

MÉTODOS NÃO-LINEARES E FUSÃO DE DADOS

Me. José Vinícius Ribeiro

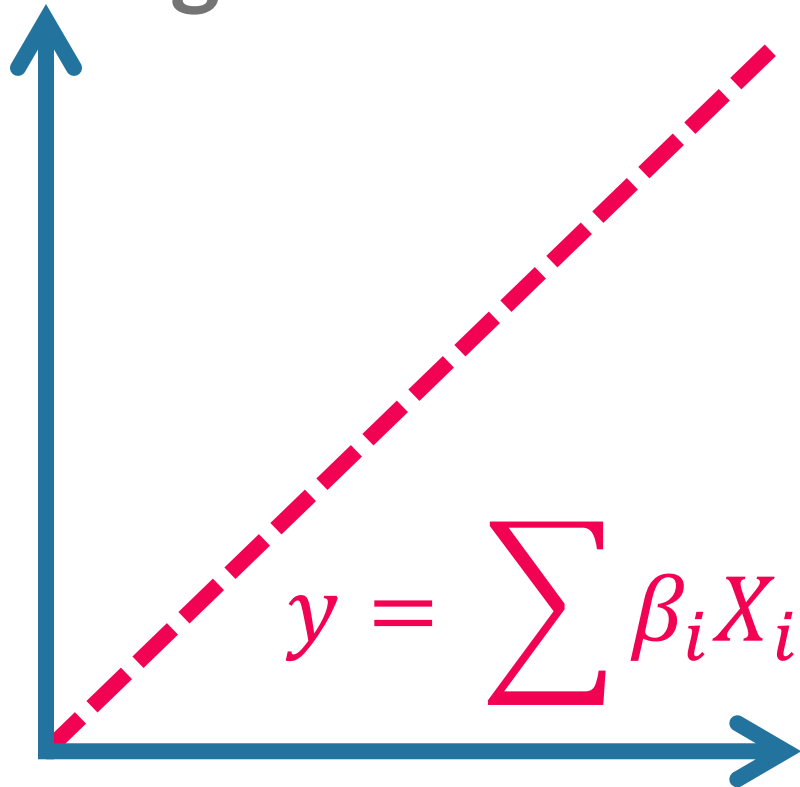
2FIS446

SUMÁRIO

- Linearidade x Não-Linearidade
- Random Forest
- Redes Neurais
- Fusão de dados
- Prática com python no google colab

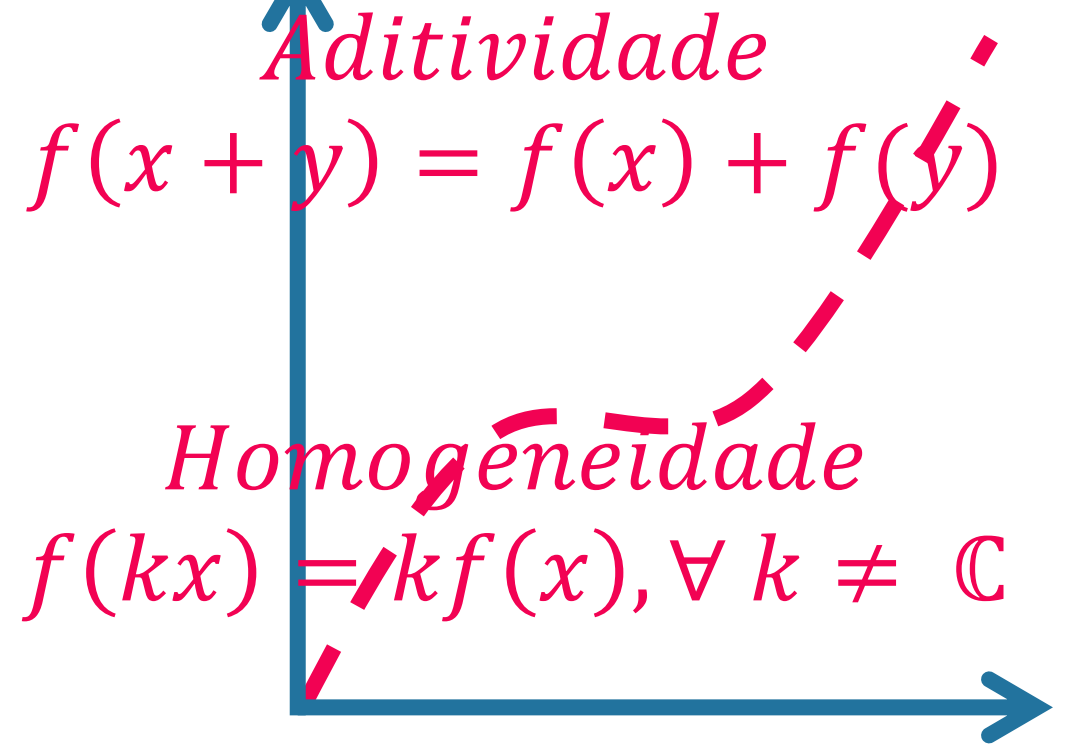
LINEARIDADE X NÃO-LINEARIDADE

Algoritmo linear



A relação entre as variáveis (matriz X) e o target (vetor y) pode ser expressa como uma combinação linear de variáveis. Apenas operações lineares.

Algoritmo não-linear



Pressuposto de que a relação entre o target e as variáveis é mais complexa. Muitas possibilidades de novas operações

PRINCIPAIS ALGORITMOS ATUALMENTE

Lineares

- Regressão Linear
- Regressão Linear Múltipla
- Regressão Logística
- Regressão Linear por Mínimos Quadrados (PLS)
- Análise discriminante por Mínimos Quadrados (PLS-DA)
- Regressão de Ridge
- Análise Discriminante Linear (LDA)
- Máquina de Vetores de Suporte
- Redes Neurais Clássicas

Não-Lineares

- Árvores de Decisão
- Floresta Aleatória
- *XGBoost*
- *Naive Bayes*
- Máquina de Vetores de Suporte
- K-ésimo Vizinho mais Próximo
- *Cubist*
- Redes Neurais Clássicas
- Redes Neurais Convolucionais
- Redes Neurais Recorrentes

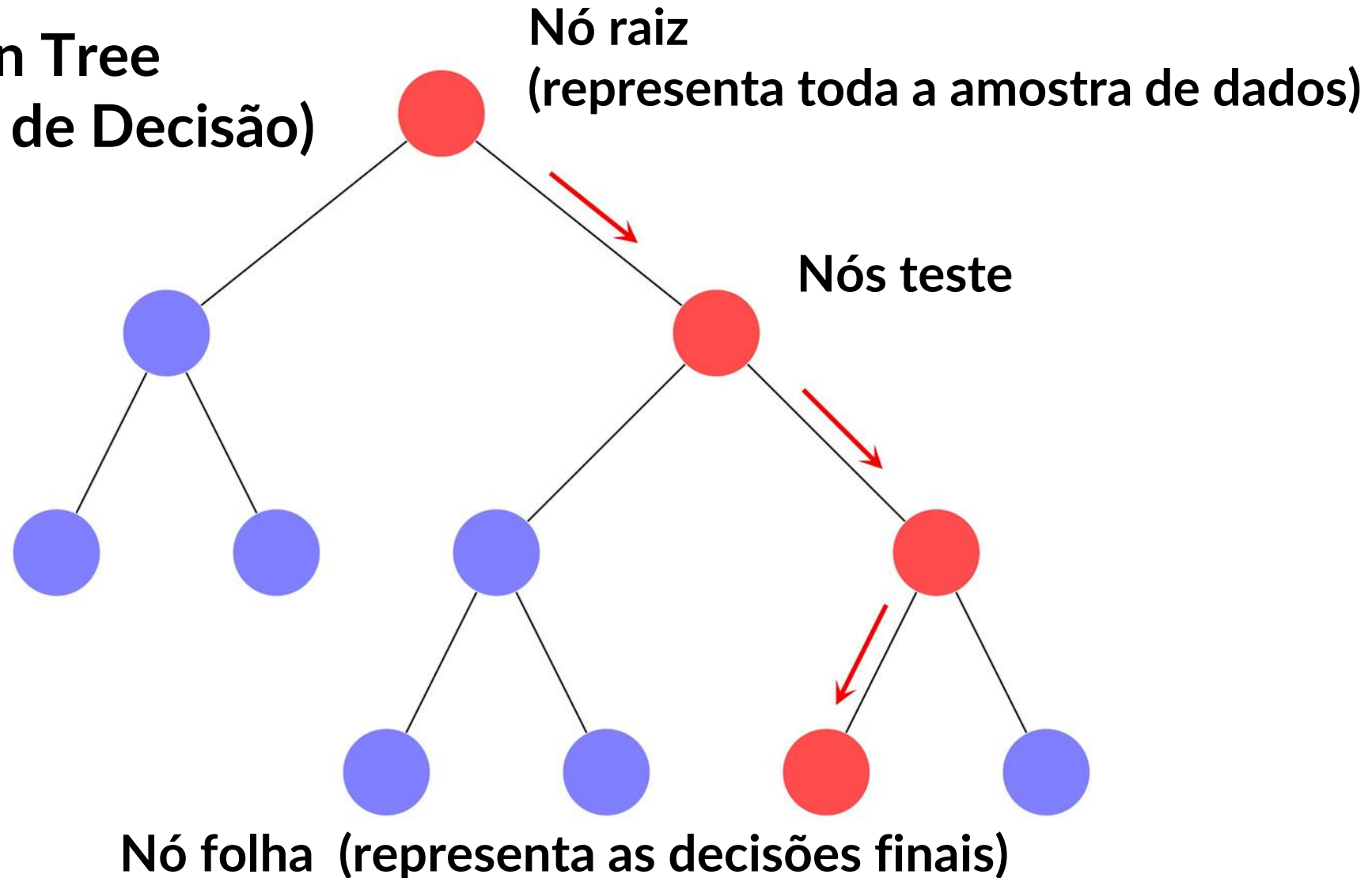
RANDOM FOREST

Random Forest (Floresta Aleatória)

O principal método baseado em *ensembles*

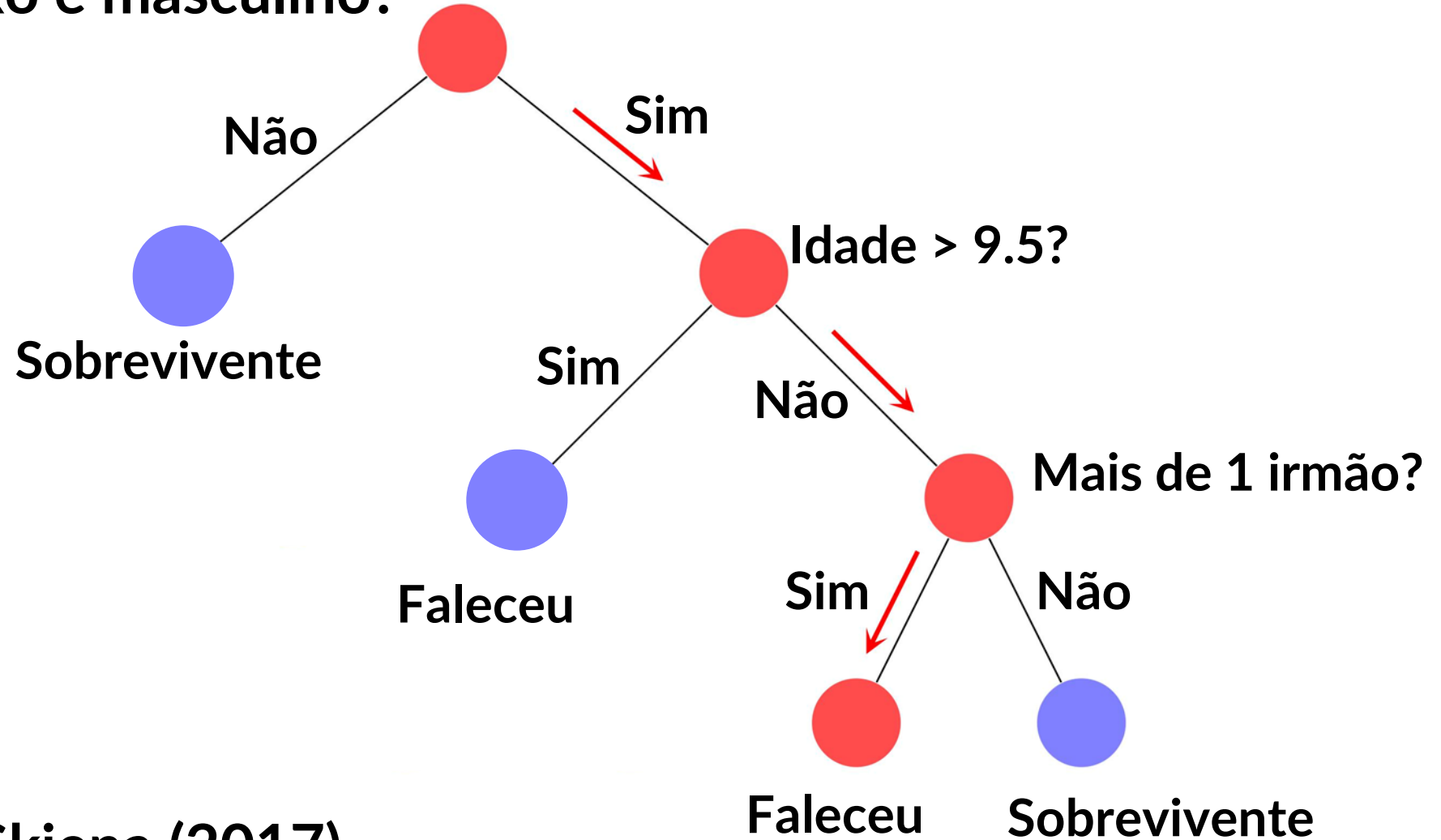
Baseado em árvores de decisão

Decision Tree
(Árvore de Decisão)



Random Forest (Floresta Aleatória)

O sexo é masculino?

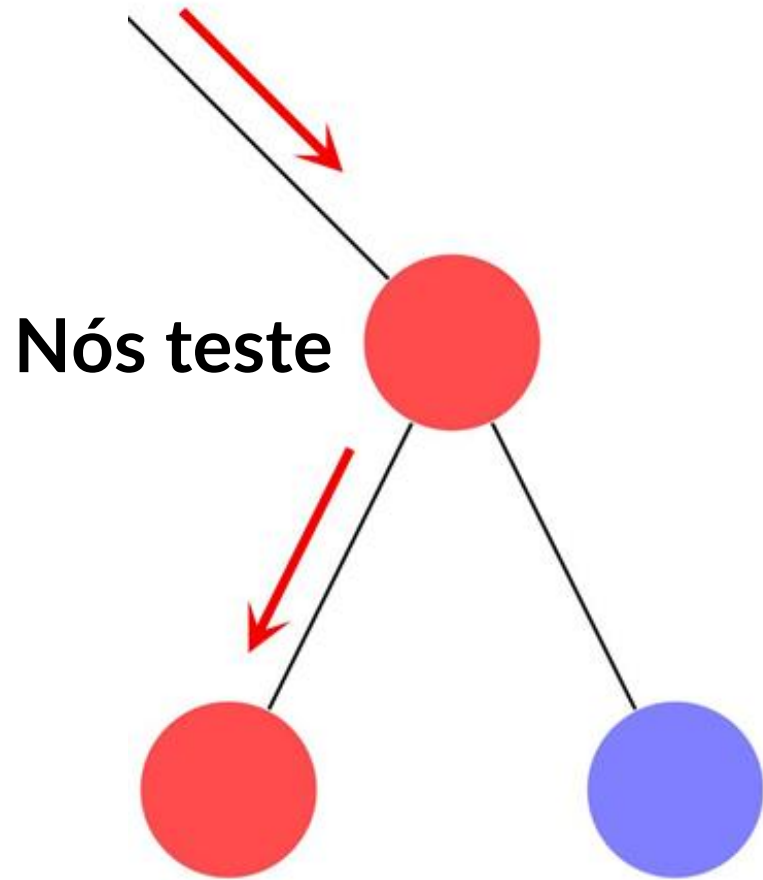


Exemplo Titanic Skiena (2017)

SKIENA, Steven S. The data science design manual. Springer, 2017

Random Forest (Floresta Aleatória)

Floresta Aleatória (Random Forest) – Classificação e Regressão



Portanto, existem *variáveis* que carregam mais informação (facilitam a divisão de classes) do que outras

Parâmetros de pureza para identificá-las

- Entropia, Índice de Gini, Erros Quadráticos Médios, Erros Absolutos Médios

Random Forest (Floresta Aleatória)

Exemplo numérico: modelo de regressão

$$X_{ij} = (X_{11}, X_{12}, \dots, X_{1n}, X_{21}, X_{22}, \dots, X_{2n}, \dots, X_{nm})$$

$$y_i = (y_1, y_2, y_3, \dots, y_n)$$

n = amostras

m = variáveis

Suponha a configuração para o modelo: n=5 e m=1

Uma possível *decision tree* seria

$$X = [1, 2, 3, 4, 5] \text{ e } y = [1.2, 1.9, 3.1, 4.2, 5.0]$$

Random Forest (Floresta Aleatória)

$X=[1, 2, 3, 4, 5]$ e $y=[1.2, 1.9, 3.1, 4.2, 5.0]$

Ponto de divisão $X=2.5$

Métrica de impureza $\rightarrow MSE = \sum_{i=1}^p \frac{(y_i - \bar{y}_i)^2}{p}$

- $X < 2.5$

$$y=[1.2, 1.9] \quad MSE = \frac{(1.2 - 1.55)^2 + (1.9 - 1.55)^2}{2}$$

$$\bar{y}_i=1.55 \quad = 0.1225$$

Random Forest (Floresta Aleatória)

$X=[1, 2, 3, 4, 5]$ e $y=[1.2, 1.9, 3.1, 4.2, 5.0]$

- $X > 2.5$ $y=[3.1, 4.2, 5.0]$ $\bar{y}_i=4.1$

$$MSE = \frac{(3.1 - 4.1)^2 + (4.2 - 4.1)^2 + (5.0 - 4.1)^2}{3}$$
$$= 0.6067$$

Random Forest (Floresta Aleatória)

$X=[1, 2, 3, 4, 5]$ e $y=[1.2, 1.9, 3.1, 4.2, 5.0]$

- MSE geral para a divisão em $X=2.5$

$$\begin{aligned}MSE_{total} &= \frac{2}{5} 0.1225 + \frac{3}{5} 0.6067 \\ &= 0.41302\end{aligned}$$

Random Forest (Floresta Aleatória)

$X=[1, 2, 3, 4, 5]$ e $y=[1.2, 1.9, 3.1, 4.2, 5.0]$

Novo ponto de divisão $X= 3.5$

$$X < 3.5$$

$$y=[1.2, 1.9, 3.1]$$

$$\bar{y}_i=2.067$$

$$MSE = 0.616$$

$$X > 3.5$$

$$y=[4.2, 5.0]$$

$$\bar{y}_i=4.6$$

$$MSE = 0.16$$

Random Forest (Floresta Aleatória)

$X=[1, 2, 3, 4, 5]$ e $y=[1.2, 1.9, 3.1, 4.2, 5.0]$

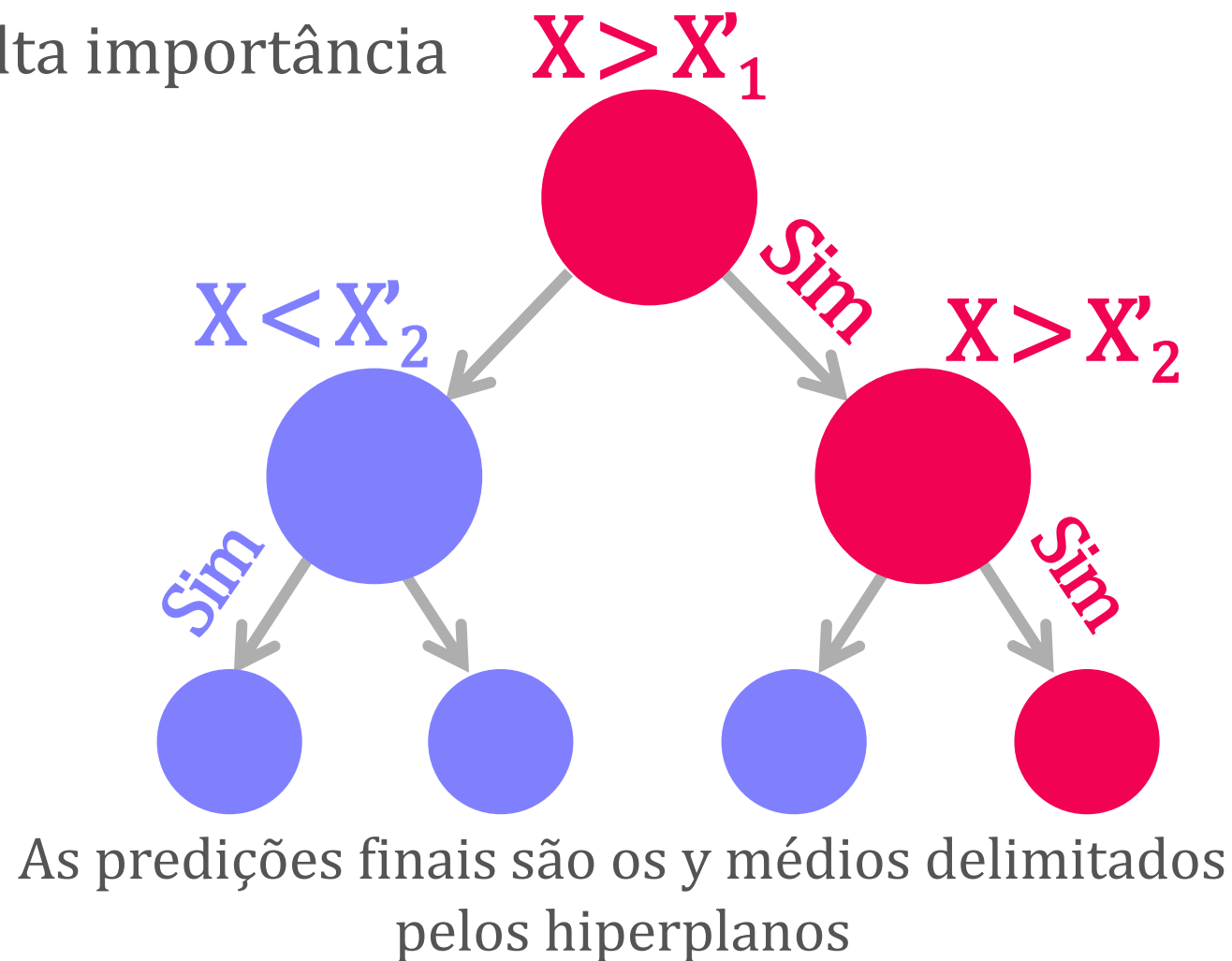
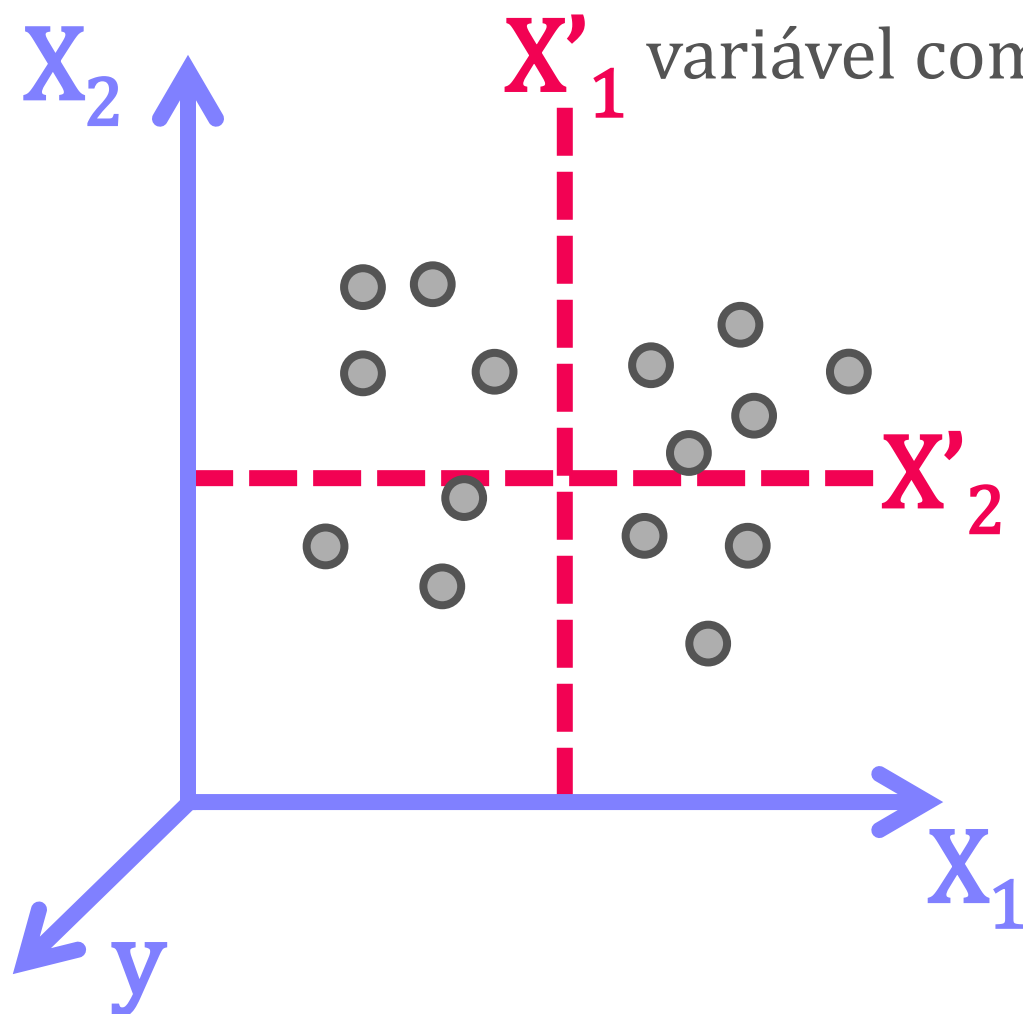
Novo ponto de divisão $X= 3.5$

$$\begin{aligned}MSE_{total} &= \frac{3}{5} 0.616 + \frac{2}{5} 0.16 \\&= 0.4336 > 0.4130\end{aligned}$$

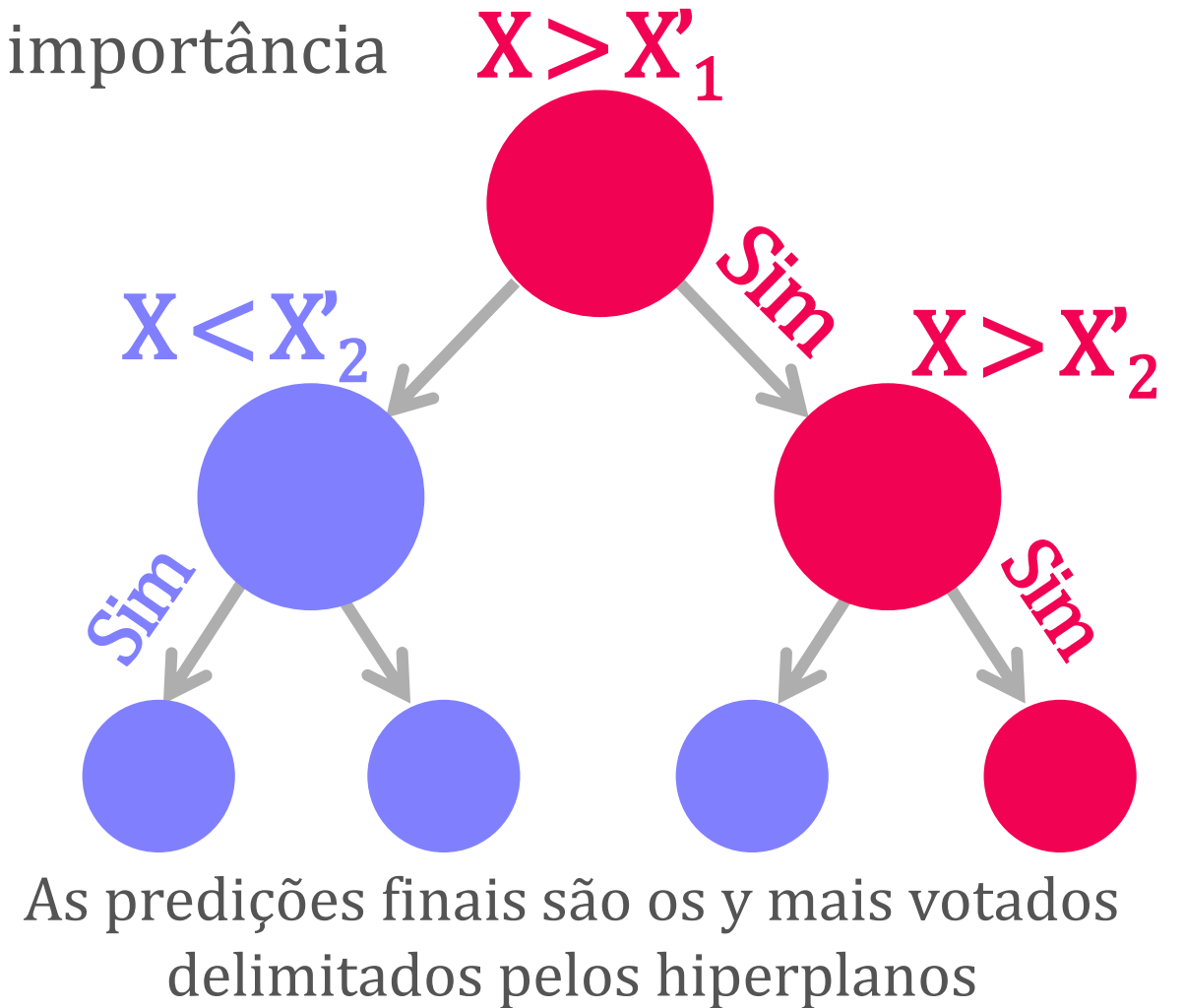
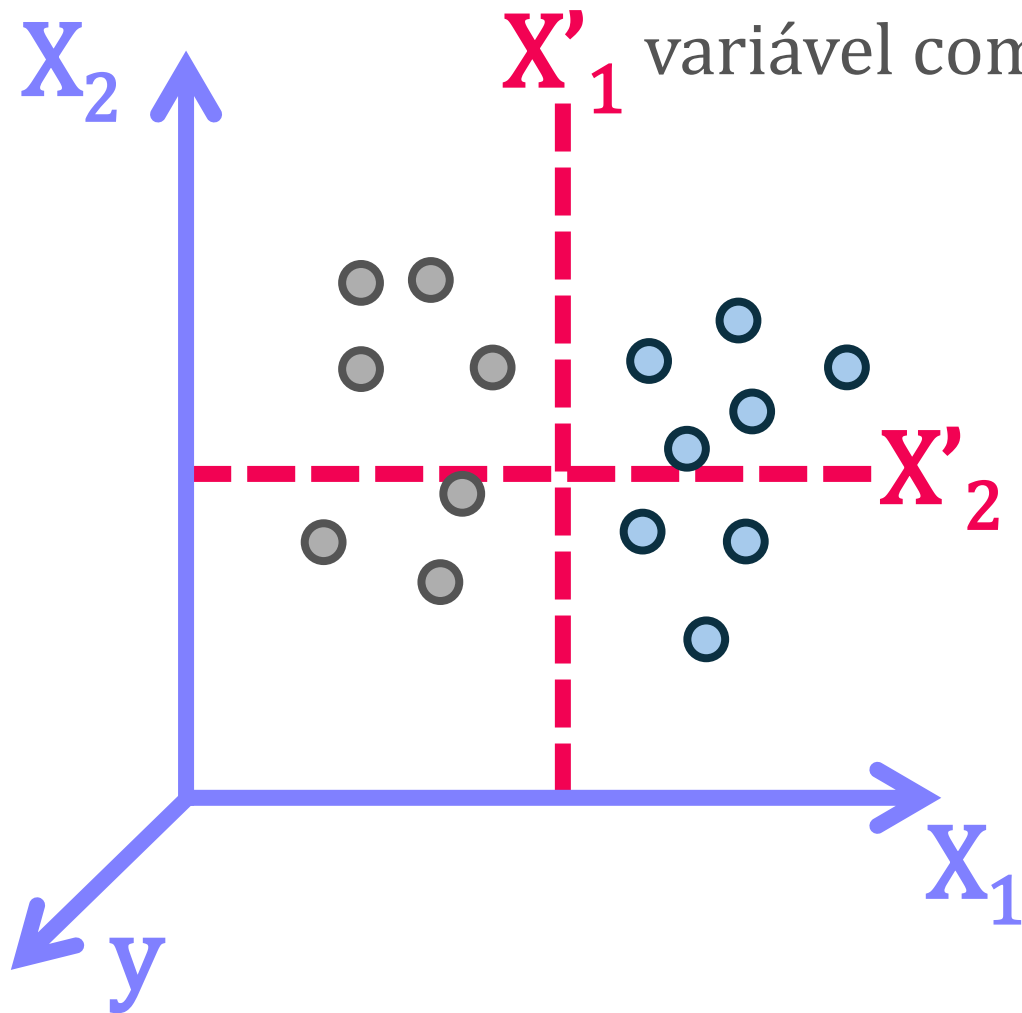
Qual o melhor ponto para dividir os dados? Ou em outras palavras, ser utilizado como o primeiro nó da árvore?

Random Forest – Regressão

Qual o melhor ponto para dividir os dados? Ou em outras palavras, ser utilizado como o primeiro nó da árvore?



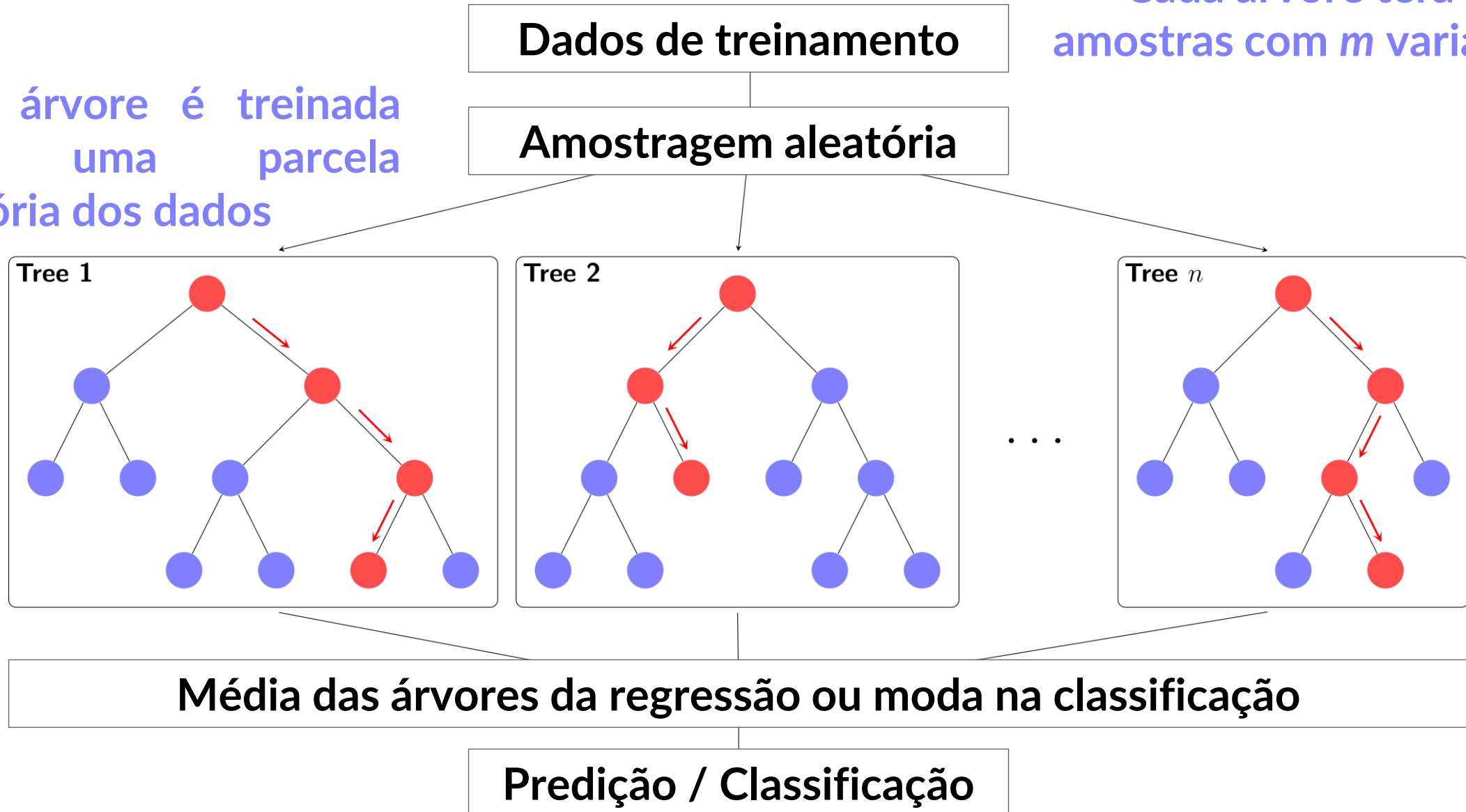
Random Forest - Classificação



Random Forest - Bagging

Cada árvore é treinada com uma parcela aleatória dos dados

Cada árvore terá n amostras com m variáveis



Random Forest

Vantagens

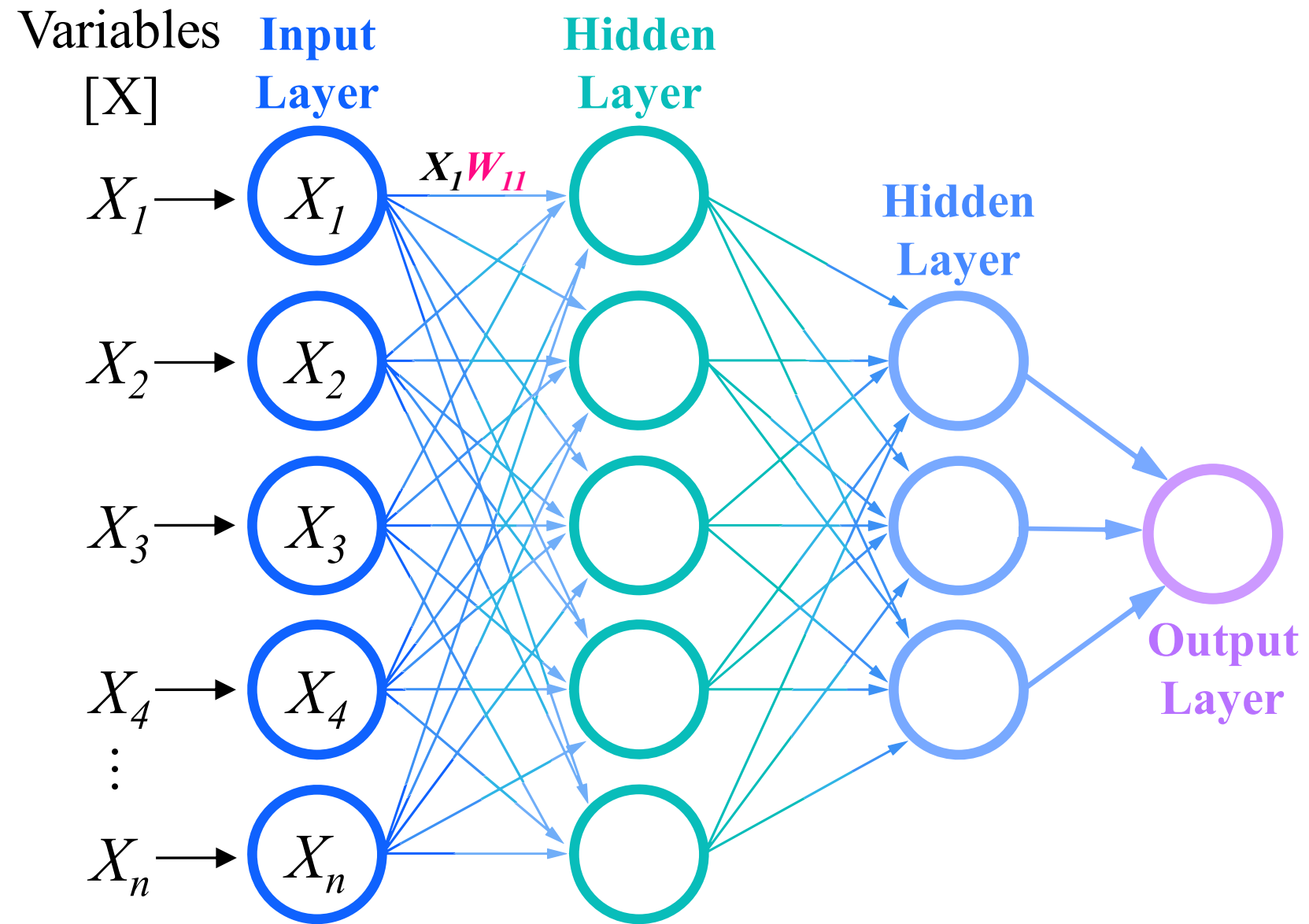
- Alta capacidade preditiva
- Poucos hiperparâmetros
- Realiza *feature importance*
- Lida bem com *datasets* grandes
- Gera métricas internas de erro
- Aleatoriedade combate o *overfitting*

Desvantagens

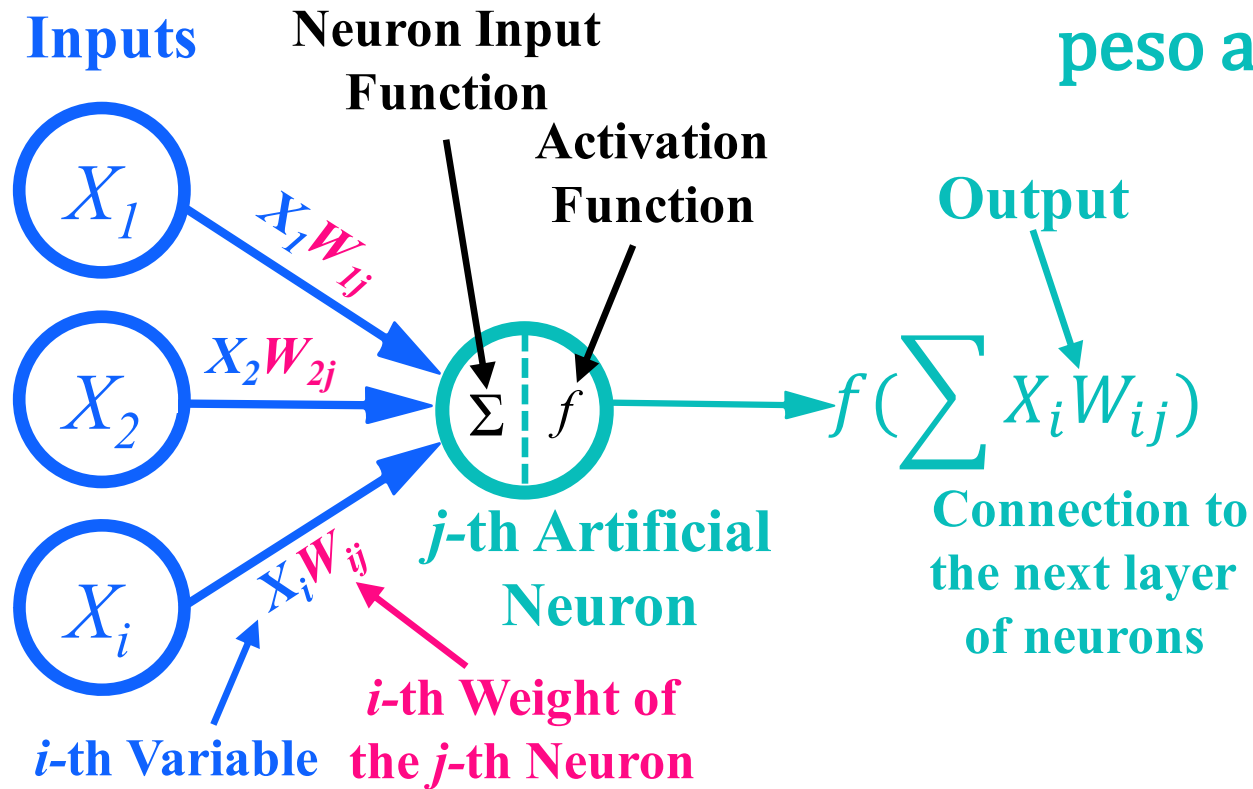
- Complexidade computacional
- Dependência dos hiperparametros
- Aleatoriedade
- Não lida bem imagem/som/texto

REDES NEURAIS CLÁSSICAS

REDE NEURAL – *MULTI LAYER PERCEPTRON*

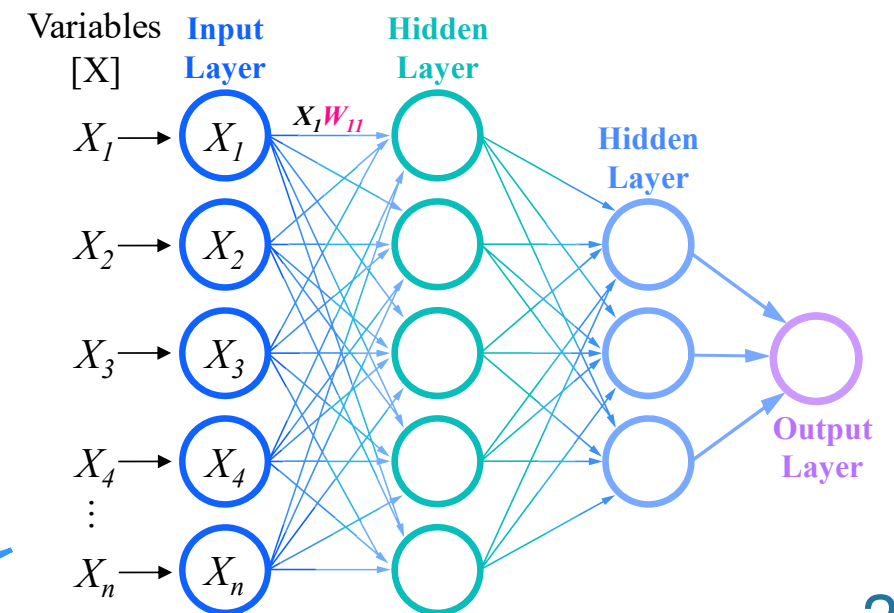


REDE NEURAL – *MULTI LAYER PERCEPTRON*



Cada neurônio associa um i - ésimo peso a i - ésima variável

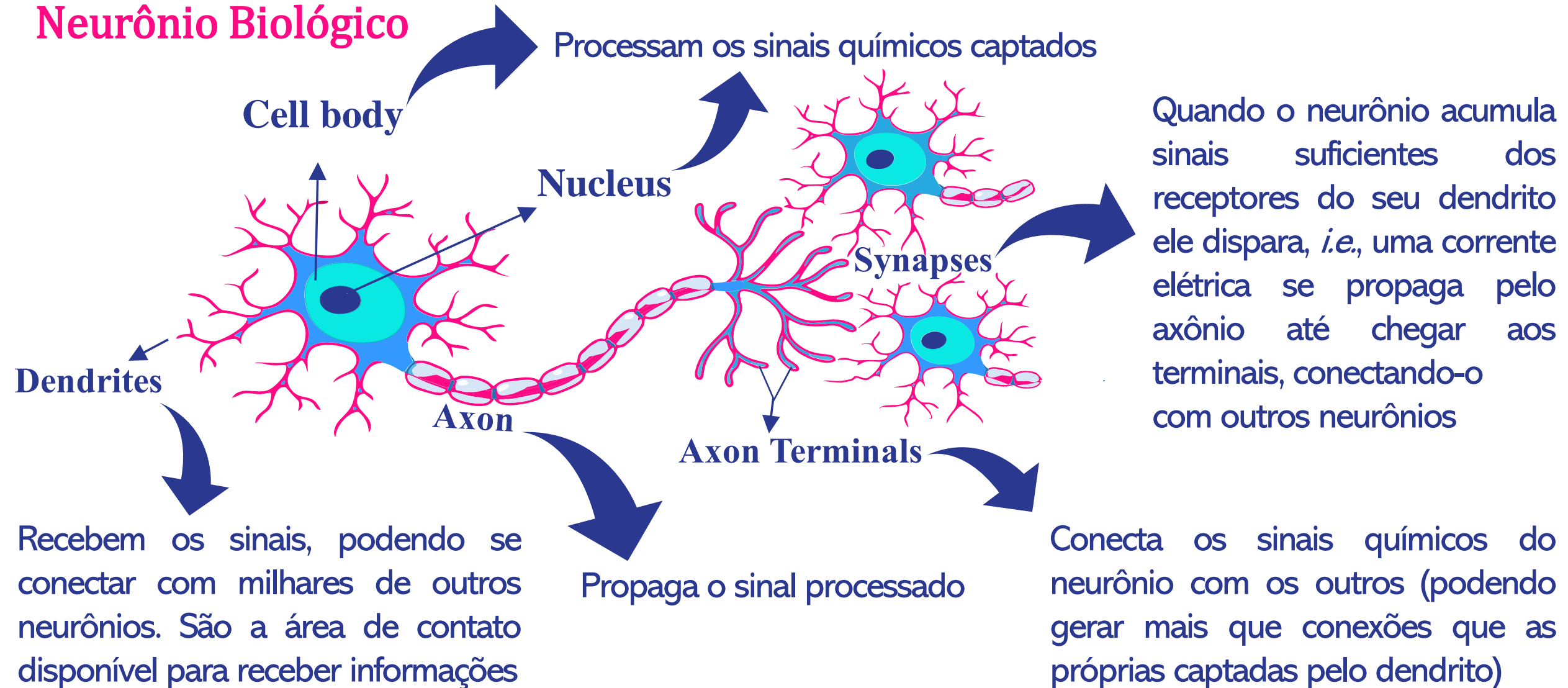
Em seguida, aplica uma função de ativação



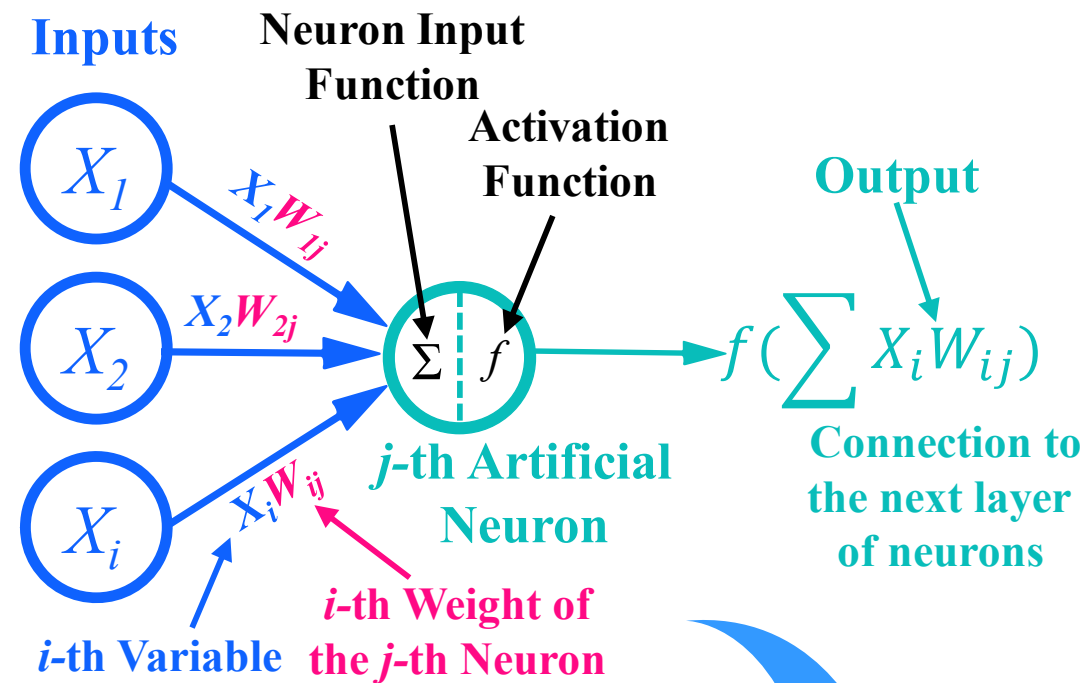
REDE NEURAL – inspiração no cérebro

Unidade básica de comunicação do sistema nervoso dos seres humanos

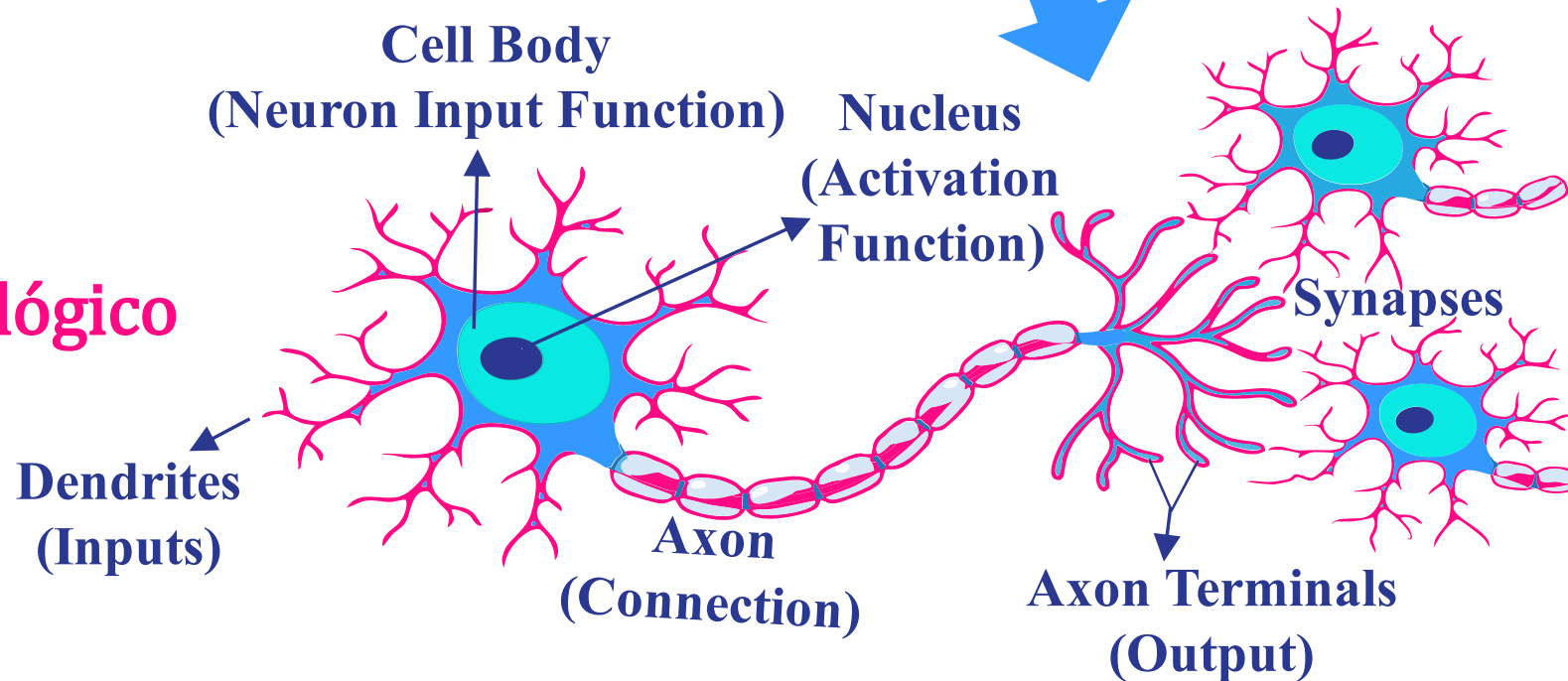
Neurônio Biológico



Neurônio Artificial (Modelo *Perceptron*)



Neurônio Biológico



Funções de ativação $f\left(\sum_{i=1}^n X_i W_{ij}\right)$

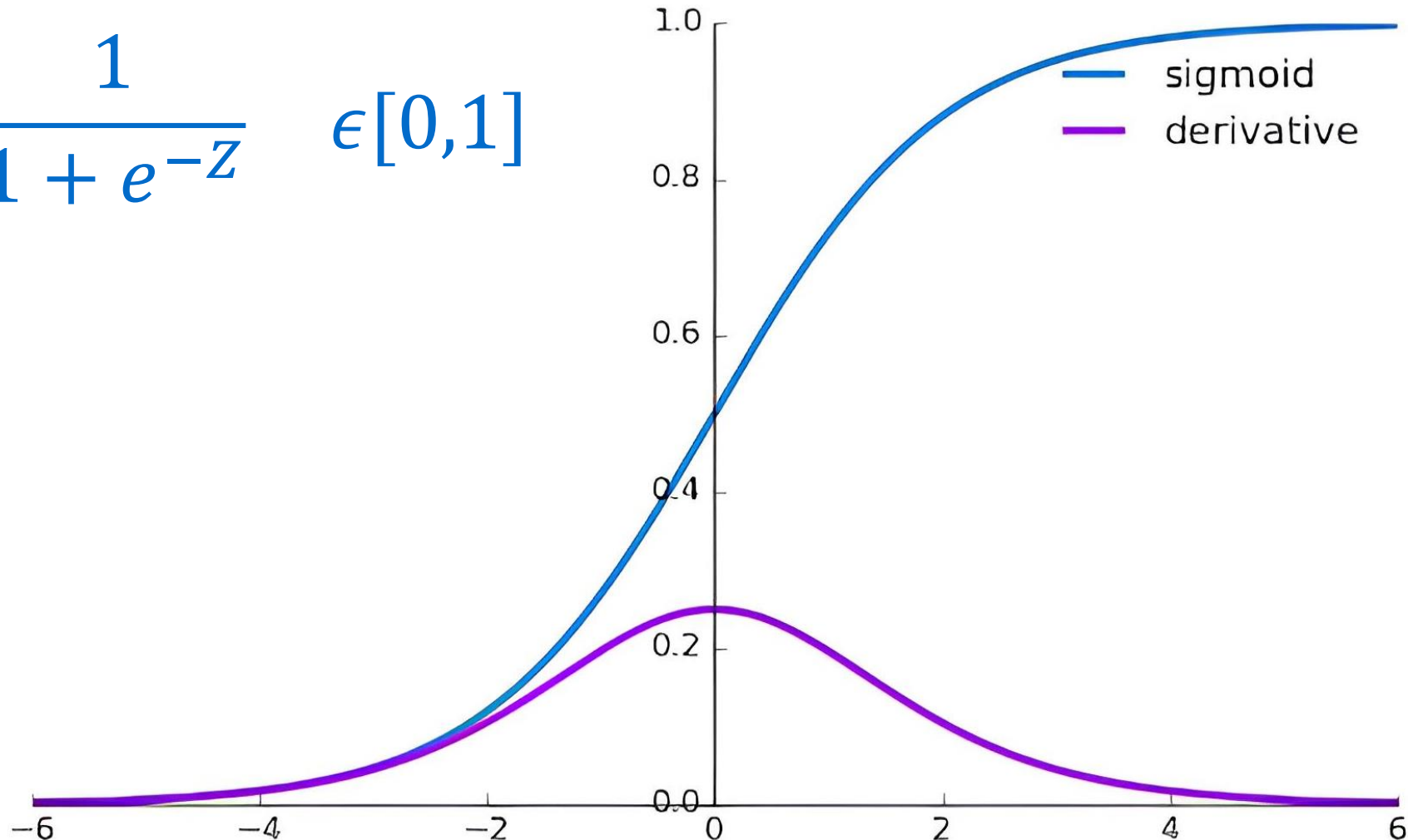
Principais objetivos

- Inserir não-linearidade na relação entre as variáveis e o target
- Facilitar a convergência (objetivo secundário)

Principais tipos: sigmóide, tanh, ReLU, Softmáx

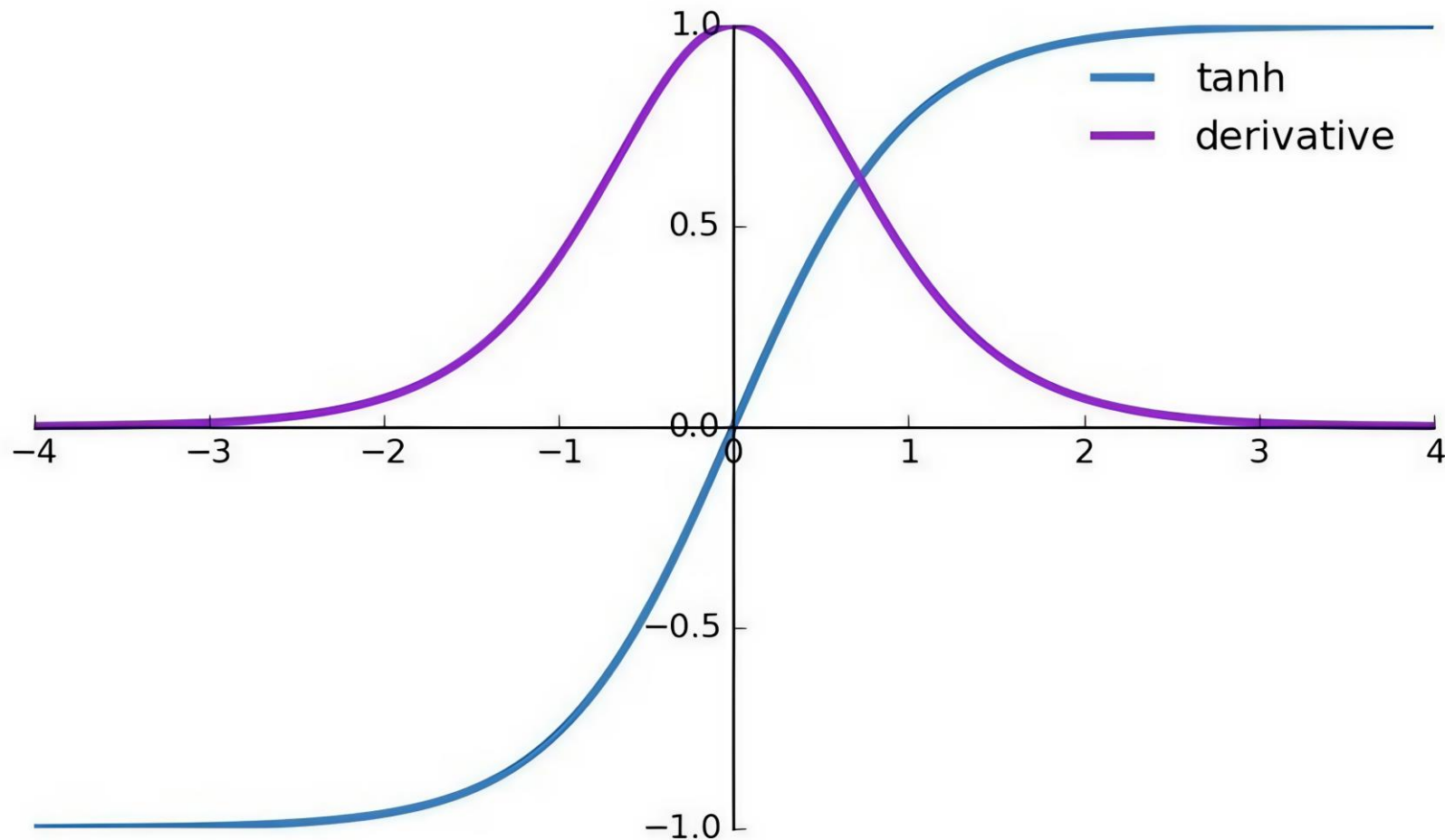
Funções de ativação: Sigmóide

$$f(z) = \frac{1}{1 + e^{-z}} \quad \epsilon[0,1]$$



REDE NEURAL – *MULTI LAYER PERCEPTRON*

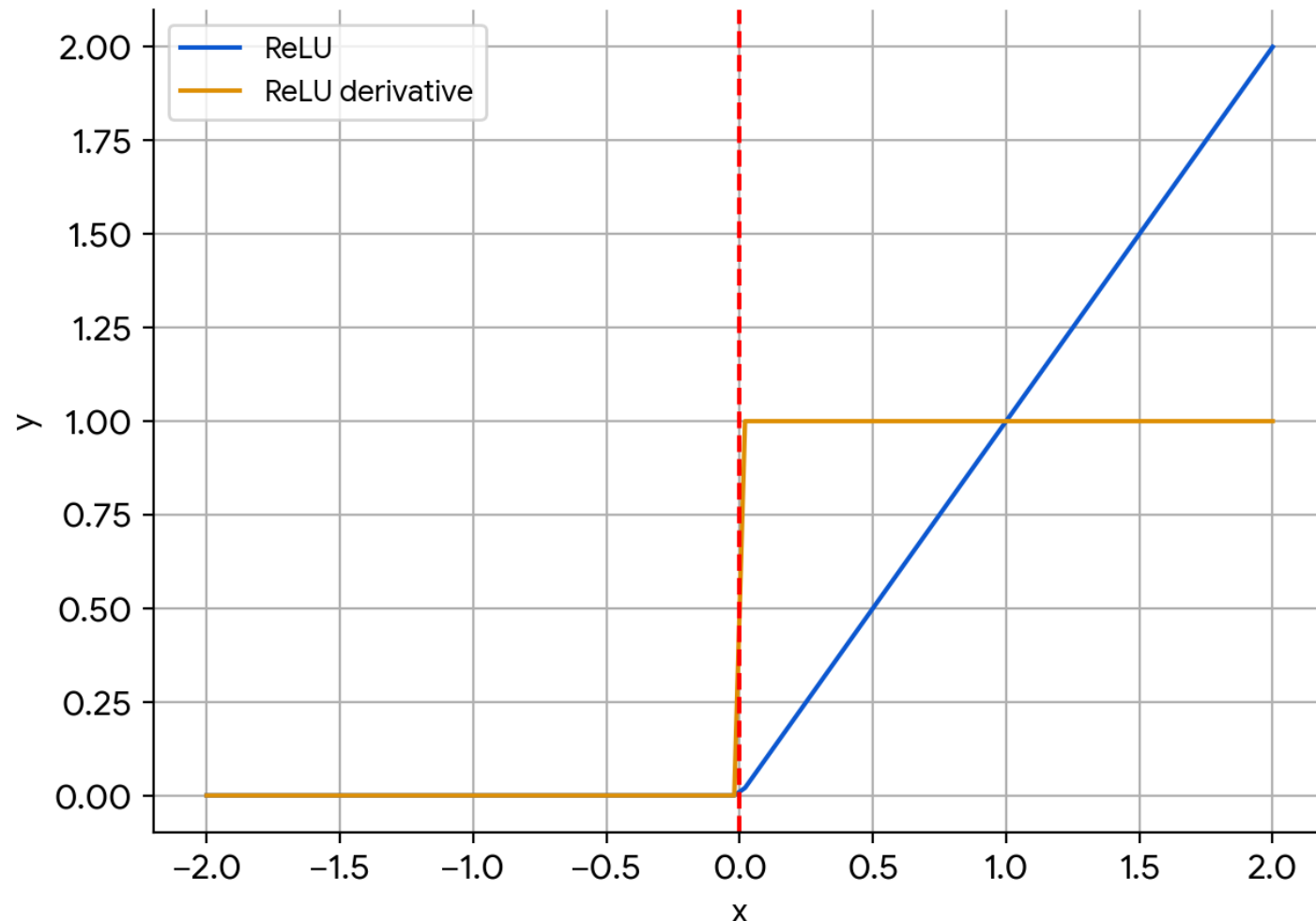
Funções de ativação: Tanh $f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \in [-1,1]$



REDE NEURAL – *MULTI LAYER PERCEPTRON*

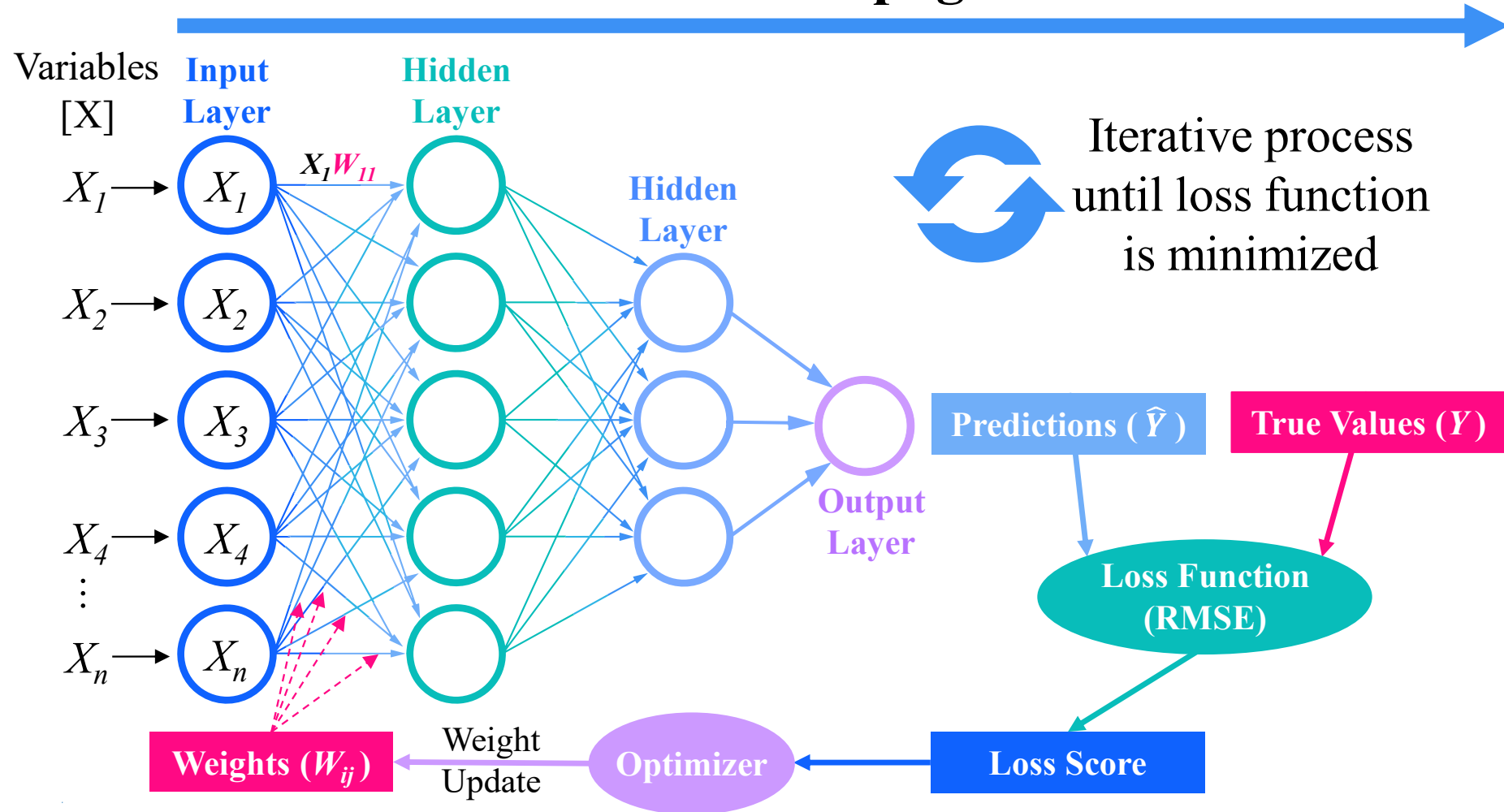
Funções de ativação: ReLU

$$\text{ReLU}(z) = 0 \text{ se } Z < 0$$
$$Z \text{ se } Z > 0$$
$$\in [0, \infty]$$



REDE NEURAL – *MULTI LAYER PERCEPTRON*

Forward Propagation



Backward Propagation

REDE NEURAL – *MULTI LAYER PERCEPTRON*

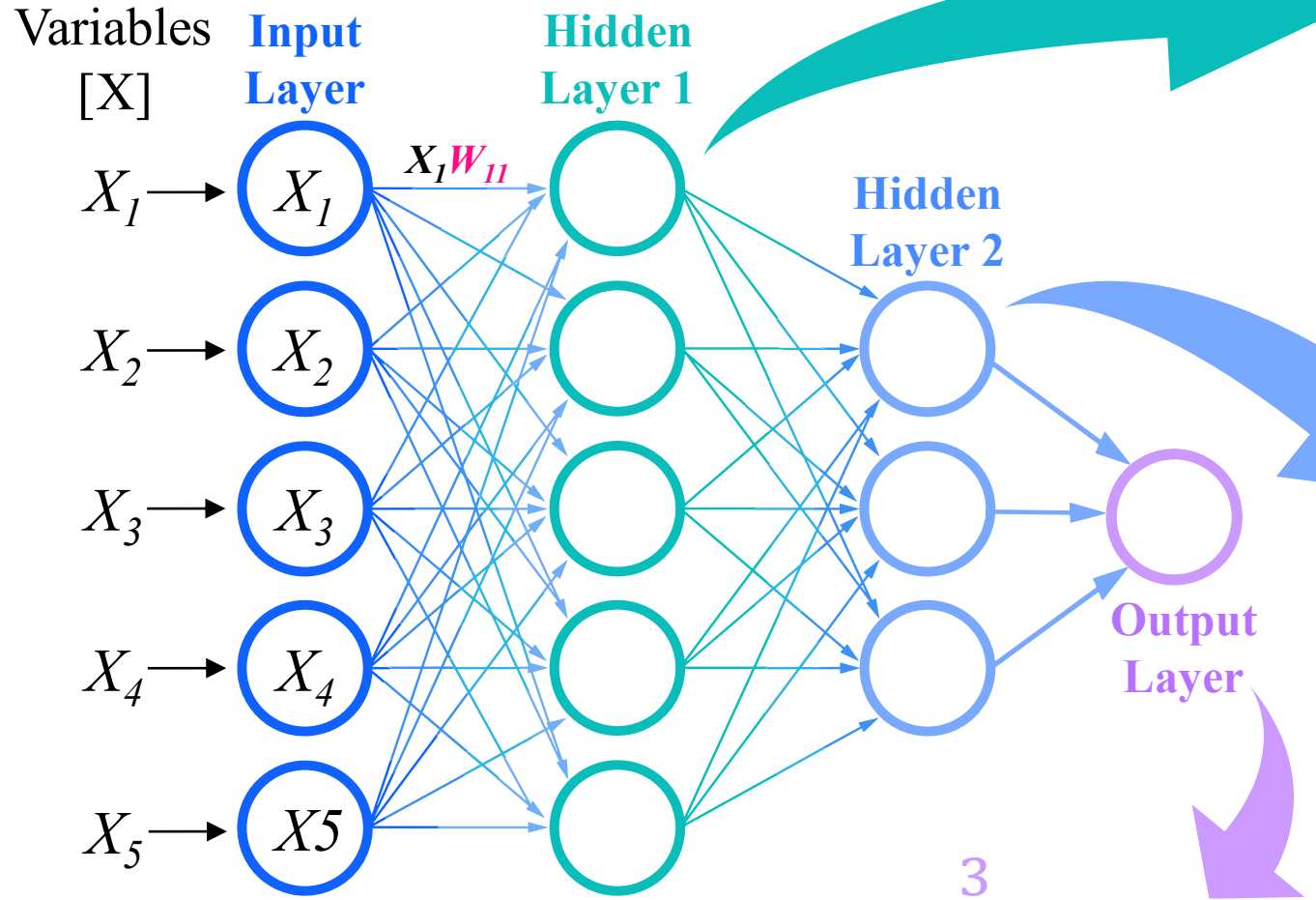
O objetivo da rede é gerar uma função com dependências em diversos parâmetros que, dado uma matriz X de variáveis de entrada, produz um vetor y com valores de saída

FASE FOWARD

A função é então aproximada (através da otimização dos hiper-parâmetros) até que atinja um nível de precisão desejado. Após esse processo ela deve apresentar capacidade de generalização

FASE BACKWARD

FASE FOWARD



Nº pesos

$$5 \times 5 + 3 \times 5 + 3 = 43$$

$$\hat{y} = \sum_{k=1}^3 w_k^{[3]} f_k^{[2]}$$

$$z_1^{[1]} = w_{11}^{[1]} x_1 + w_{12}^{[1]} x_2 + w_{13}^{[1]} x_3 + w_{14}^{[1]} x_4 + w_{15}^{[1]} x_5$$

$$f_j^{[1]} = f\left(\sum_{j,i}^5 w_{ji}^{[1]} x_i\right)$$

$$z_1^{[2]} = w_{11}^{[2]} f_1^{[1]} + w_{12}^{[2]} f_2^{[1]} + w_{13}^{[2]} f_3^{[1]} + w_{14}^{[2]} f_4^{[1]} + w_{15}^{[2]} f_5^{[1]}$$

$$f_k^{[2]} = f\left(\sum_{k,j}^{3,5} w_{kj}^{[2]} f_j^{[1]}\right)$$

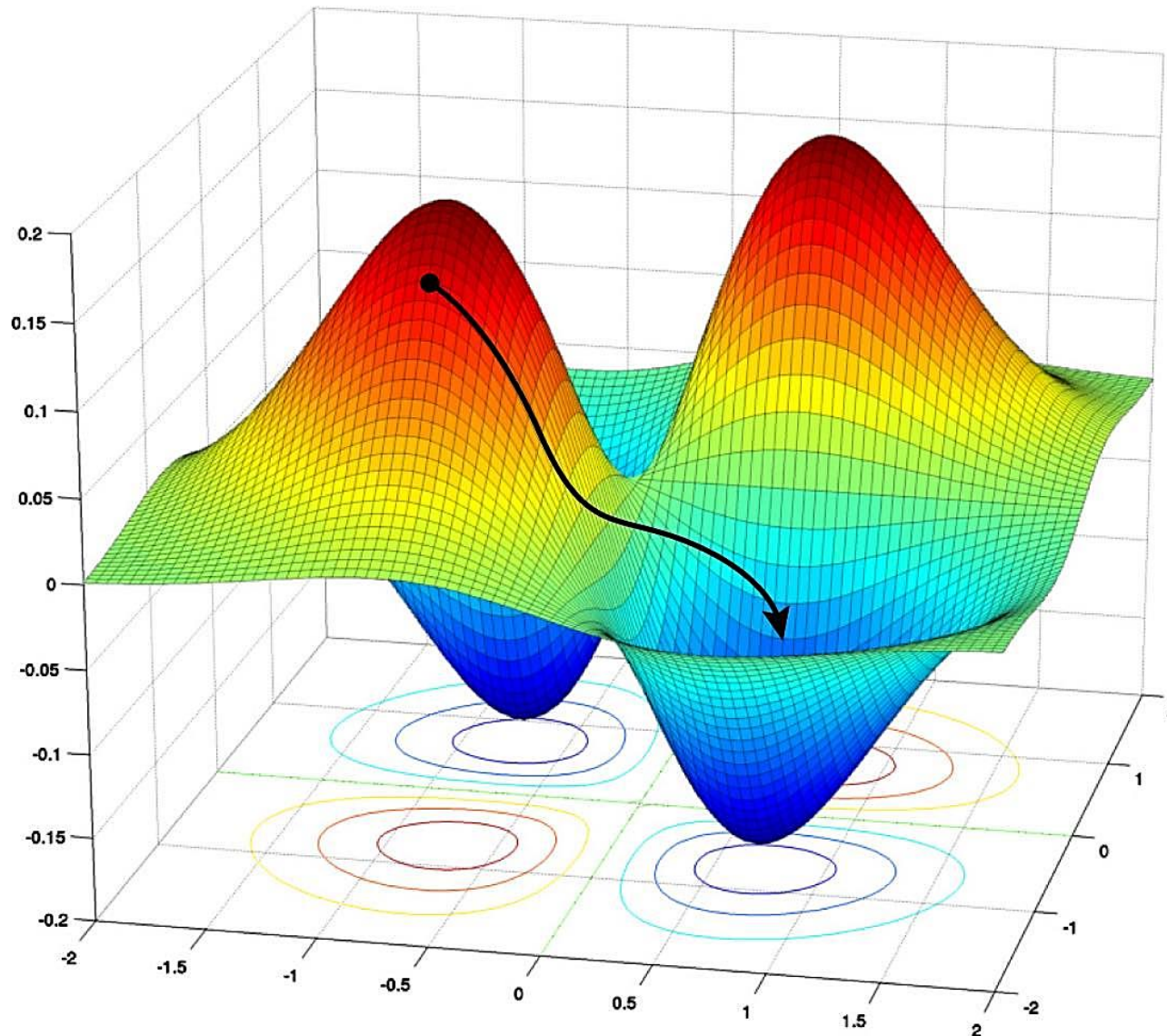
REDE NEURAL – FASE *BACKWARD*

$Loss = y - \hat{y} = L(\theta)$ Queremos o conjunto de parâmetros θ^* que minimiza $L(\theta)$

Dado um conjunto de valores θ^t (dado pelas condições iniciais), a uma nova configuração (θ^t) na qual $L(\theta)$ mais irá crescer é dada pelo gradiente aplicado no ponto, *i.e.*, $\nabla L(\theta^t)$

Logo, em uma época futura ($t+1$): $\theta^{t+1} = \theta^t - \nabla L(\theta^t)$ nos fornecerá uma nova configuração de parâmetros (*i.e.*, novo ponto) que minimizará L

REDE NEURAL – GRADIENTE DESCENDENTE



$$\theta^{t+1} = \theta^t - \eta_t \nabla L(\theta^t)$$

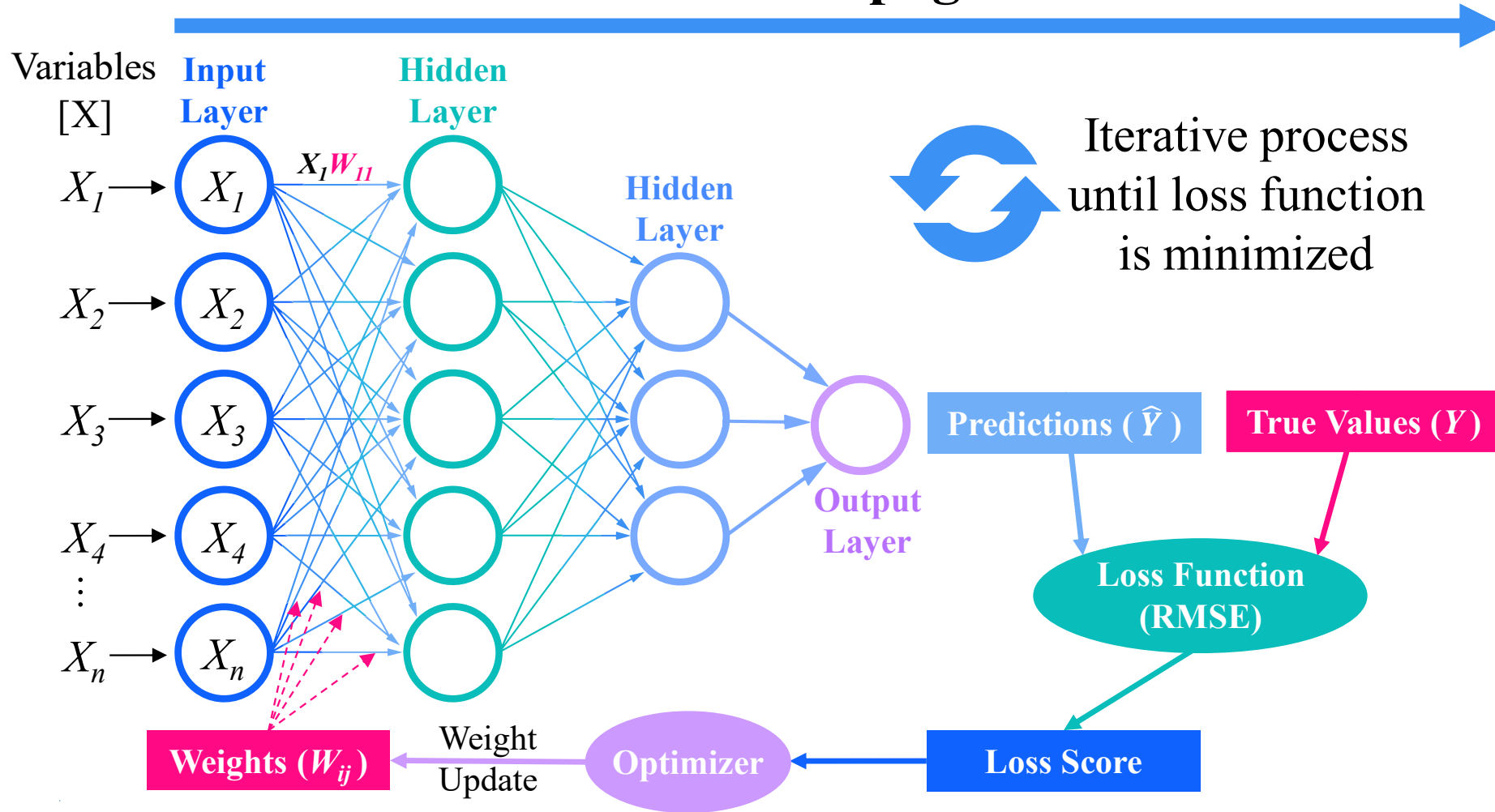
η_t : taxa de aprendizagem

SGD (*Stochastic Gradient Descent* - 1950)
Na prática o cálculo é feito computacionalmente, por meio de algoritmos otimizados para o cálculo das derivadas
L-BFGS (*Limited-Memory Broyden-Fletcher-Goldfarb-Shannon* - 1989)

ADAM (*Adaptive Moment Estimation* - 2014)

REDE NEURAL – BACKPROPAGATION

Forward Propagation



Vantagens

- Capacidade de modelar relações complexas
- Alta capacidade preditiva
- Lida bem com *datasets* grandes
- Adaptabilidade/flexibilidade para os mais diversos problemas

Desvantagens

- Complexidade computacional
- Necessidade de muitos dados
- Propensão a *overfitting*
- Dificuldade de interpretar
- Complexidade no ajuste dos hiperparâmetros

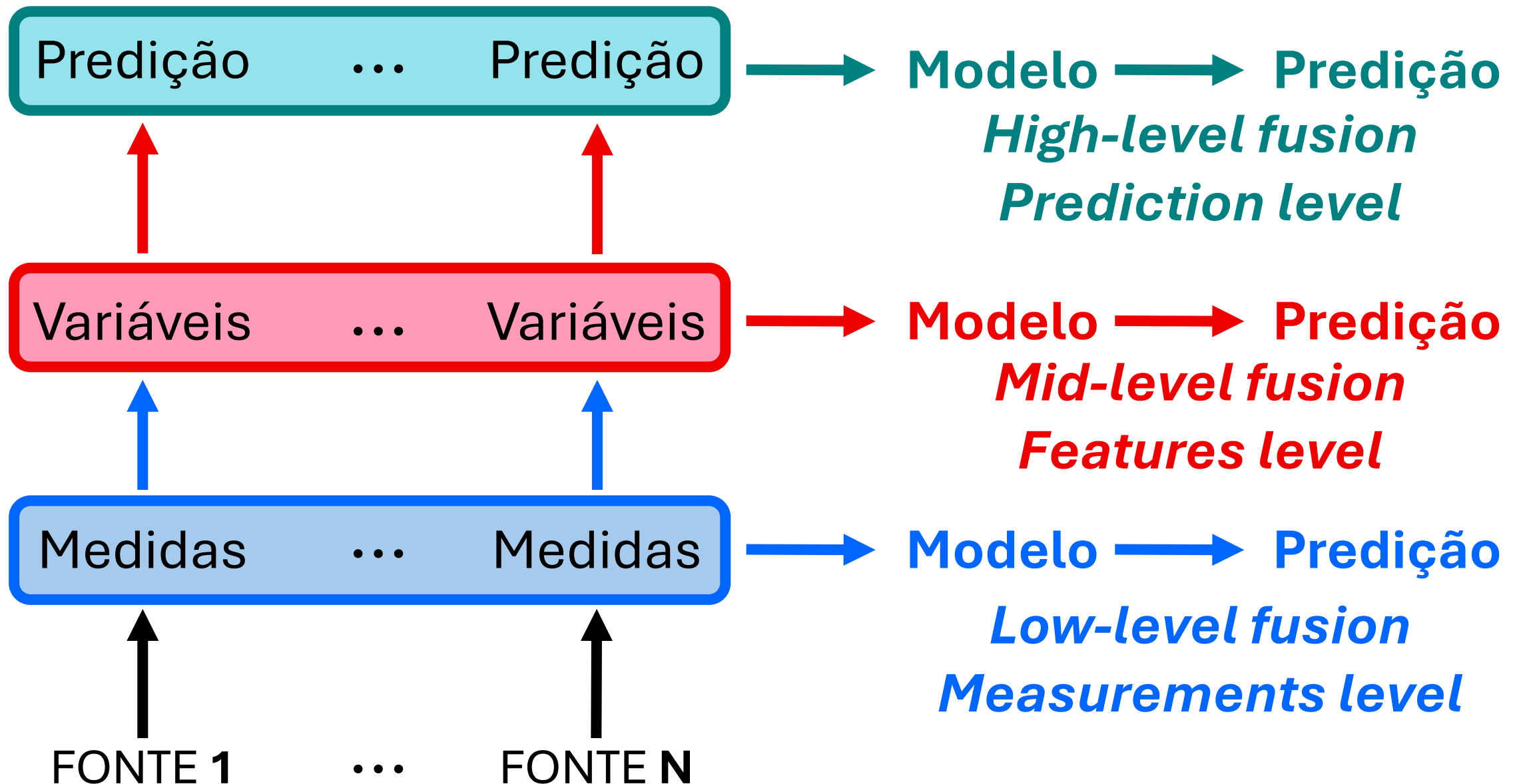
FUSÃO DE DADOS

FUSÃO DE DADOS

Conceito: combinar dados ou modelos com diferentes características afim de explorar sua sinergia e obter inferências mais precisas



NÍVEIS POSSÍVEIS



LOW-LEVEL FUSION

$$\text{CONCAT} \left[\begin{pmatrix} X_{11}^{[1]} & \dots & X_{1j}^{[1]} \\ \vdots & \ddots & \vdots \\ X_{i1}^{[1]} & \dots & X_{ij}^{[1]} \end{pmatrix}, \dots, \begin{pmatrix} X_{11}^{[N]} & \dots & X_{1j}^{[N]} \\ \vdots & \ddots & \vdots \\ X_{i1}^{[N]} & \dots & X_{ij}^{[N]} \end{pmatrix} \right]$$
$$= \begin{pmatrix} X_{11}^{[1]} & \dots & X_{1j}^{[1]} & \dots & X_{11}^{[N]} & \dots & X_{1j}^{[N]} \\ \vdots & & \vdots & & \vdots & & \vdots \\ X_{i1}^{[1]} & \dots & X_{ij}^{[1]} & \dots & X_{i1}^{[N]} & \dots & X_{ij}^{[N]} \end{pmatrix}$$

- Os dados precisam ser pré-processados de maneira adequada*
- Normalizações são recomendadas

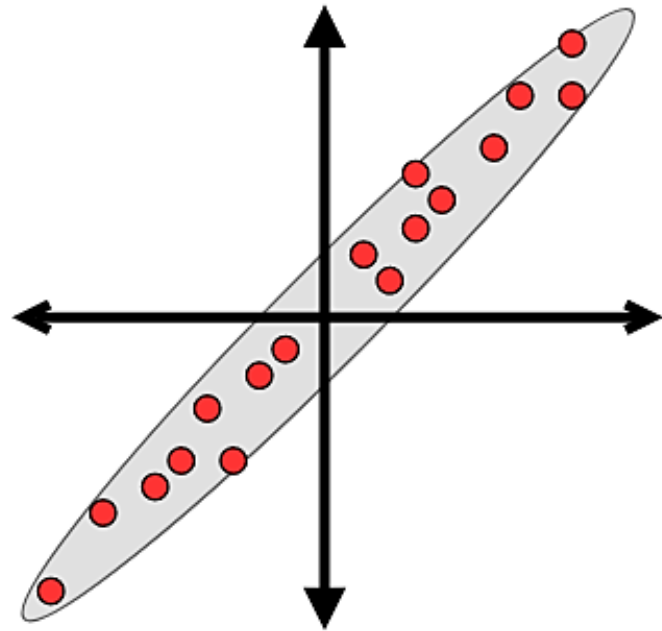
MID-LEVEL FUSION

$$f\left(\begin{array}{ccc} X_{11}^{[1]} & \dots & X_{1j}^{[1]} & \dots & X_{11}^{[N]} & \dots & X_{1j}^{[N]} \\ \vdots & & \vdots & & \vdots & & \vdots \\ X_{i1}^{[1]} & \dots & X_{ij}^{[1]} & \dots & X_{i1}^{[N]} & \dots & X_{ij}^{[N]} \end{array}\right)$$

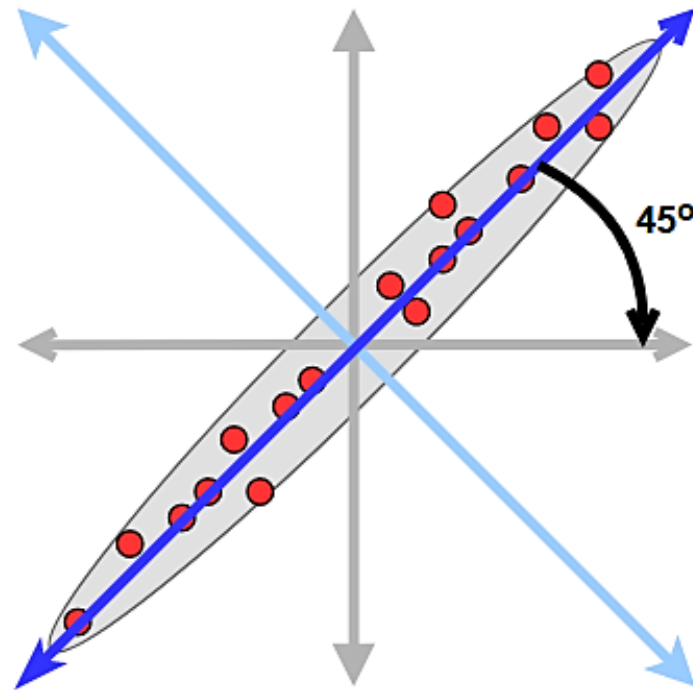
- Métodos para extração de *features complexas* são utilizados. Conhecido como *Features-level*
- Redução de dimensionalidade
- Seleção de variáveis

MID-LEVEL FUSION

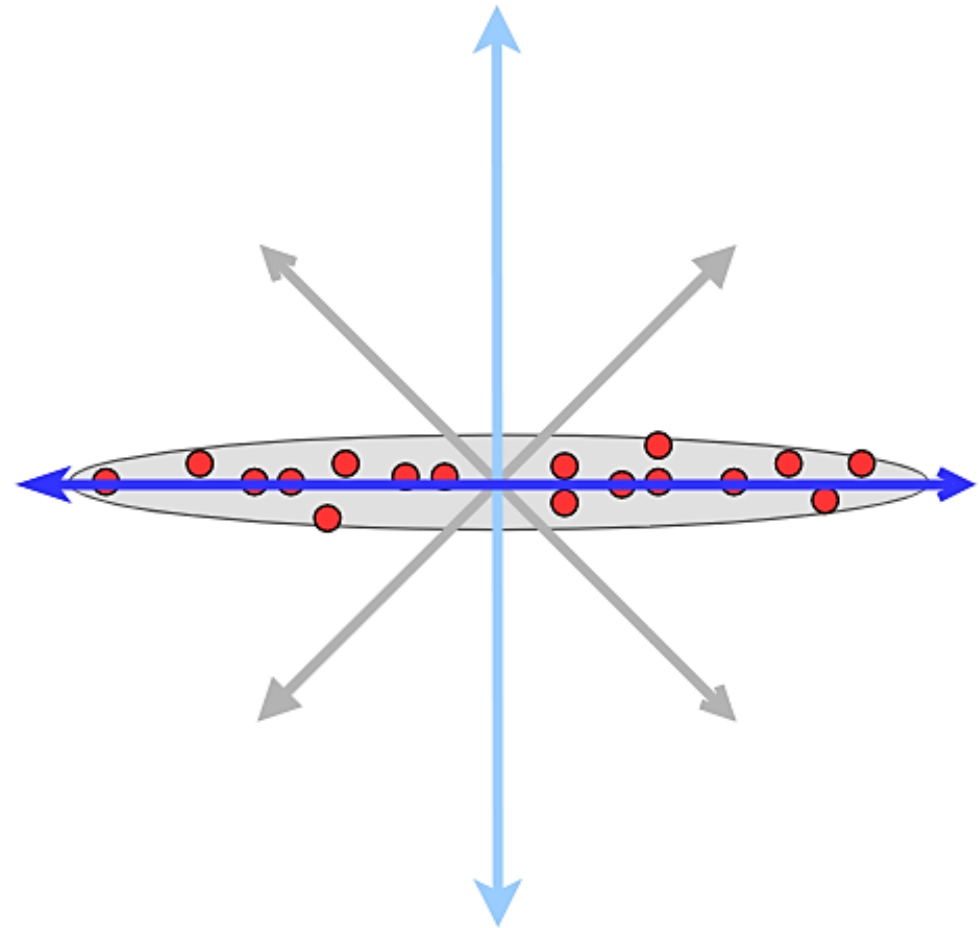
Exemplo: PCA



Original Coordinates



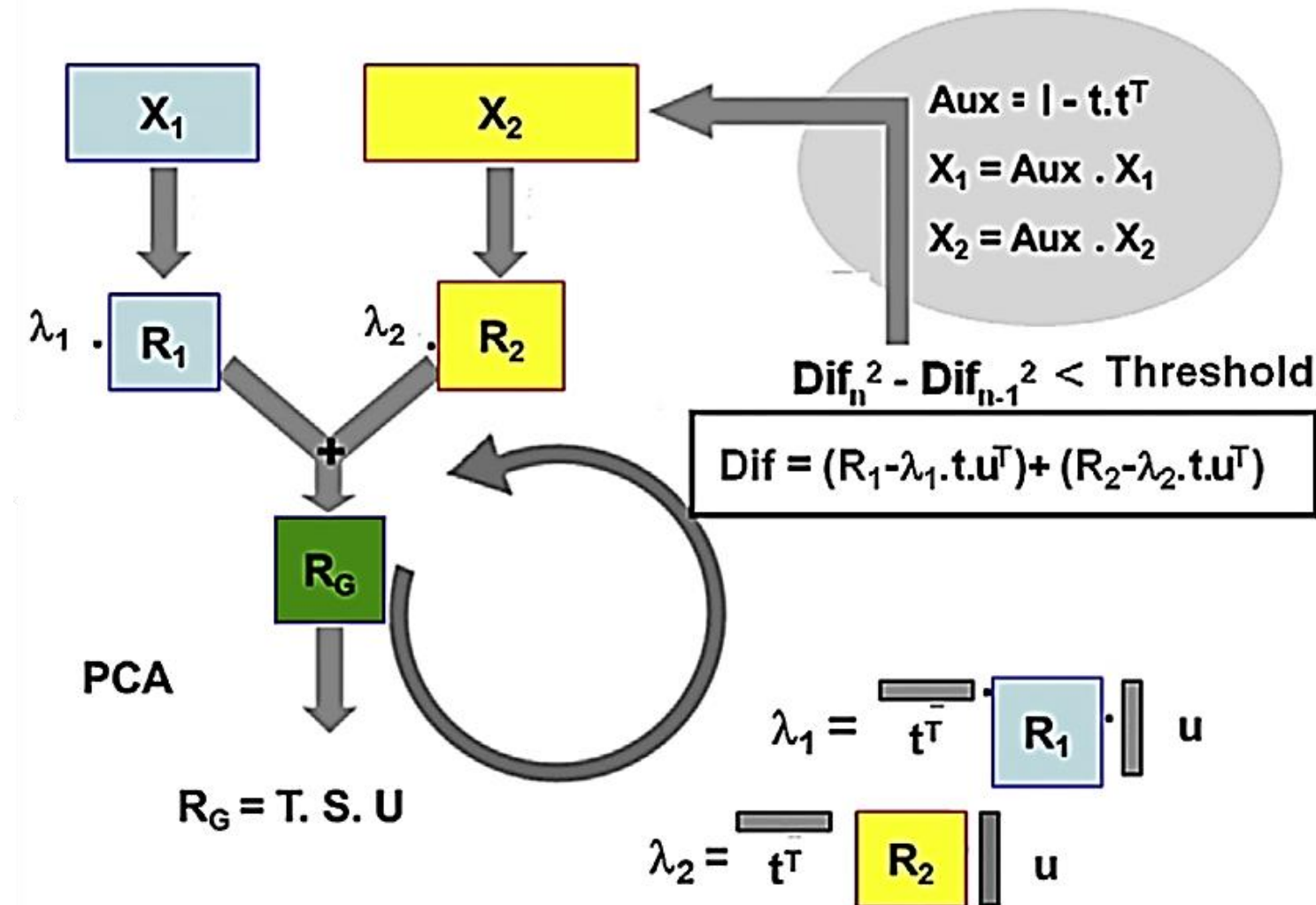
Principle Component



New Rotated Coordinates

MID-LEVEL FUSION

Exemplo: Análise de dimensões comuns (ComDim)



$$R_i = X_i X_i^t y y^t$$

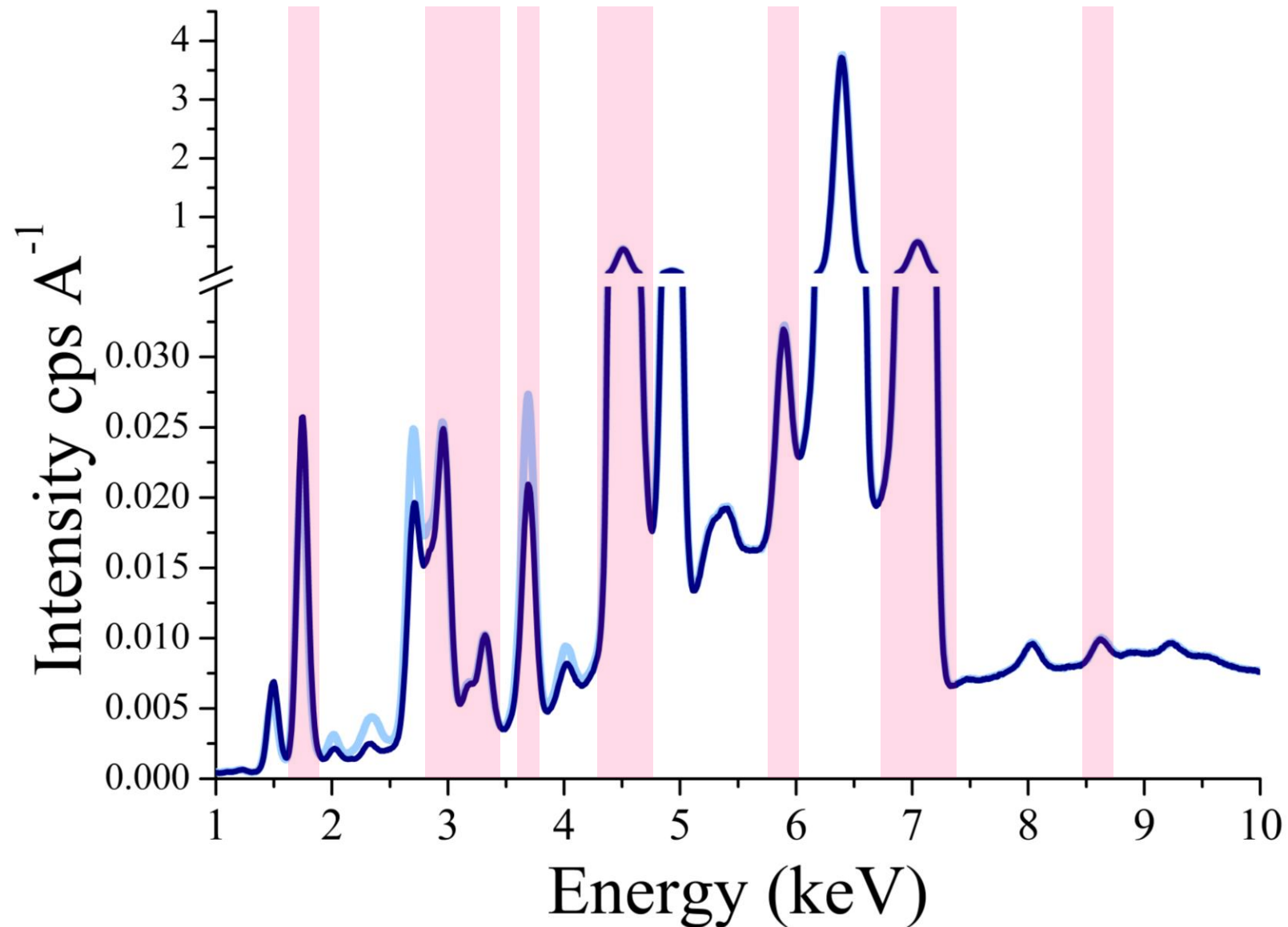
$i = 1, 2, \dots n^o \text{ de blocos}$

$$Y = bT^{(DC)}$$

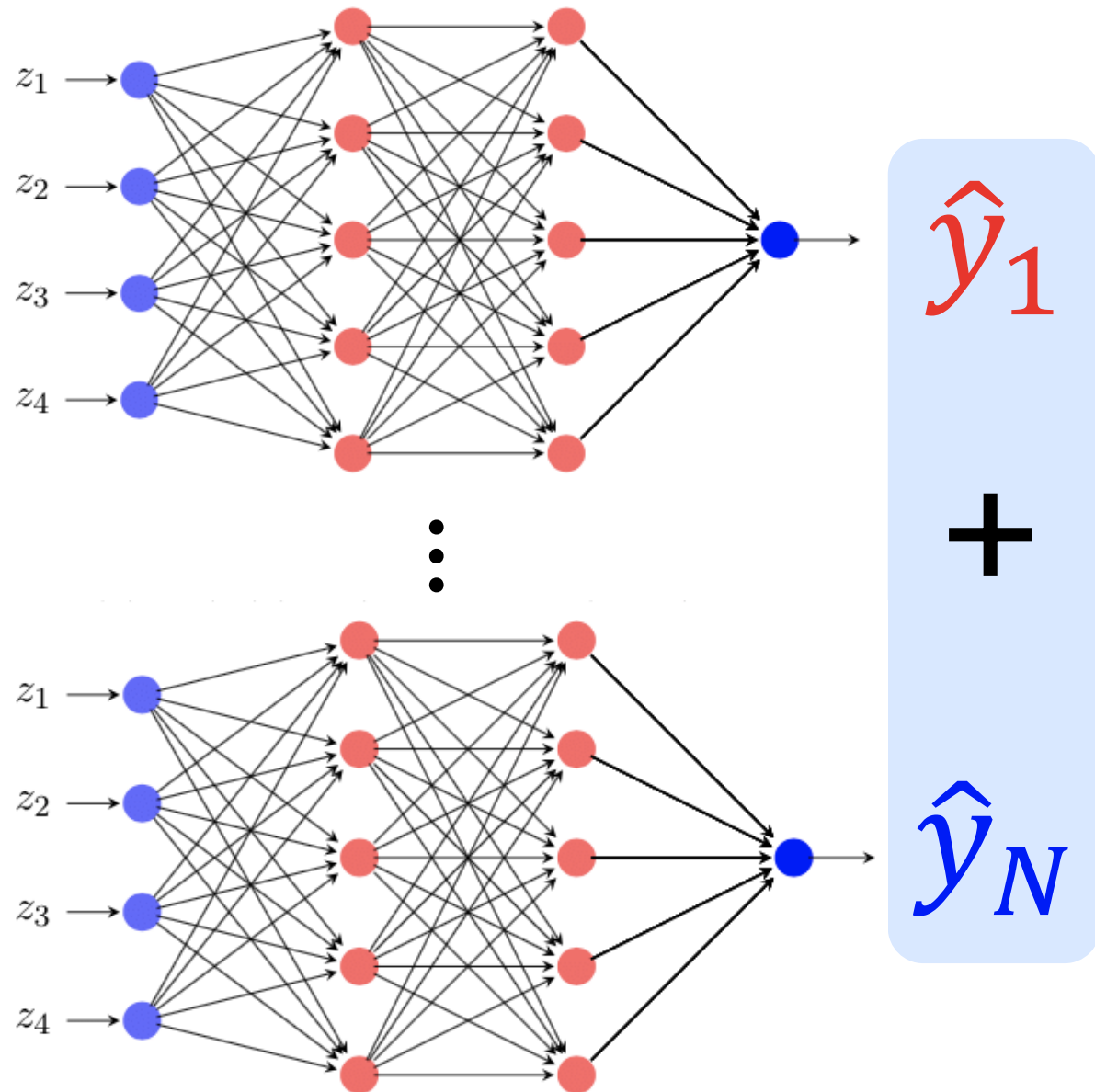
$$b = (T^t T)^{-1} X^t Y$$

MID-LEVEL FUSION

Ex: Seleção de variáveis



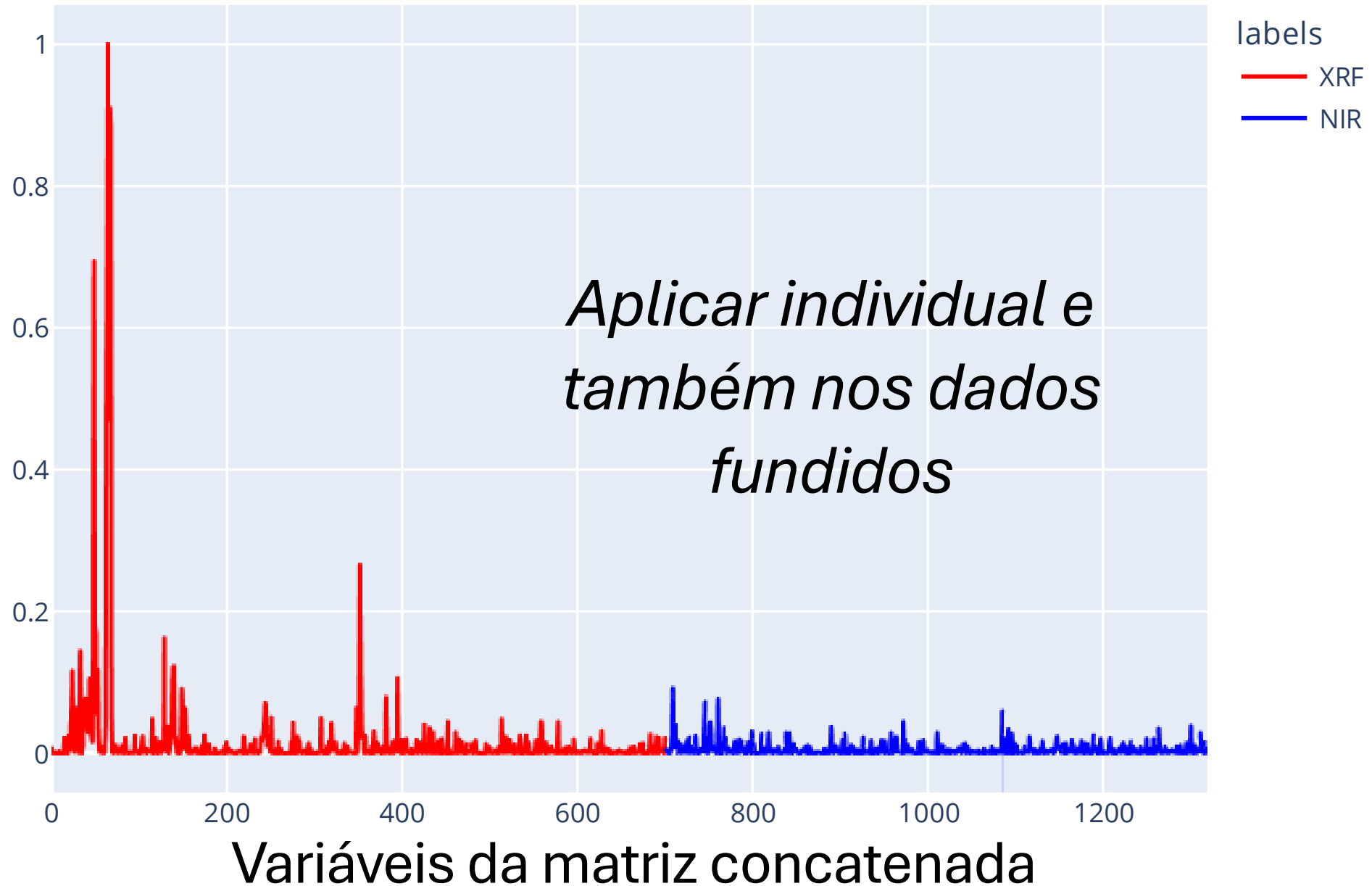
HIGH-LEVEL FUSION



$$\hat{y}_{HL} = b_0 + b_1 \hat{y}_1 + \dots + b_N \hat{y}_N$$

Também chamado
de ***Prediction-Level***

- Pré-processamentos continuam sendo importantes!



FUSÃO DE DADOS

Vantagens

- Predição utilizando características complementares
- Alta capacidade preditiva
- Detecção aprimorada de padrões complexos:
- Redução de incertezas
- Pode ser melhor do que modelos individuais

Desvantagens

- Complexidade computacional aumentada
- Problemas de qualidade e compatibilidade de dados (outliers)
- Necessário dados de mais de uma técnica
- *Sincronização e alinhamento de dados*
- Muitas variáveis (*overfitting*)

PRÁTICA – GOOGLE COLAB