{a} = \mathbf{a} + \alpha \left( y(i) - h_{\mathbf{a}}(\mathbf{x}(i)) \right) \mathbf{x}(i), \forall i.$$

- Por razões que discutiremos mais adiante, esta regra é chamada de ***regra de aprendizagem do perceptron***.
- Essa regra de aprendizagem é ***aplicada a um exemplo por vez***, escolhido de forma ***aleatória***, assim como fizemos com o ***gradiente descendente estocástico***.
  - Ou seja, ***atualiza-se os pesos usando-se apenas um exemplo, tomado de forma aleatória*** do conjunto de treinamento, por vez.
- Como estamos considerando classificadores, os quais têm valores de saída iguais a 0 ou 1, o comportamento da regra de atualização será diferente do comportamento para a regressão linear, como veremos a seguir.

# Classificação com limiar de decisão rígido



- Observem a equação de atualização dos pesos

$$\mathbf{a} = \mathbf{a} + \alpha \left( y(i) - h_{\mathbf{a}}(\mathbf{x}(i)) \right) \mathbf{x}(i),$$

onde  $\mathbf{a} = [a_0 \quad a_1 \quad \cdots \quad a_K]^T$  e  $\mathbf{x}(i) = [x_0(i) \quad x_1(i) \quad \cdots \quad x_K(i)]^T$ .

- Ambos, o valor esperado,  $y$ , e a saída da **função hipótese de classificação**,  $h_{\mathbf{a}}(\mathbf{x})$ , assumem os valores 0 ou 1. Portanto, existem **apenas 3 possibilidades**.

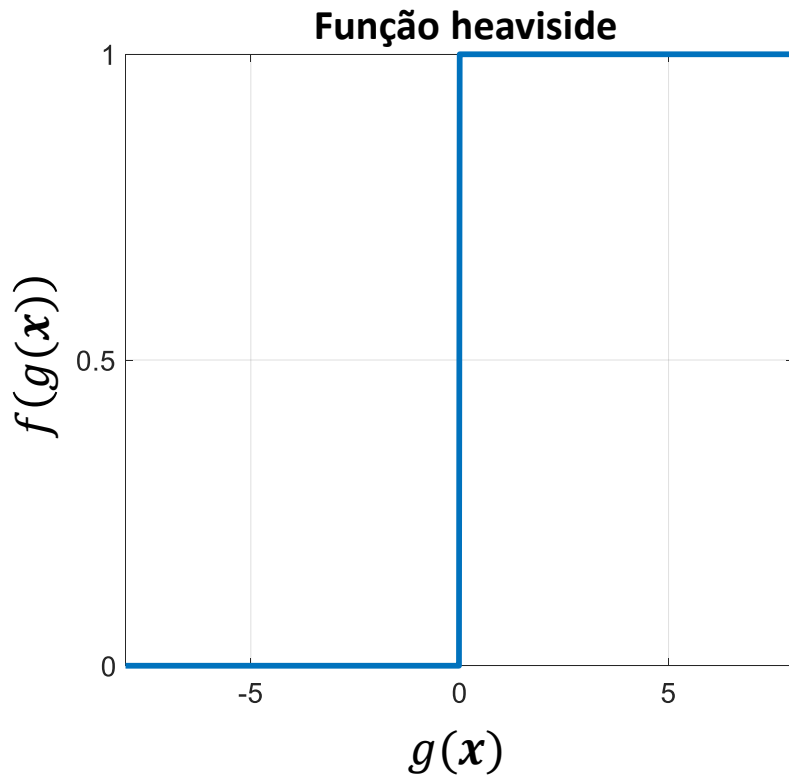
- Primeira possibilidade

- Se o valor de saída do classificador for igual ao esperado, i.e.,  $h_{\mathbf{a}}(\mathbf{x}(i)) = y(i)$ , então

$$y(i) - h_{\mathbf{a}}(\mathbf{x}(i)) = 0.$$

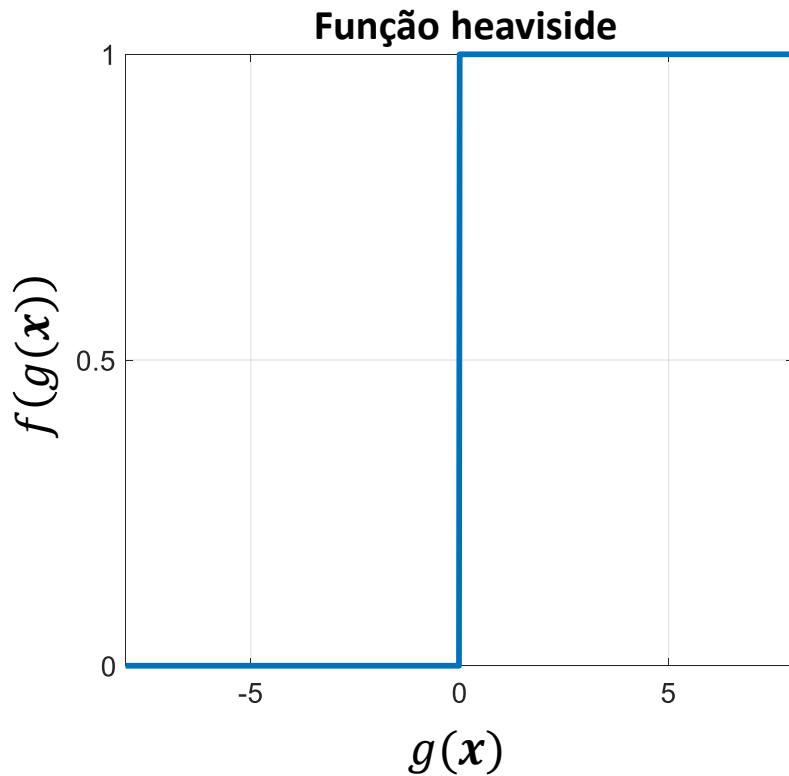
- Portanto, **os pesos não são atualizados**.

# Classificação com limiar de decisão rígido



- Equação de atualização dos pesos
$$\mathbf{a} = \mathbf{a} + \alpha \left( y(i) - h_{\mathbf{a}}(\mathbf{x}(i)) \right) \mathbf{x}(i).$$
- Segunda possibilidade
  - Se  $y(i) = 1$ , mas  $h_{\mathbf{a}}(\mathbf{x}(i)) = 0$ , então
$$\mathbf{a} = \mathbf{a} + \alpha \mathbf{x}(i).$$
  - Assim, o  $k$ -ésimo peso,  $a_k$ , tem seu valor **aumentado** quando o valor de  $a_k \times x_k$  é positivo e **diminuído** quando o valor de  $a_k \times x_k$  é negativo.
    - Isso faz sentido pois nós queremos **aumentar** o valor de  $g(\mathbf{x})$ , de tal forma que  $g(\mathbf{x}) > 0$  e, conseqüentemente,  $h_{\mathbf{a}}(\mathbf{x})$  tenha como saída o valor 1.
    - Lembrando que  $g(\mathbf{x}) = a_0 + a_1x_1 + \dots + a_Kx_K$ .

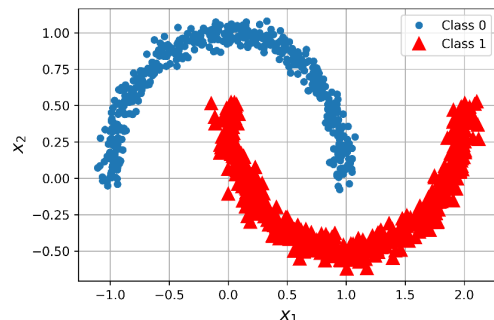
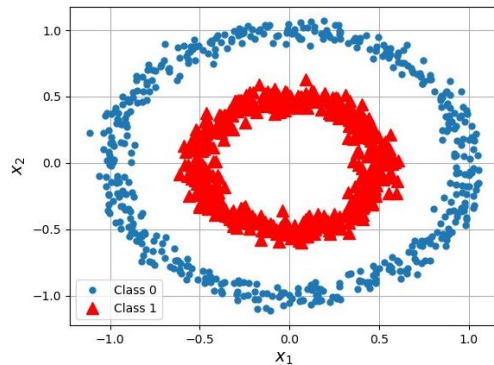
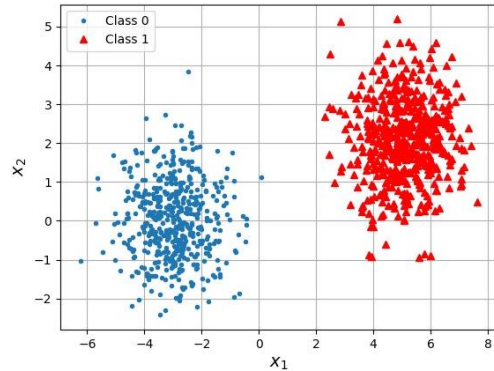
# Classificação com limiar de decisão rígido



- Equação de atualização dos pesos
$$\mathbf{a} = \mathbf{a} + \alpha \left( y(i) - h_{\mathbf{a}}(\mathbf{x}(i)) \right) \mathbf{x}(i).$$
- Terceira possibilidade
  - Se  $y(i) = 0$ , mas  $h_{\mathbf{a}}(\mathbf{x}(i)) = 1$ , então
$$\mathbf{a} = \mathbf{a} - \alpha \mathbf{x}(i).$$
  - Assim, o  $k$ -ésimo peso,  $a_k$ , tem seu valor **diminuído** quando o valor de  $a_k \times x_k$  é positivo e **aumentado** quando o valor de  $a_k \times x_k$  é negativo.
    - Isso faz sentido pois nós queremos **diminuir** o valor de  $g(\mathbf{x})$ , de tal forma que  $g(\mathbf{x}) < 0$  e, conseqüentemente,  $h_{\mathbf{a}}(\mathbf{x})$  tenha como saída o valor 0.
    - Lembrando que  $g(\mathbf{x}) = a_0 + a_1x_1 + \dots + a_Kx_K$ .

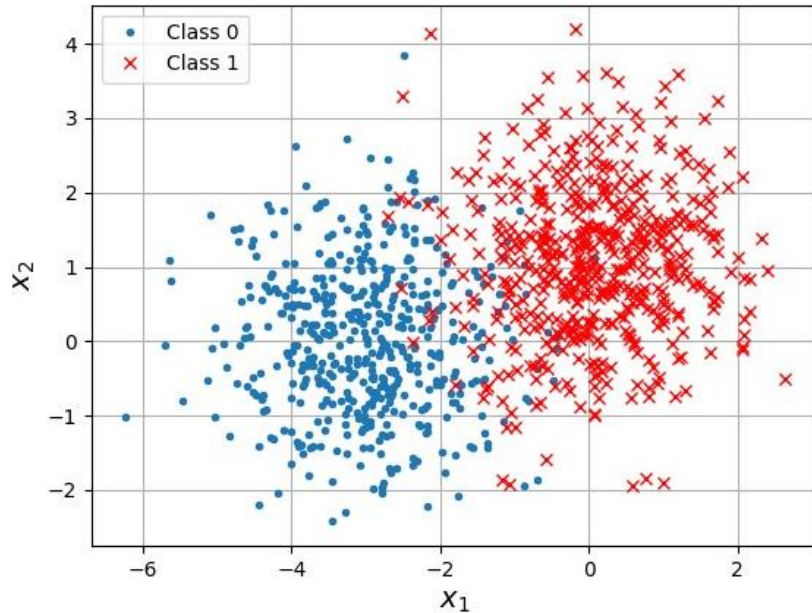


# Classificação com limiar de decisão rígido



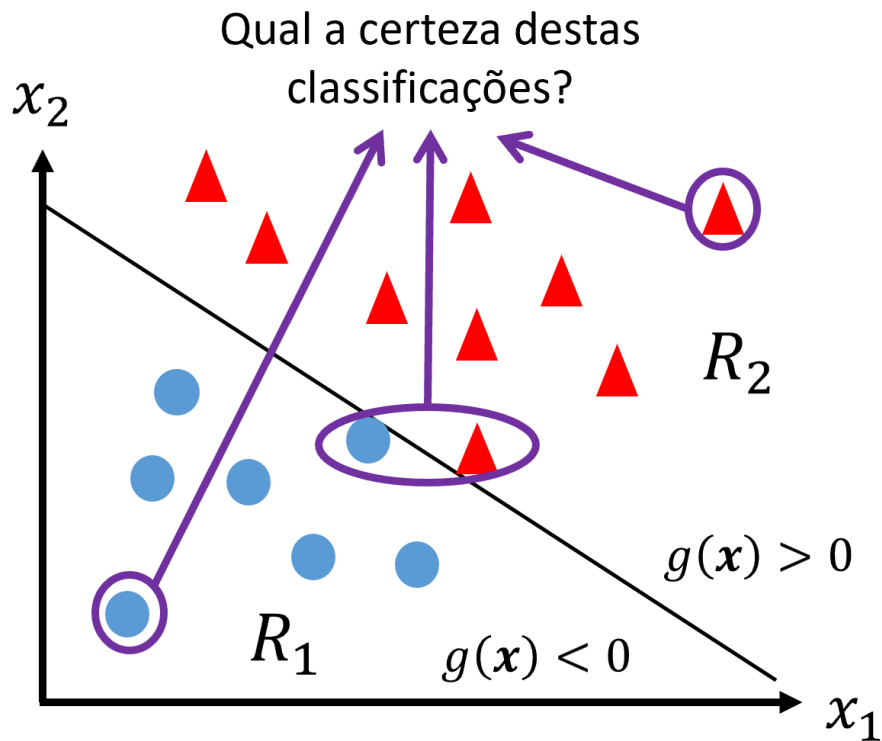
- A **regra de aprendizagem do perceptron** converge para um **separador perfeito** quando:
  - As classes são **suficientemente separadas** umas das outras, ou seja, não se sobrepõem.
  - E existe uma **função discriminante adequada para o problema**, mesmo que não seja um **hiperplano**.
- **Separador perfeito**: com erro de classificação igual a zero, ou seja, todos os exemplos são perfeitamente classificados.
- Porém, na prática essa situação não é muito comum.

# Classificação com limiar de decisão rígido



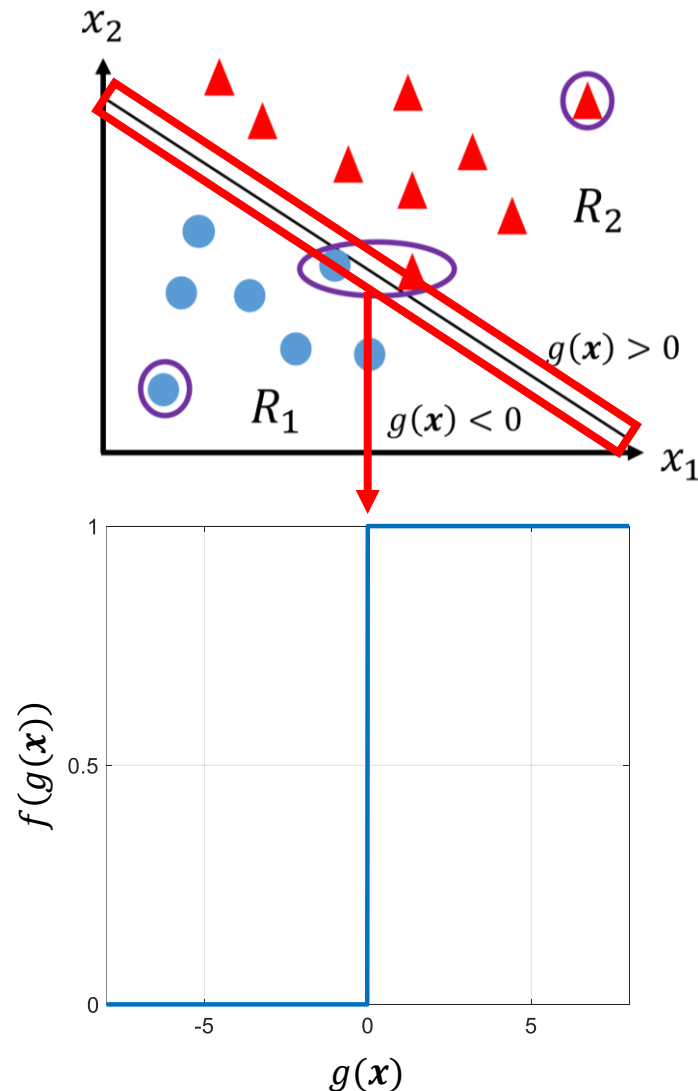
- Quando as classes se sobrepõem, a **regra de aprendizagem do perceptron** falha em convergir para uma solução perfeita.
- Nesse caso, a regra não converge para uma solução **estável** para **valores fixos do passo de aprendizagem**,  $\alpha$ , assim como acontece com o GDE.
- Porém, se  $\alpha$  decresce de acordo com as iterações de treinamento, então a regra tem uma chance de convergir para uma solução de erro mínimo quando os exemplos são apresentados de forma aleatória.
- Podemos também usar o **early-stop** e utilizar os **pesos** que resultaram no menor erro de validação.

# Classificação com limiar de decisão rígido



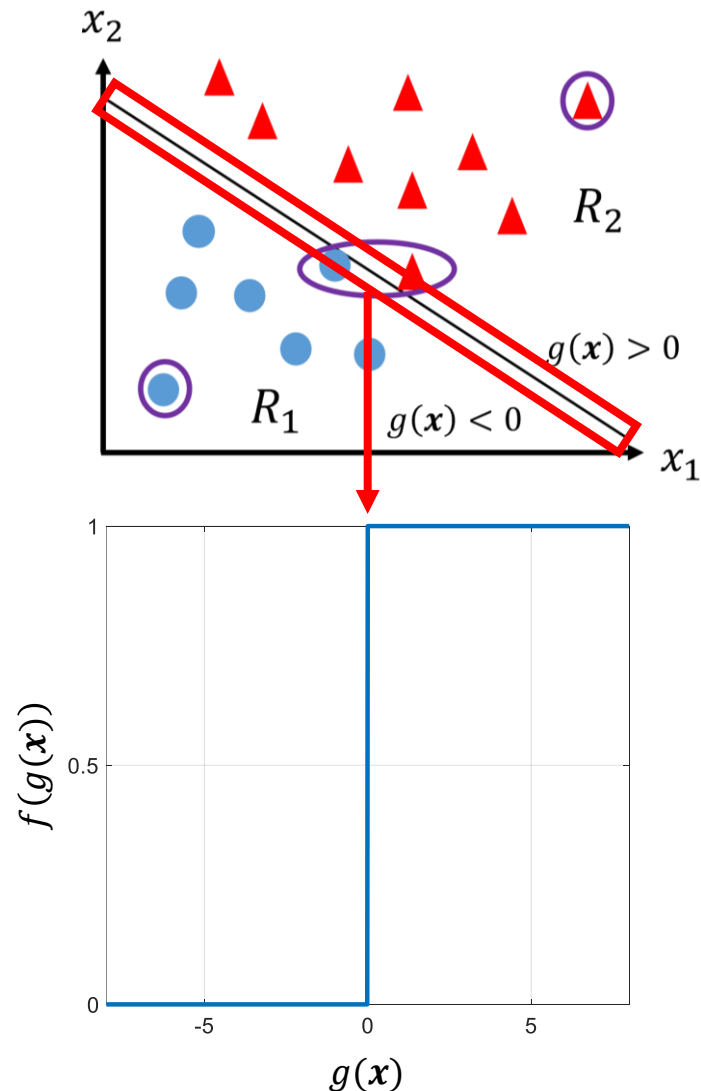
- Outro problema com classificadores que usam **limiar de decisão rígido** é a **falta de informação sobre a confiança** quanto a uma classificação.
- Na figura ao lado, dois exemplos estão bem próximos da **fronteira de decisão** enquanto outros dois estão bem distantes dela.
- Como o classificador com **limiar de decisão rígido** classificaria esses exemplos?

# Classificação com limiar de decisão rígido



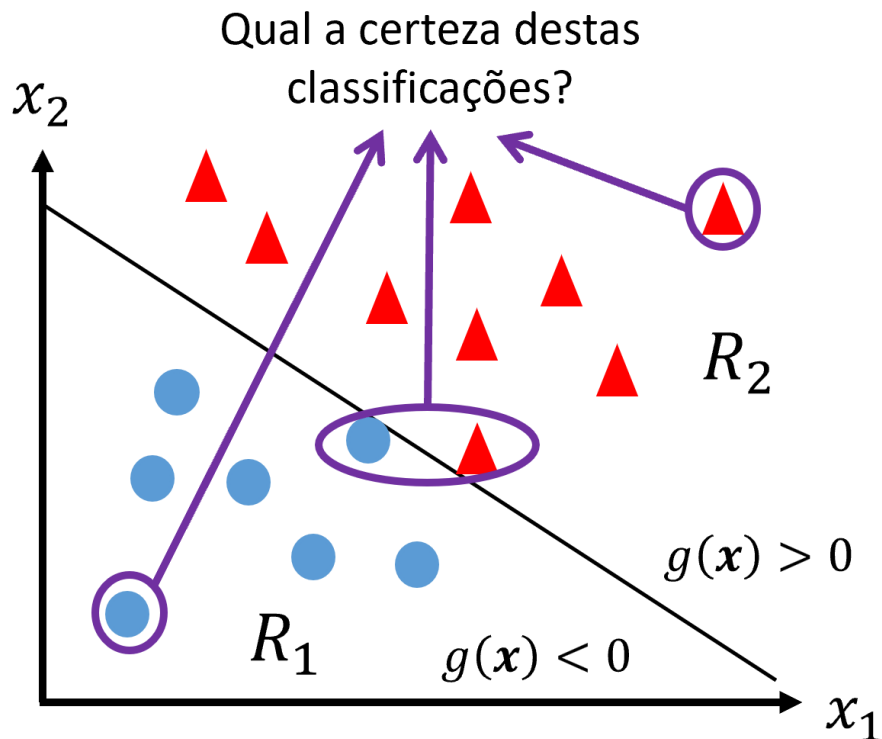
- Olhando para a função de **limiar de decisão rígido**, percebemos que o classificador faz previsões  **muito confiantes**  sempre iguais 0 para  $g(x) < 0$  e iguais a 1 quando  $g(x) > 0$ ,  **independente se o exemplo está distante ou não da fronteira de decisão** .
- Exemplos  **mais distantes da fronteira**  têm uma  **probabilidade maior**  de realmente  **pertencerem à classe da região onde se encontram e não serem outliers** .
  - Quanto maior o valor absoluto de  $g(x)$ , mais distante da fronteira está o exemplo.

# Classificação com limiar de decisão rígido



- Assim, usando **limiar de decisão rígido**, os dois **pontos azuis** seriam classificados como pertencentes à **classe negativa** (valor 0) e os dois **triângulos vermelhos** classificados como pertencentes à **classe positiva** (valor 1), mesmo tendo valores **absolutos de  $g(x)$  bem diferentes**.
  - Pontos muito próximos da fronteira de decisão têm valor absoluto de  $g(x)$  próximo de zero.
  - Já pontos muito distantes têm valor absoluto de  $g(x)$  muito maior do que zero.

# Classificação com limiar de decisão rígido



- Em resumo, *pontos distantes da fronteira* de decisão *deveriam ter uma confiança* (ou probabilidade) de *pertencerem a uma determinada classe bem maior do que pontos próximos*.
- Porém, isso não é refletido na saída do classificador com limiar de decisão rígido.
- Entretanto, em muitas situações, nós precisamos de previsões mais graduadas, que indiquem incertezas quanto à predição.

# Tarefas

- **Quiz:** “*T320 - Quiz - Classificação (Parte II)*” que se encontra no MS Teams.
- **Exercício Prático:** [\*\*Laboratório #2\*\*](#).
  - 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!



