

CLASSIFICADORES PARAMÉTRICOS

André Gustavo Adami
Daniel Luis Notari

INTRODUÇÃO

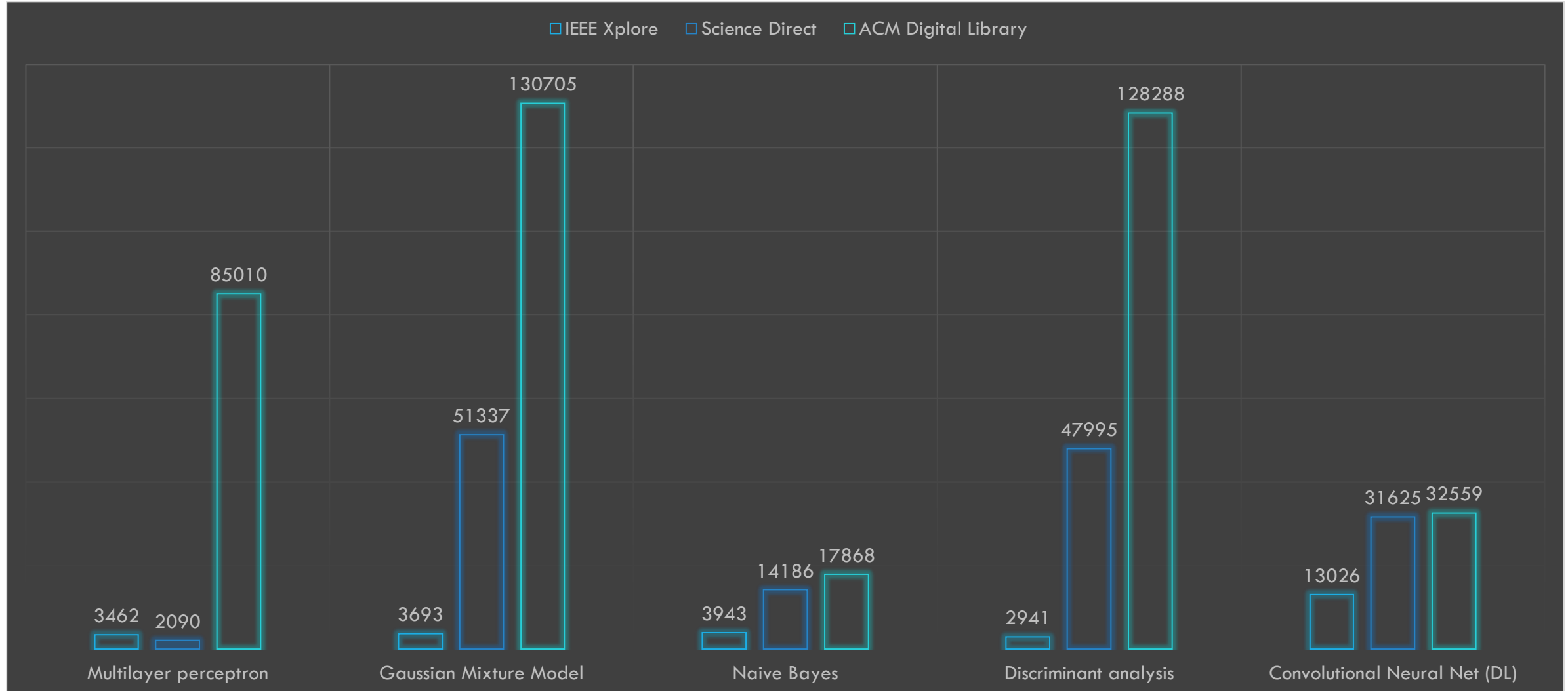
O métodos paramétricos tem uma preferência devido ao fato que o número de parâmetros é fixo (independe da quantidade de amostras de treinamento)

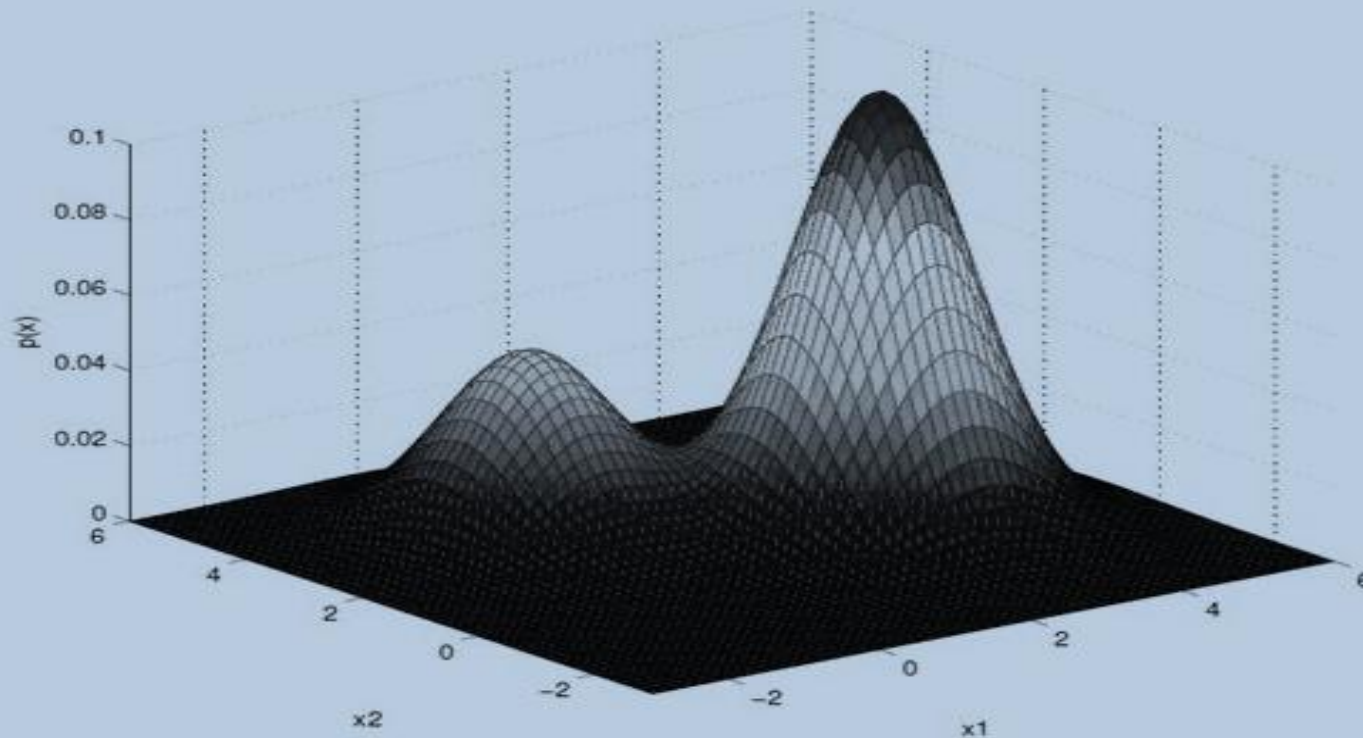
Após a estimação do modelo a partir de um conjunto de dados de treinamento, estes dados não são mais necessários

Vamos conhecer dois modelos que são comumente utilizados

- Modelo de Misturas Gaussianas (*Gaussian Mixture Model – GMM*)
- Perceptron Multicamadas (*Multilayer Perceptron – MLP*)

PUBLICAÇÕES NOS ÚLTIMOS 5 ANOS (2016-2021)





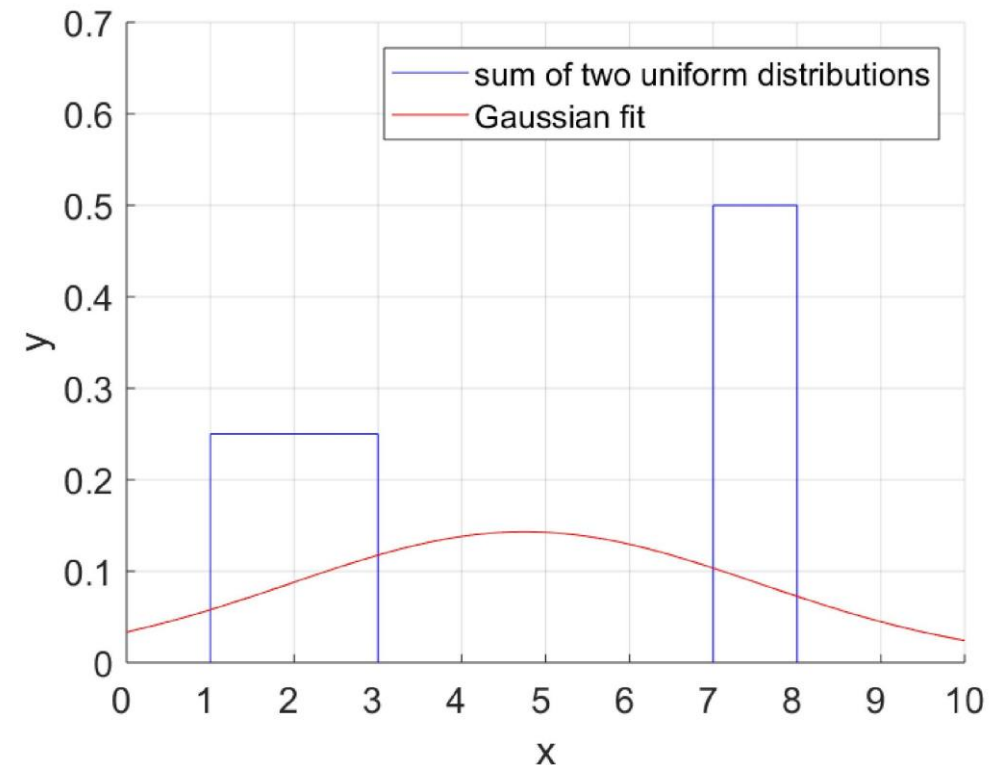
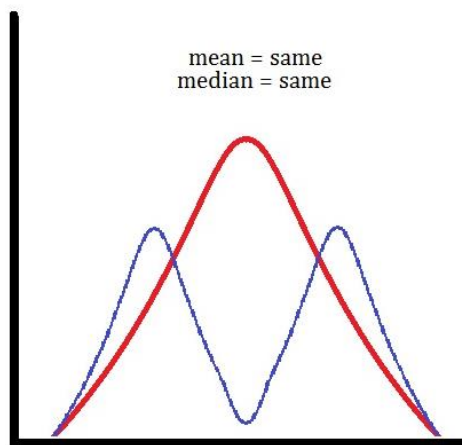
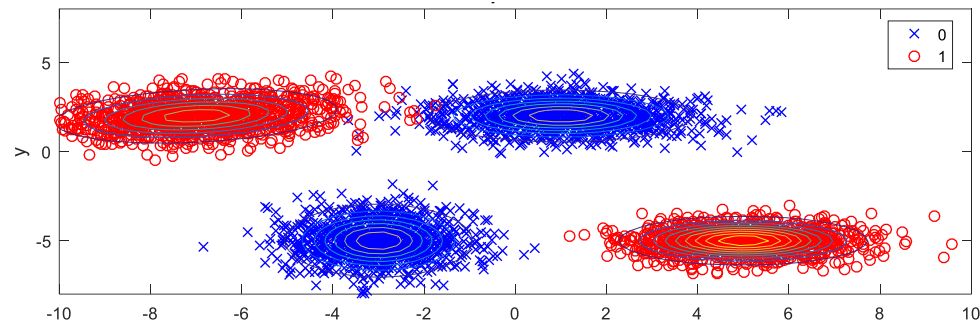
GAUSSIAN MIXTURE MODEL

CronJ

MODELO DE MISTURAS GAUSSIANAS

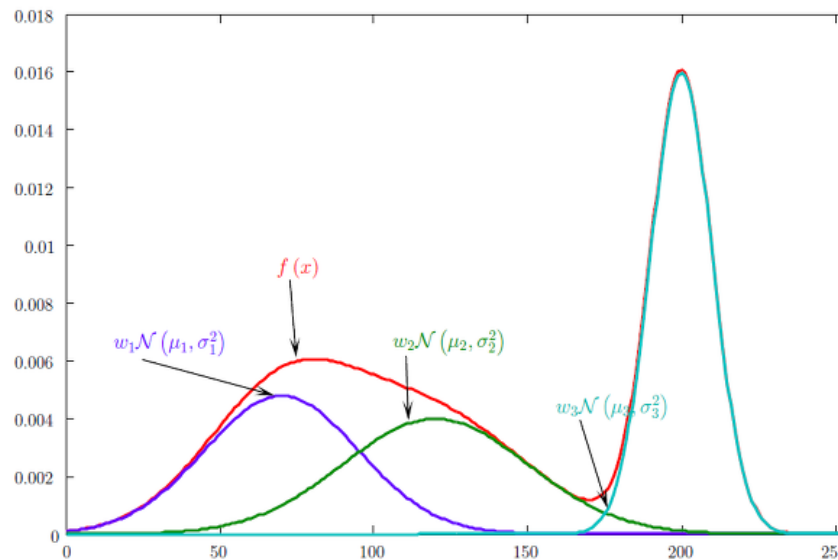
MODELO DE MISTURAS

As funções densidade de probabilidade multimodais (isto é, possuem diversos picos – máximos locais) apresentam um problema para as funções unimodais como Gaussiana, Bernoulli, Poisson, ...



MODELO DE MISTURAS

Para contornar este problema é possível modelar a distribuição utilizando uma combinação linear de funções densidade de probabilidade



A flexibilidade do método de modelagem permite aproximar teoricamente qualquer tipo de distribuição (desde que existe um número suficiente de gaussianas)

MODELO DE MISTURAS GAUSSIANNAS

Um modelo de misturas Gaussianas (*Gaussian Mixture Model* – GMM) é uma função densidade de probabilidade paramétrica representada como uma soma ponderada de M densidades dos componentes Gaussianos

$$p(\mathbf{x}|\Theta) = \sum_{i=1}^M \alpha_i f(\mathbf{x}|\boldsymbol{\mu}_i, \Sigma_i)$$

onde \mathbf{x} é um vetor d -dimensional de variáveis contínuas, α_i , $i = 1, \dots, M$, são as proporções da mistura (normalização ou contribuição),

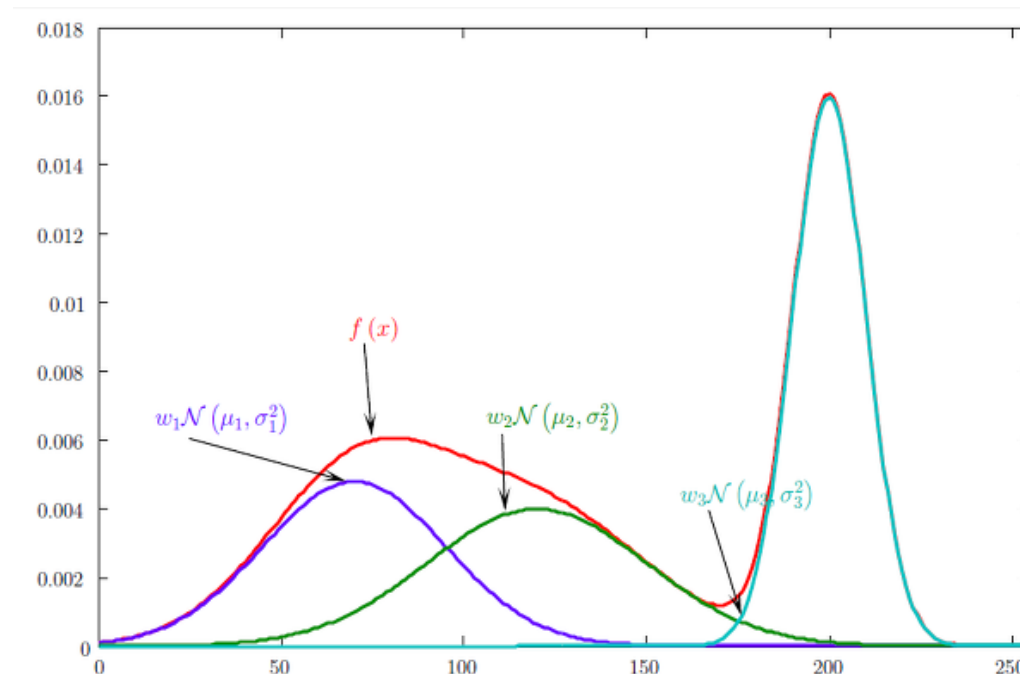
$$\sum_{i=1}^M \alpha_i = 1, \quad \alpha_i > 0$$

e $f(\mathbf{x}|\boldsymbol{\mu}_i, \Sigma_i)$, $i = 1, \dots, M$, são as densidades para cada componente

MODELO DE MISTURAS GAUSSIANAS

Como cada componente é uma função densidade Gaussiana, então a função pode ser reescrita como

$$p(\mathbf{x}|\Theta) = \sum_{i=1}^M \alpha_i \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$



MODELO DE MISTURAS GAUSSIANNAS

Portanto, o modelo de misturas Gaussianas é parametrizado pelos vetores média, matrizes de covariâncias e a proporção das misturas de todas as densidades

$$\Theta = \{\alpha_i, \mu_i, \Sigma_i\} \quad i = 1, \dots, M$$

Assim, este modelo possui como parâmetros M proporções $M \times d$ médias e $M \times d \times (d + 1)/2$ variâncias (matriz de covariância completa)

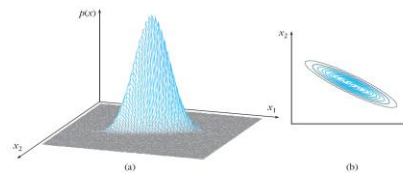
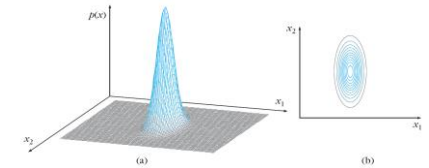
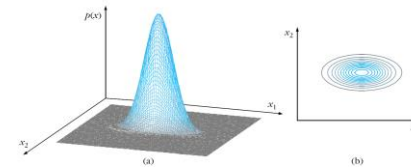
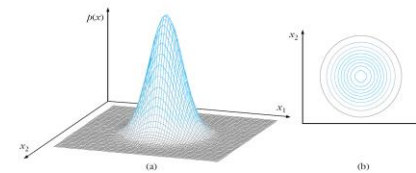
- Como as matrizes de covariância demandam muitos dados para serem estimadas, assume-se que as características são independentes e portando a matriz de covariância possui somente valores na diagonal principal: M proporções $M \times d$ médias e $M \times d$ variâncias

MODELO DE MISTURAS GAUSSIANAS: HIPERPARÂMETROS

Número de componentes gaussianos ($M=?$)

Tipo de Matriz de Covariância Σ_k

- Σ_k é diagonal ($\sigma_{ij} = 0, i \neq j$) e as variâncias são todas iguais ($\sigma_{ii} = \lambda$) – **Esférica**
- Σ_k é diagonal ($\sigma_{ij} = 0, i \neq j$) e as variâncias são diferentes – **Diagonal**
- Σ_k é o caso geral – **Completa**



Além disso, é possível ter a mesma matriz para todos os componentes (**tied-mixture**)

MODELO DE MISTURAS GAUSSIANAS NO R

O mclust é um pacote para agrupamento baseado em modelo, classificação e estimação de densidade baseado na modelagem de utilizando misturas gaussianas

```
> library(mclust)
```

The logo for the mclust R package, consisting of a stylized, abstract pattern of red lines forming a series of interconnected, elongated shapes that resemble a cluster or a network.

version 5.4.7
Type 'citation("mclust")' for citing this R package in publications.

MODELO DE MISTURAS GAUSSIANNAS: MCLUST

Para permitir um maior controle das características da matriz de covariância, pode-se utilizar a seguinte decomposição da matriz de covariância

$$\Sigma_k = \lambda_k D_k A_k D_k^T$$

onde

λ_k é um escalar que controla o volume do elipsoide

A_k é uma matriz diagonal, cujos elementos são proporcionais aos autovalores, que especifica a forma dos contornos da densidade com $\det(A_k) = 1$

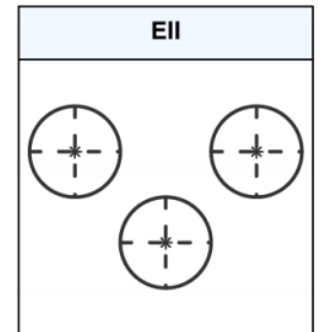
D_k é uma matriz ortogonal de autovetores que determina a orientação do elipsoide

MODELO DE MISTURAS GAUSSIANAS: MCLUST

Esféricas

- Um único valor de variância é estimado para todas as variáveis para todos os componentes (EII) ou individualmente (VII)
- Para ambos os modelos, são necessários $M - 1$ proporções + $M \times d$ médias
- Para o modelo EII, é necessário estimar somente 1 variância (independentemente da dimensionalidade)
- Para o modelo VII, é necessário estimar M variâncias (1 para cada componente)
- O total de parâmetros deve ser ainda multiplicado pelo número de classes, pois é um modelo para cada classe

Modelo	Σ_k	Volume	Forma	Orientação
EII	λI	Igual	Igual	—
VII	$\lambda_k I$	Variável	Igual	—

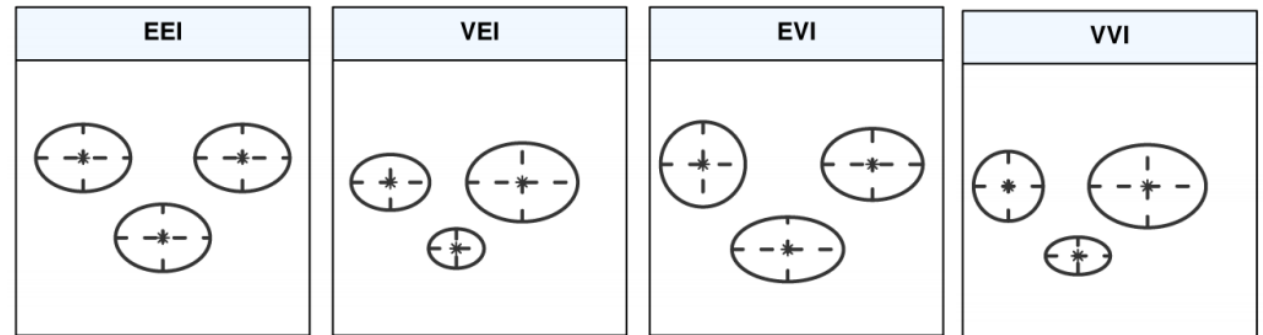


MODELO DE MISTURAS GAUSSIANAS: MCLUST

Diagonais

- Os autovetores são os próprios eixos do espaço de características
- Combinação dos autovalores e coeficientes de volume dos elipsoides
- A variância para cada variável é estimada para todos os componentes ou individualmente
- “VVI” é o que a literatura trata como matriz de covariância diagonal

Modelo	Σ_k	Volume	Forma	Orientação
EEI	λA	Igual	Igual	Eixos das coordenadas
VEI	$\lambda_k A$	Variável	Igual	Eixos das coordenadas
EVI	λA_k	Igual	Variável	Eixos das coordenadas
VVI	$\lambda_k A_k$	Variável	Variável	Eixos das coordenadas

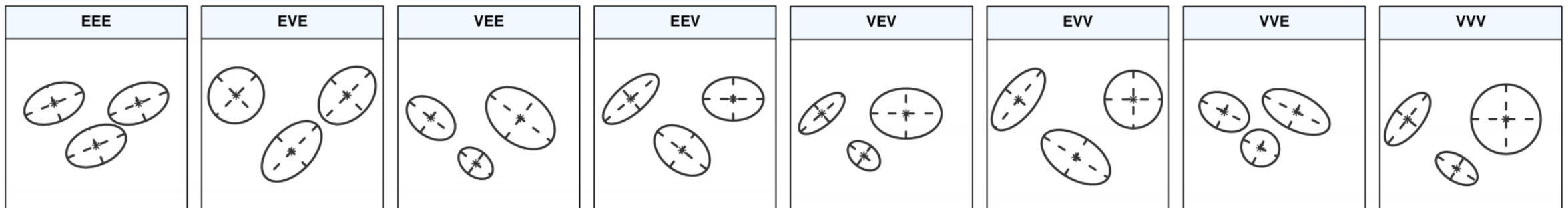


MODELO DE MISTURAS GAUSSIANAS: MCLUST

Completa

- Combinação dos autovalores, autovetores e coeficientes de volume dos elipsoides
- Compartilhamento dos termos da decomposição
- “VVV” é o que a literatura trata como matriz de covariância completa

Modelo	Σ_k	Volume	Forma	Orientação
EEE	$\lambda D A D^T$	Igual	Igual	Igual
EVE	$\lambda D A_k D^T$	Igual	Variável	Igual
VEE	$\lambda_k D A D^T$	Variável	Igual	Igual
VVE	$\lambda_k D A_k D^T$	Variável	Variável	Igual
EEV	$\lambda D_k A D_k^T$	Igual	Igual	Variável
VEV	$\lambda_k D_k A D_k^T$	Variável	Igual	Variável
EVV	$\lambda D_k A_k D_k^T$	Igual	Variável	Variável
VVV	$\lambda_k D_k A_k D_k^T$	Variável	Variável	Variável



MODELO DE MISTURAS GAUSSIANAS: MCLUST

Número de parâmetros (proporções, médias e variâncias) de cada modelo (6 variáveis e 4 componentes)

```
> mapply(nMclustParams, mclust.options("emModelNames"), d = 6, G = 4)
```

```
EII VII EEI VEI EVI VVI EEE VEE EVE VVE EEV VEV EVV VVV  
28 31 33 36 48 51 48 51 63 66 93 96 108 111
```

```
> mapply(nMclustParams, "VVV", d = 6, G = 4)
```

```
VVV
```

```
111
```

Número de parâmetros de variância (6 variáveis e 4 componentes)

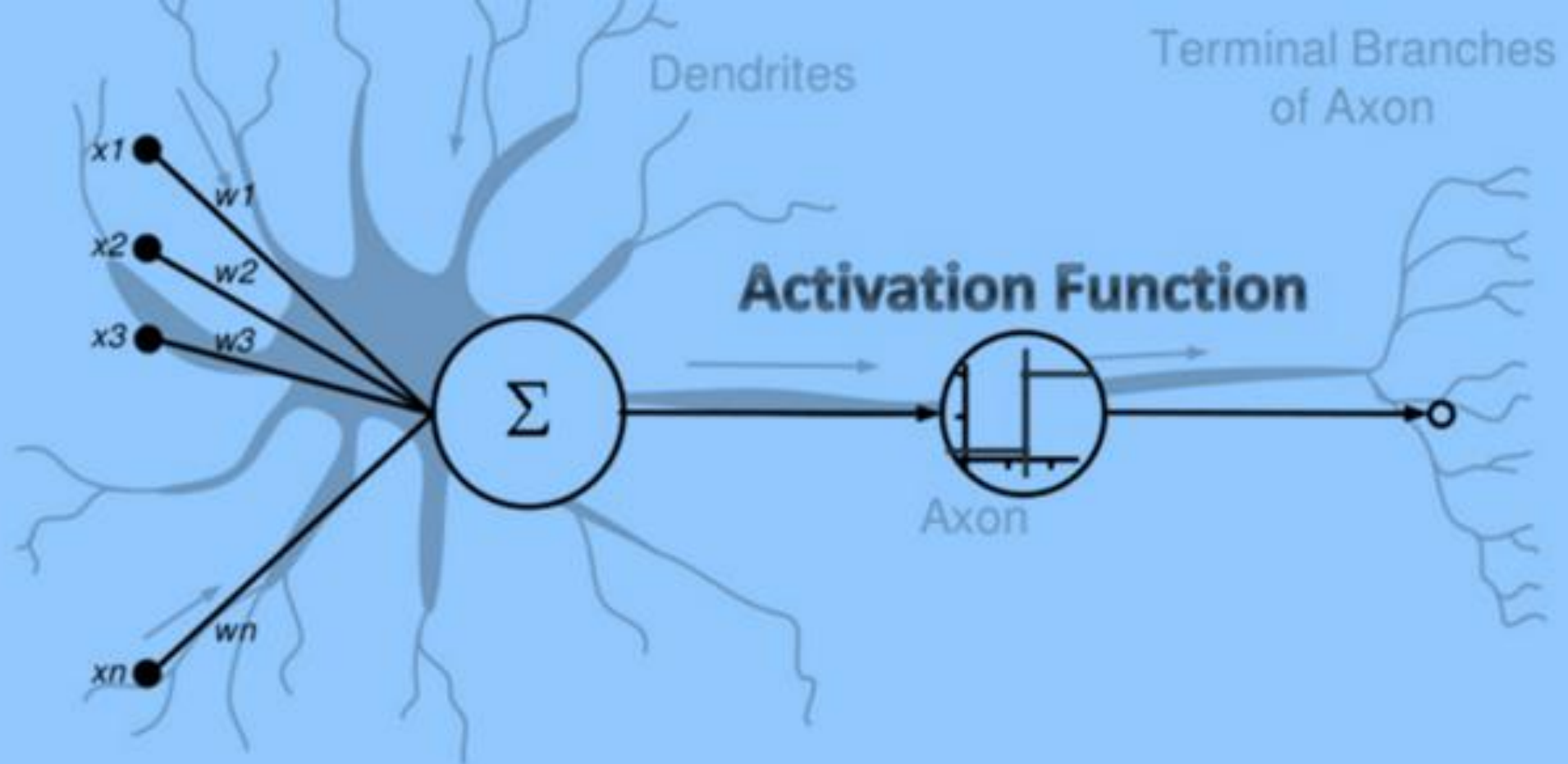
```
> mapply(nVarParams, mclust.options("emModelNames"), d = 6, G = 4)
```

```
EII VII EEI VEI EVI VVI EEE VEE EVE VVE EEV VEV EVV VVV  
1 4 6 9 21 24 21 24 36 39 66 69 81 84
```

```
> mapply(nVarParams, "VVV", d = 6, G = 4)
```

```
VVV
```

```
84
```

REDES NEURAIS ARTIFICIAIS

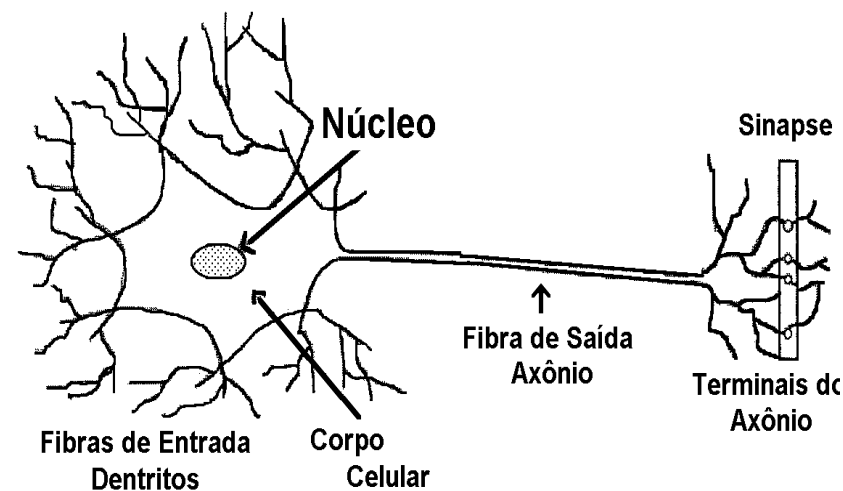
REDES NEURAIS ARTIFICIAIS

“Cérebro é um computador altamente complexo, não-linear e paralelo” (Haykin, 2001)

Uma rede neural é um processador paralelo distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível ao uso

Semelhante ao cérebro humano pois o conhecimento é adquirido via aprendizagem e as forças das conexões entre os neurônios (pesos sinápticos) são utilizadas no armazenamento do conhecimento adquirido

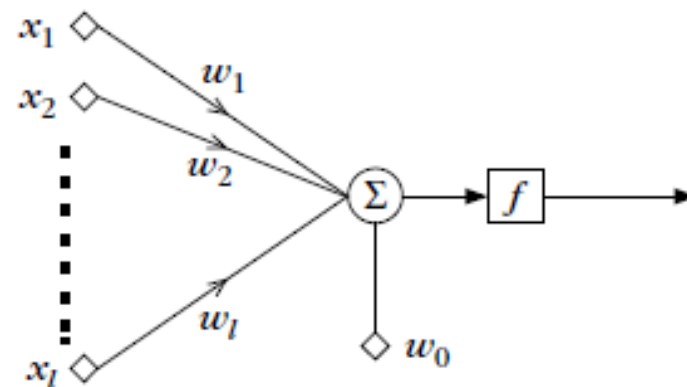
Um neurônio é uma unidade de processamento de informação fundamental à operação de uma rede neural



REDES NEURAIS ARTIFICIAIS

O neurônio perceptron implementa uma combinação linear dos pesos sinápticos \mathbf{w} (w_1, w_2, \dots, w_n) com os dados de entrada \mathbf{x} (x_1, x_2, \dots, x_n)

$$\mathbf{w}^T \mathbf{x} + w_0$$



O resultado da combinação linear passa por um estágio não linear, que implementa a função de ativação f , que restringe a amplitude da saída do neurônio

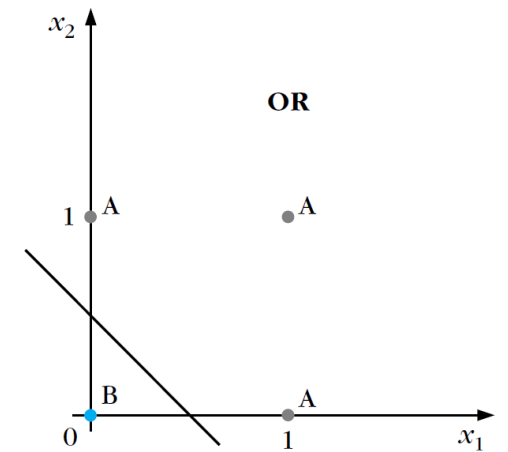
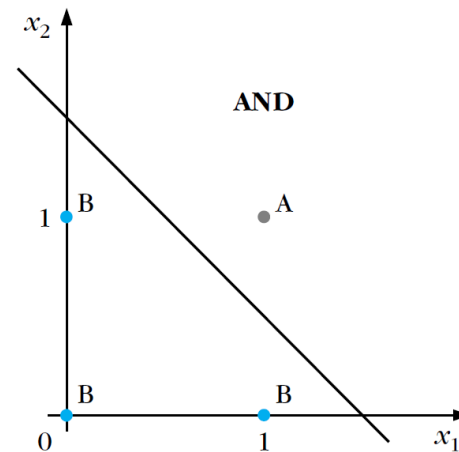
- A escolha mais comum é a função limiar: $f(x) = 1, se x > 0$ e $f(x) = -1, se x < 0$

REDES NEURAIS ARTIFICIAIS

Assim, a classificação baseia-se na seguinte regra

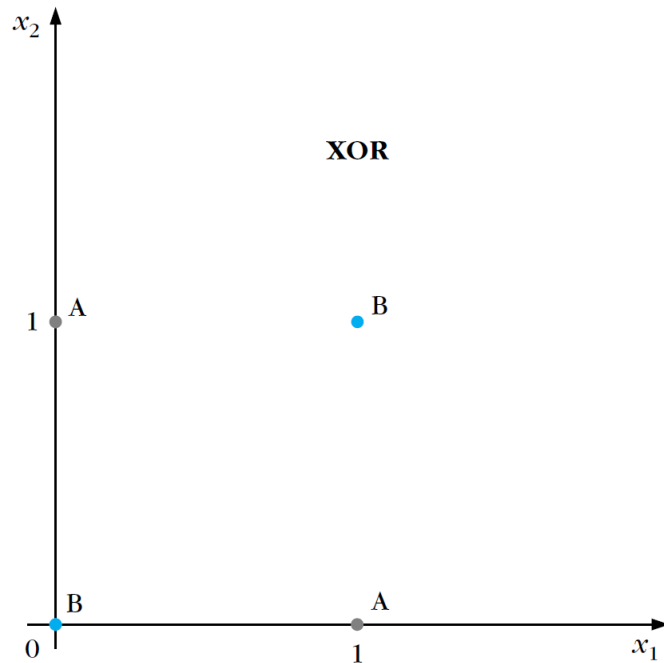
$$\begin{aligned} \mathbf{w}^T \mathbf{x} + w_0 &> 0, & \mathbf{x} &\in \omega_1 \\ \mathbf{w}^T \mathbf{x} + w_0 &< 0, & \mathbf{x} &\in \omega_2 \end{aligned}$$

O algoritmo perceptron pode ser aplicado em problemas do tipo das funções booleanas AND e OR, que são **linearmente separáveis**



REDES NEURAIS ARTIFICIAIS

Em problemas que não são linearmente separáveis (por exemplo o do XOR), o algoritmo perceptron falhará



x_1	x_2	XOR	Classe
0	0	0	B
0	1	1	A
1	0	1	A
1	1	0	B

Para resolver isso, pode-se pensar em utilizar múltiplos hiperplanos

REDES NEURAIS ARTIFICIAIS

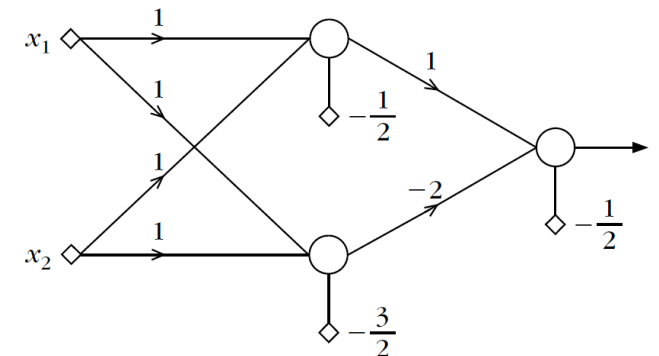
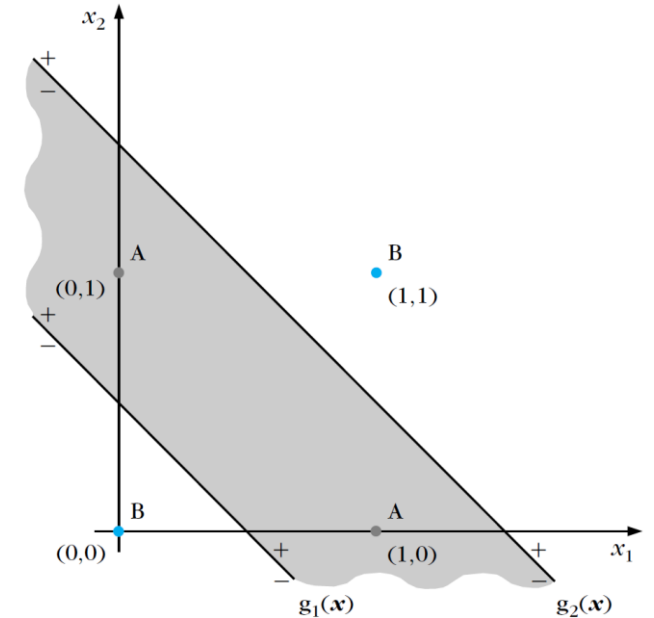
Assim, o uso de duas retas definidas por $g_1(\mathbf{x}) = g_2(\mathbf{x}) = 0$ permite separar as 2 classes

$$g_i(\mathbf{x}) \begin{matrix} > \\ < \end{matrix} 0$$

A implementação de tal solução é através de uma rede neural de 2 camadas

- A primeira camada realiza um mapeamento do vetor de entrada \mathbf{x} a um novo $\mathbf{y} = [y_1, y_2]$
- A decisão na segunda camada é baseada nos dados transformados

O mapeamento da primeira camada transforma um problema não-linearmente separável em um linearmente separável



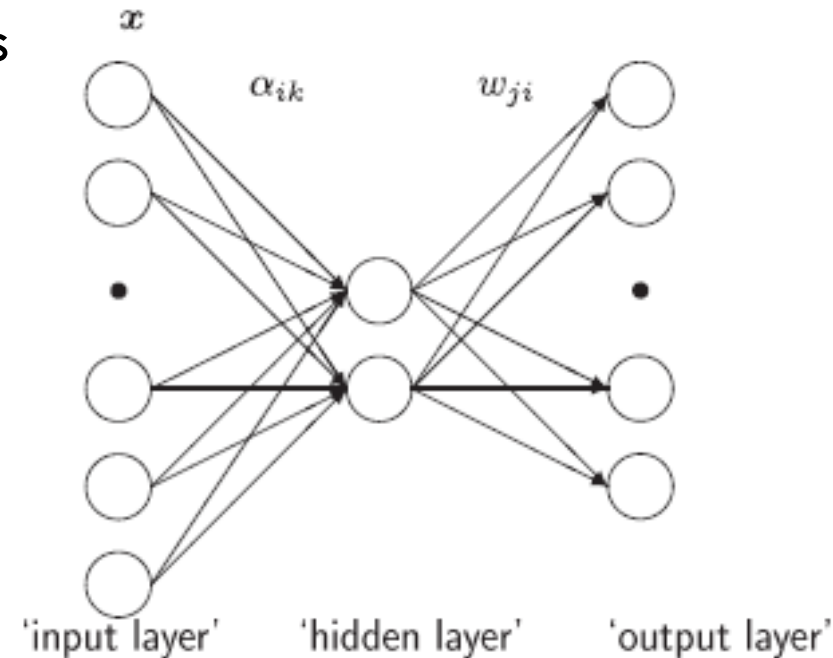
MULTILAYER PERCEPTRON

A rede Multilayer Perceptron (MLP) é uma rede de múltiplas camadas que produz uma transformação de um padrão $\mathbf{x} \in \mathbb{R}^d$ em um espaço com n' dimensões

$$g_j(\mathbf{x}) = \sum_{i=1}^m w_{ji} \phi_i(\alpha_i^T \mathbf{x} + \alpha_{i0}) + w_{j0}, j = 1, \dots, n'$$

onde ϕ_i são funções contínuas diferenciáveis (não-lineares) da família sigmoïdal

$$\phi_i(y) = \phi(y) = \frac{1}{1 + \exp(-y)}$$



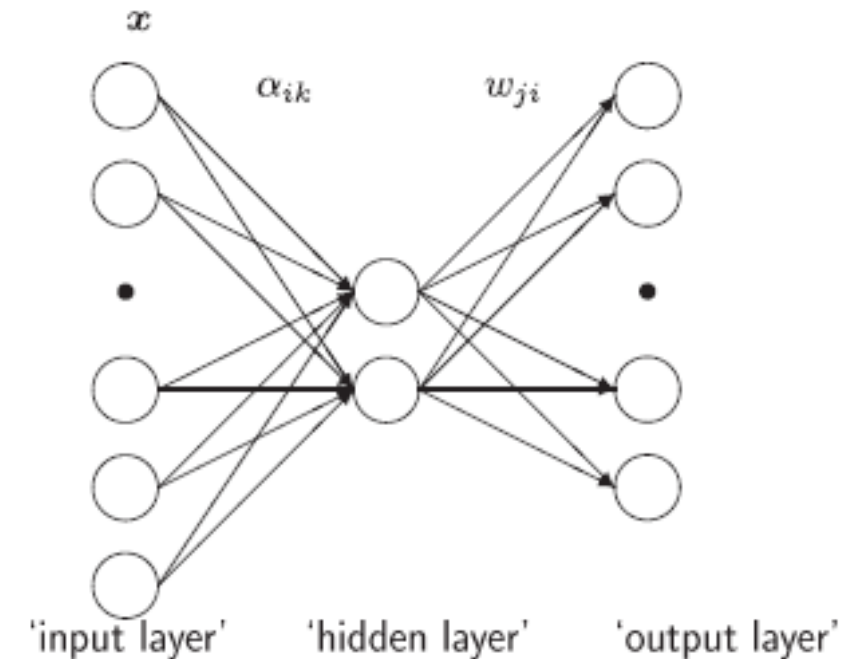
A MLP

- Projeta os dados em cada uma das m direções descritas pelo vetor $\alpha_i = (\alpha_{i1}, \dots, \alpha_{id})$
- Transforma os dados projetados (deslocados por um bias α_{i0}) pela função não linear $\phi_i(y)$
- Realiza uma combinação linear utilizando os pesos w_{ji} (deslocados por um bias w_{j0})

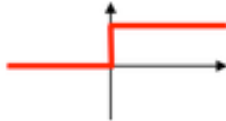
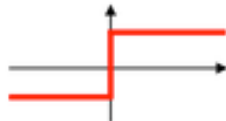
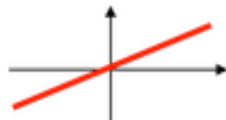
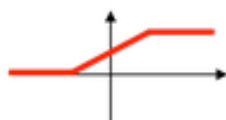
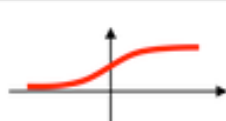
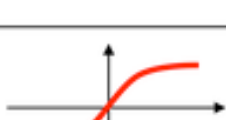
MULTILAYER PERCEPTRON

Tipos de camada

- **Entrada** (1 camada): os nodos de entrada recebem o vetor ou padrão de entrada. O número de nodos é igual ao número de dimensões do vetor de características
- **Escondida ou Oculta** (1+ camada): existem pesos associados com a ligação dos nodos de entrada com os nodos escondidos, os quais realizam a combinação linear $y = \alpha_i^T x + \alpha_{i0}$ e a transformação não-linear $\phi(y)$. As ligações entre as camadas escondidas possuem o mesmo tipo de peso, mas das saídas da camada anterior. O número de camadas e o número de nodos são alguns dos parâmetros da rede MLP
- **Saída** (1 camada): os nodos de saída realizam uma combinação linear das saídas dos nodos escondidos e entregam estes como saídas da rede. O número de saídas geralmente é igual ao número de classes a serem reconhecidas



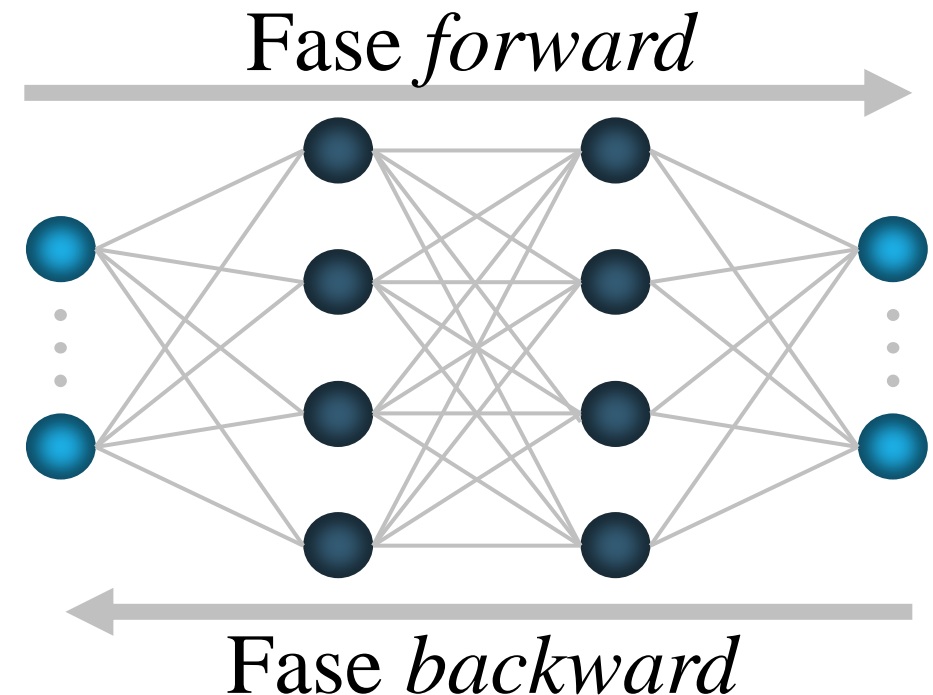
MULTILAYER PERCEPTRON: FUNÇÕES DE ATIVAÇÃO

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer NN	

MULTILAYER PERCEPTRON: TREINAMENTO

O treinamento de uma rede neural MLP é realizado de maneira supervisionada com o algoritmo *Backpropagation* (ou retropropagação do erro)

Nesse algoritmo, a determinação do erro é um processo recursivo que se inicia nos neurônios da camada de saída e vai até os neurônios da primeira camada intermediária



MULTILAYER PERCEPTRON: VANTAGENS X DESVANTAGENS

Vantagens

- Simples de implementar
- Boa capacidade de generalização

Desvantagens

- Dificuldade de justificar as respostas
- Custo computacional significativo
- Baixa velocidade de aprendizado

MULTILAYER PERCEPTRON: HIPERPARÂMETROS

Camada Escondida

- Número de camadas (1 a 2 camadas)
- Número de neurônios (entre o tamanho da camada de entrada e da saída, $2/3$ da camada de entrada + tamanho da camada de saída, menor que o dobro do que a camada da entrada, 2 vezes o número de classes,...)
- Função de ativação

Camada de Saída

- Número de neurônios (geralmente utilizam 1 neurônio para cada classe)
- Função de ativação

REDES NEURAIS NO CARET

Para saber mais que modelos são disponibilizados pelo pacote caret, consulte

https://topepo.github.io/caret/train-models-by-tag.html#Neural_Network

Outros pacotes incluem (mas também são utilizados pelo caret)

- 1 (somente 1 camada escondida)
- keras
- neuralnet
- mxnet
- mlp