MÉTODOS NÃO-LINEARES E FUSÃO DE DADOS

Me. José Vinícius Ribeiro

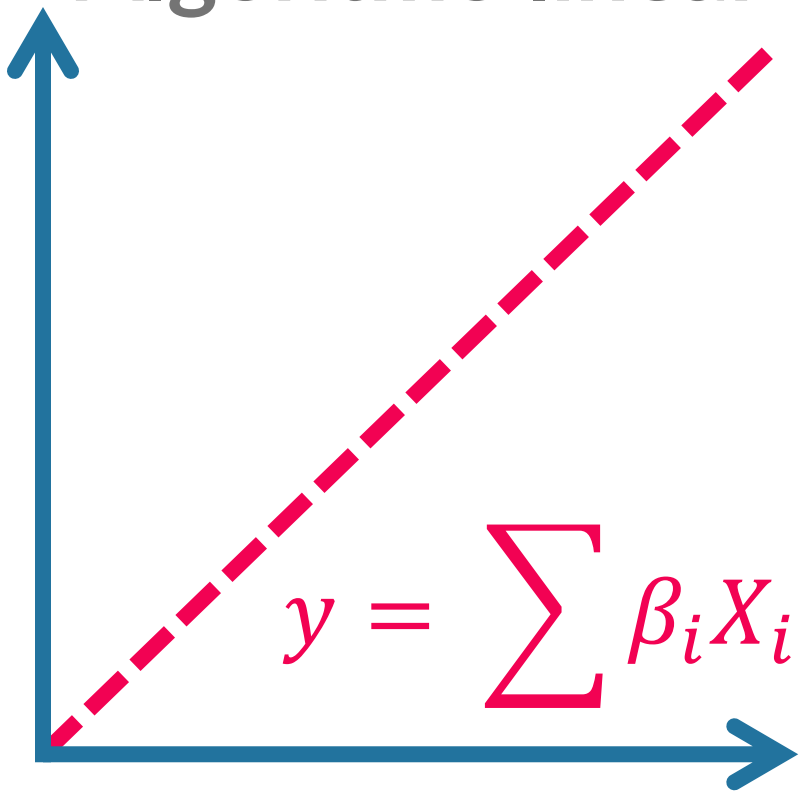
2FIS446

SUMÁRIO

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

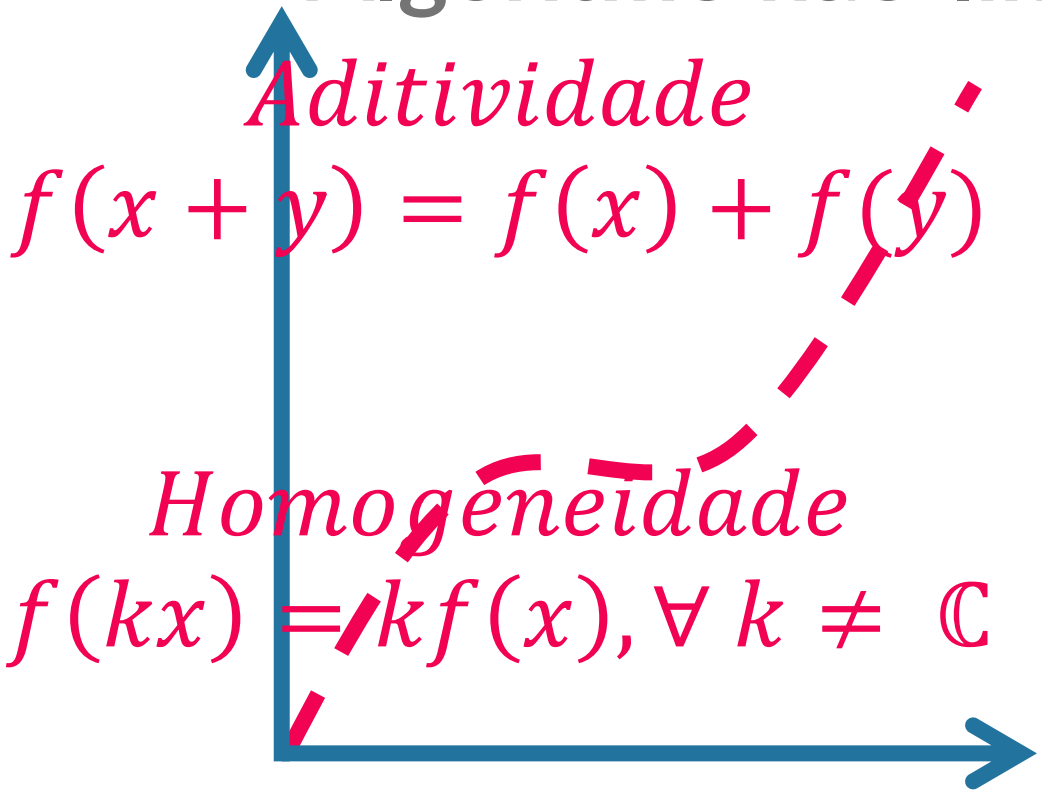
LINEARIDADE X NÃO-LINEARIDADE

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

Não-Lineares

- Arvores 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

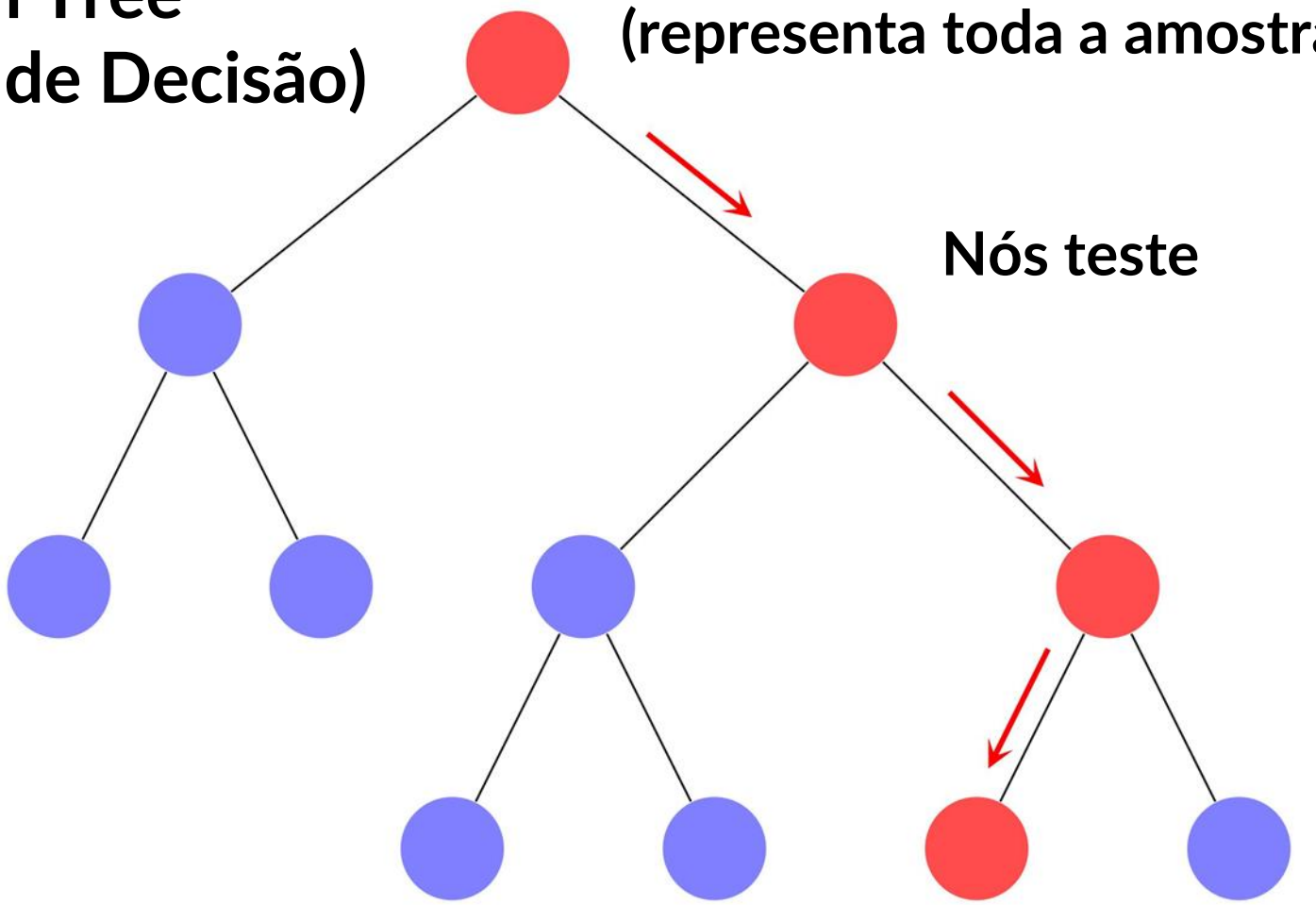
RANDOM FOREST

Random Forest (Floresta Aleatória)

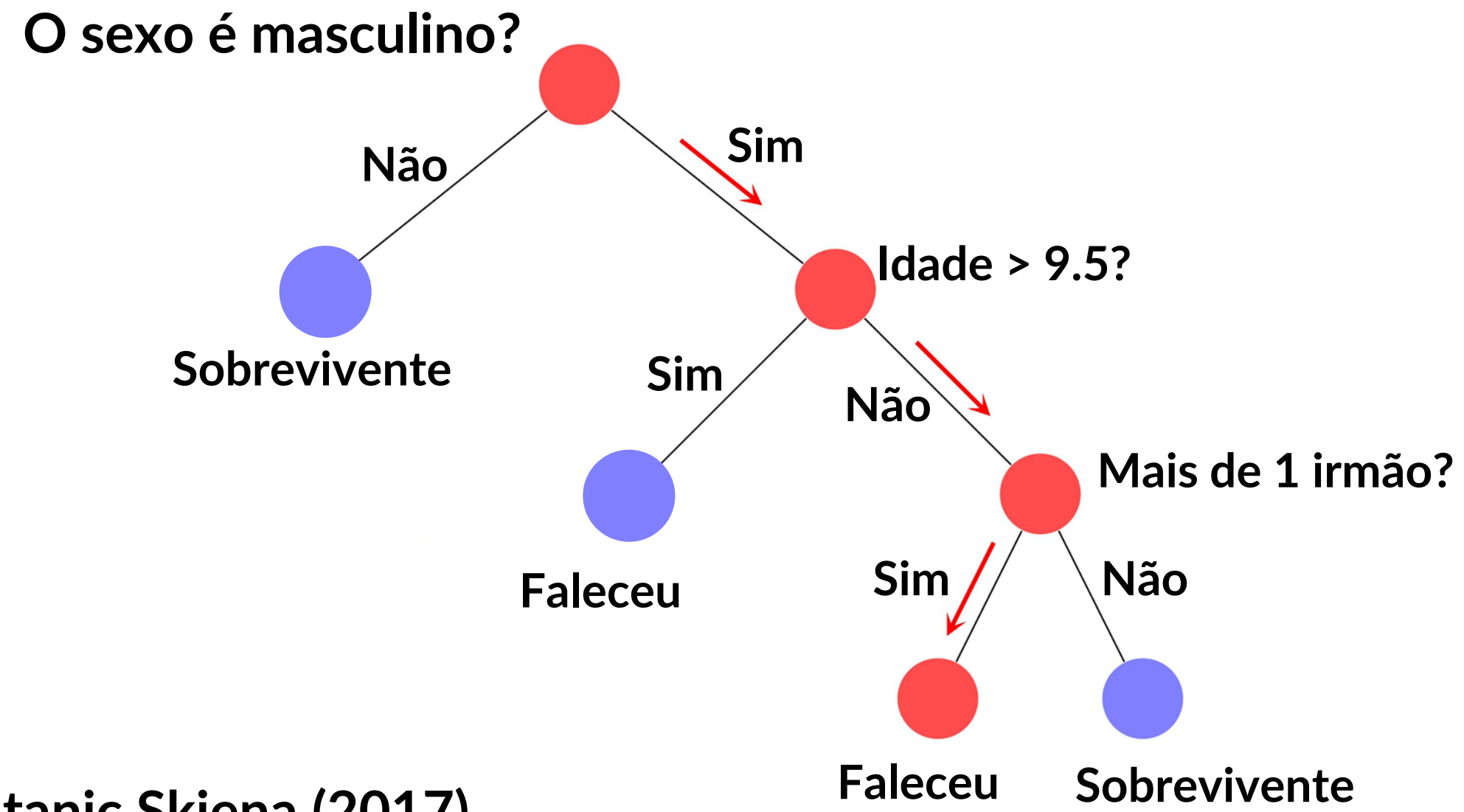
O principal método baseado em *ensembles*

Baseado em arvores de decisão

Decision Tree
(Árvore de Decisão)



Random Forest (Floresta Aleatória)

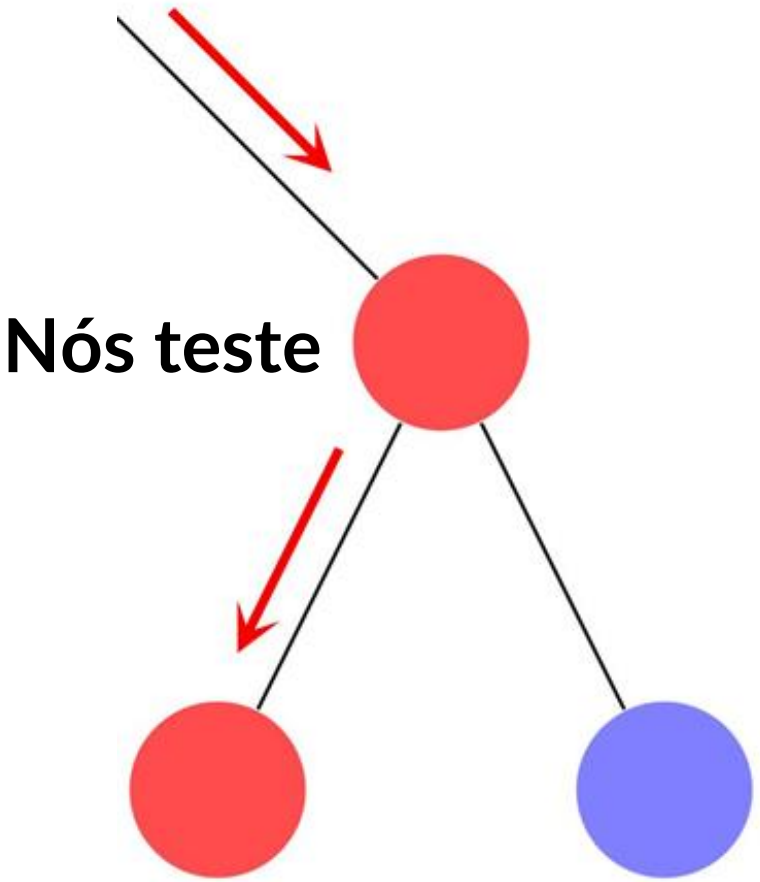


Exemplo Titanic Skiena (2017)

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

Random Forest (Floresta Aleatória)

Floresta Aleatória *(Random Forest)* – *Classificação e Regressão*



Portanto, existem *variaveis* 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

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

n= amostras

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

m = variáveis

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

Uma possivel *decision tree* seria
X=[1, 2, 3, 4, 5] 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 → $MSE = \sum_{i=1}^{p} \frac{(y_i - \bar{y}_i)^2}{p}$

- X < 2.5

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

$\bar{y}_i$=1.55

$= 0.1225$

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

$$MSE_{total} = \frac{2}{5}\,0.1225 + \frac{3}{5}\,0.6067$$

$$= 0.41302$$

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

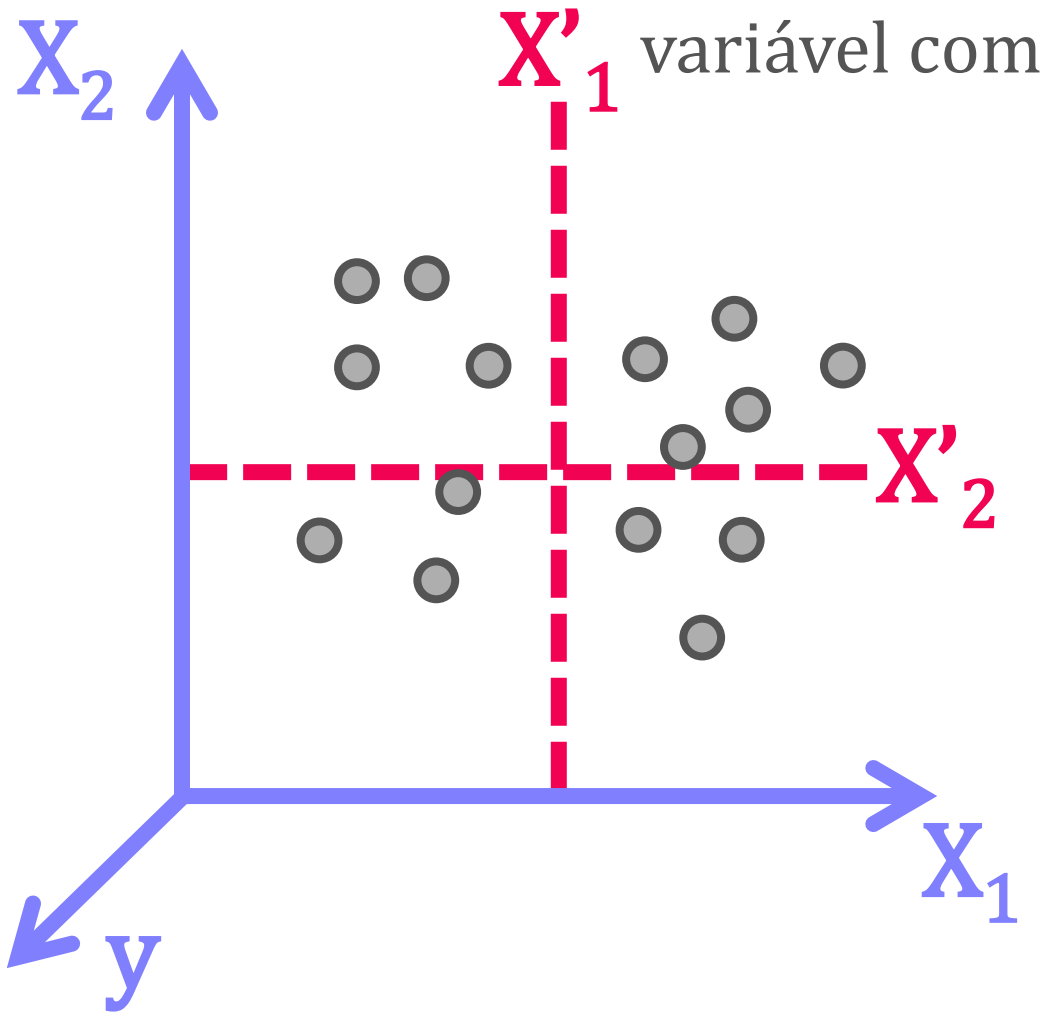
$$MSE_{total} = \frac{3}{5}\,0.616 + \frac{2}{5}\,0.16$$

$$= 0.4336 > 0.4130$$

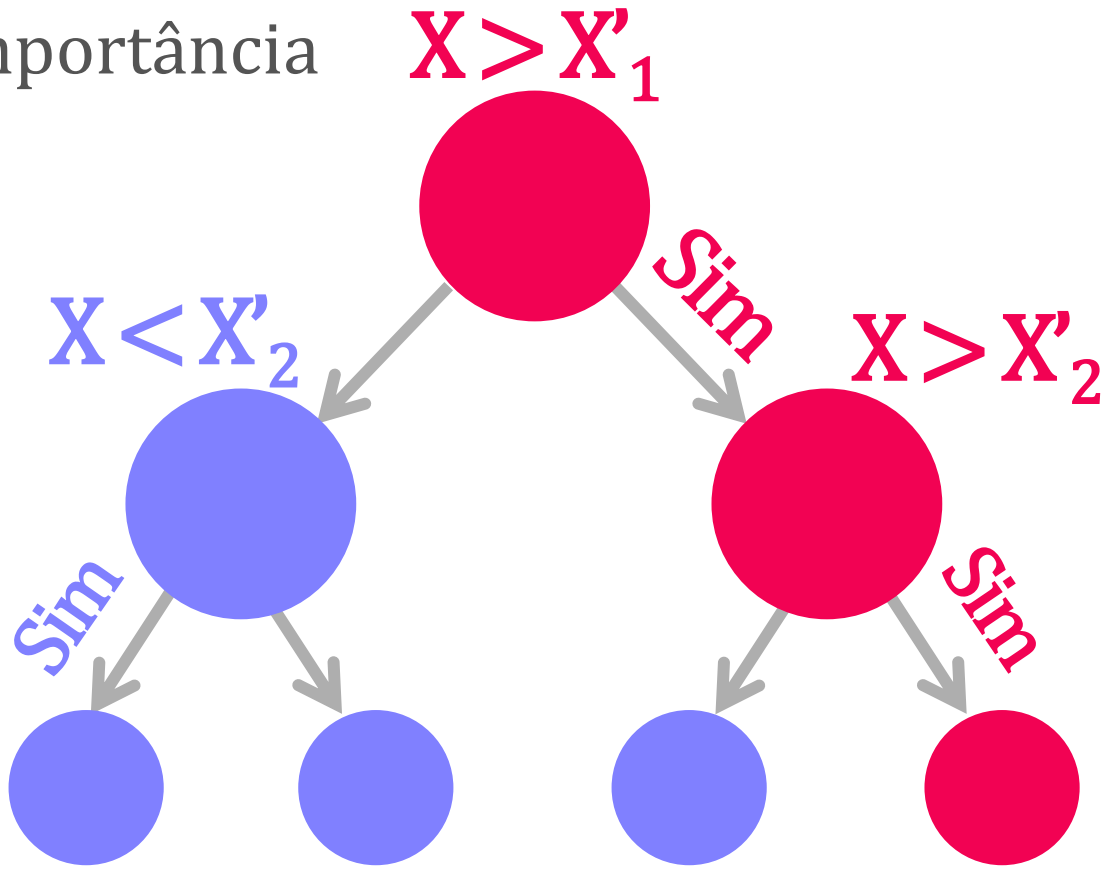
Qual o melhor ponto para dividir os dados? Ou em outras palavras, ser utilizado como o primeiro nó da arvore?

Random Forest – Regressão

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

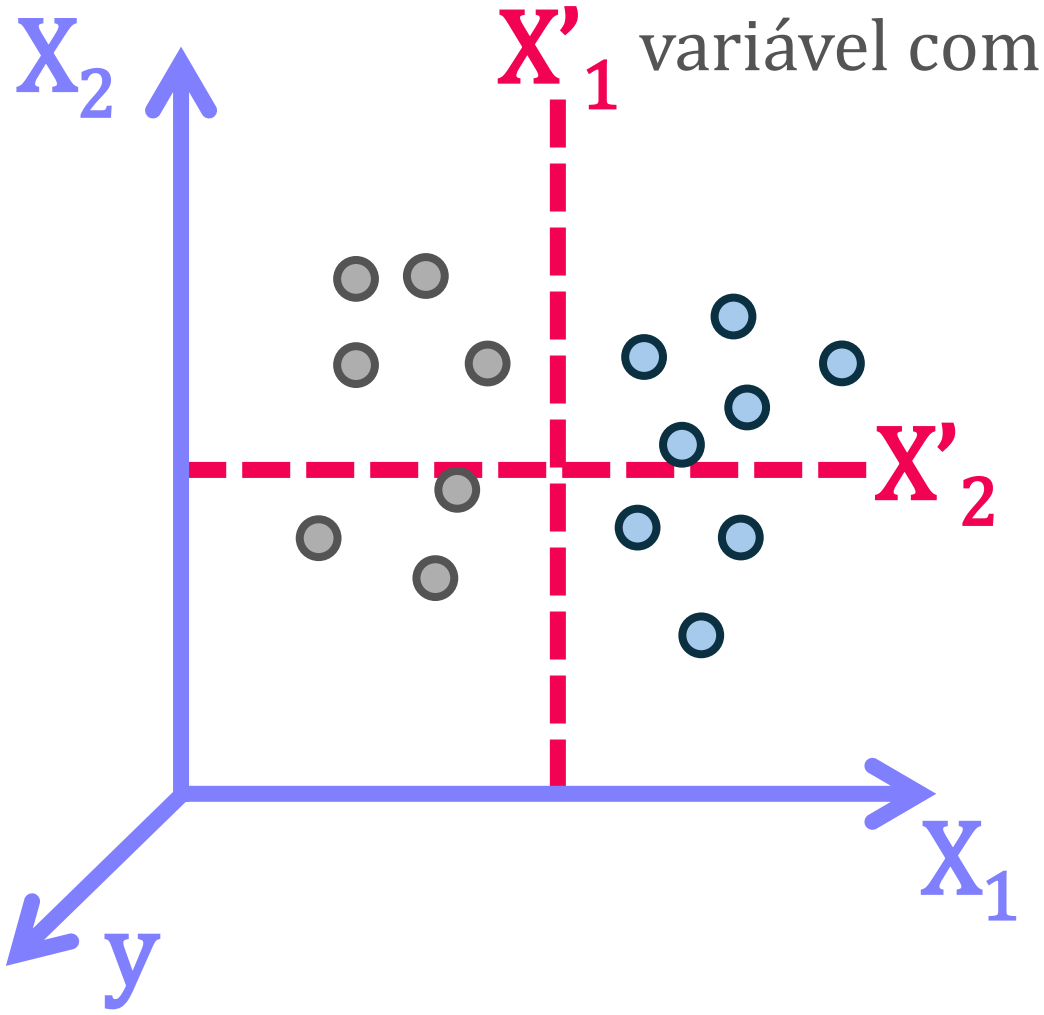


variável com alta importância

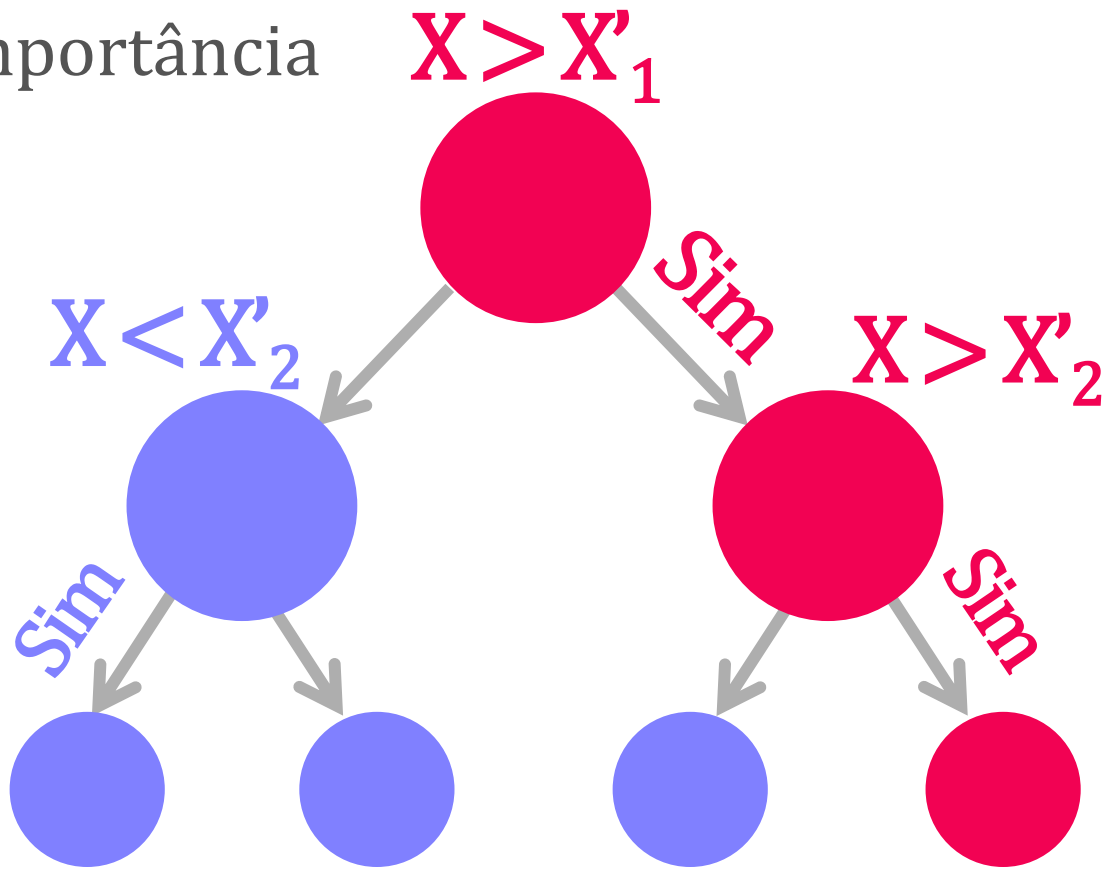


As predições finais são os y médios delimitados pelos hiperplanos

Random Forest - Classificação



variável com alta importância

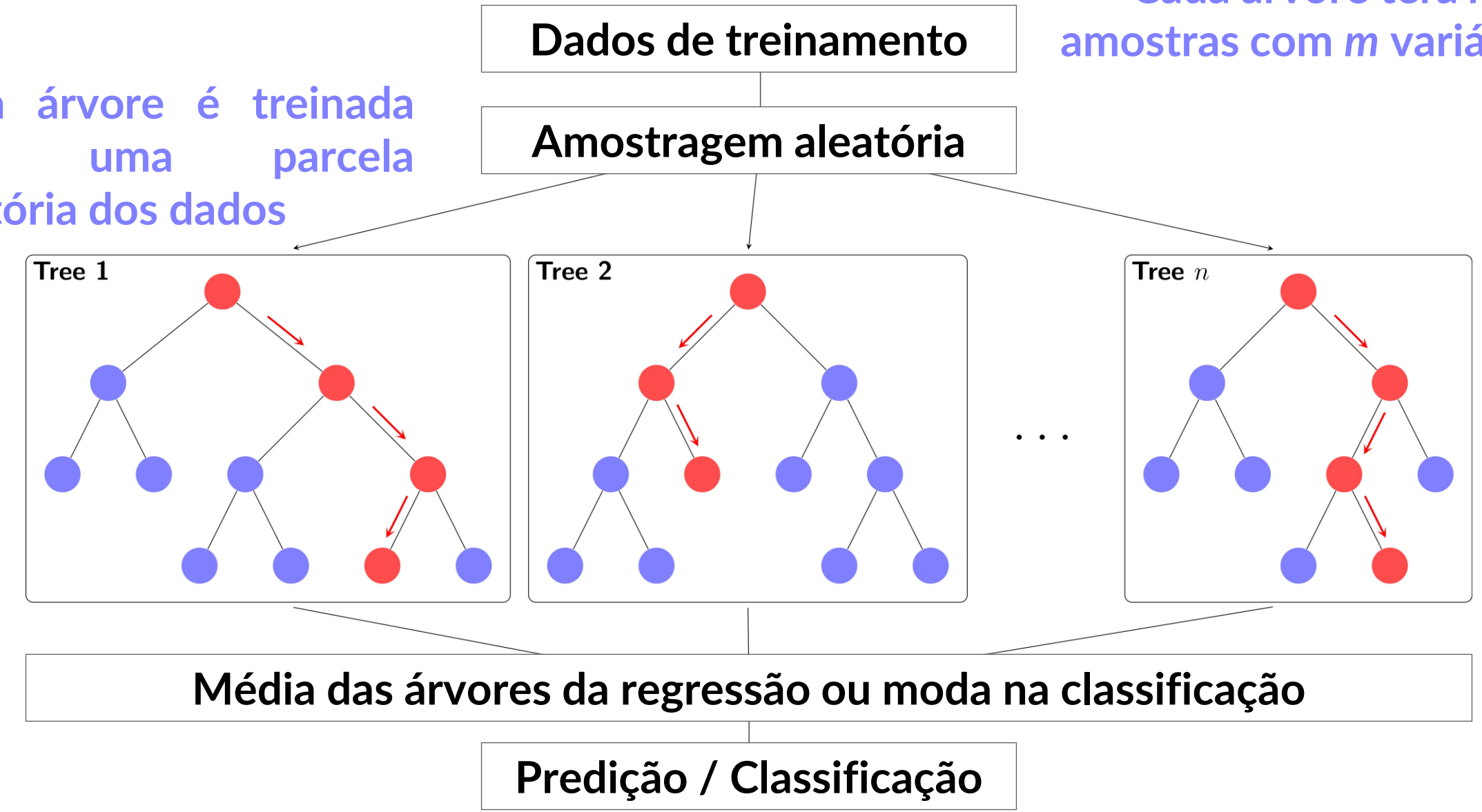


As predições finais são os y mais votados delimitados pelos hiperplanos

Random Forest - Bagging

Cada arvore terá *n* amostras com *m* variáveis

Cada árvore é treinada com uma parcela aleatória dos dados



Random Forest

Vantagens

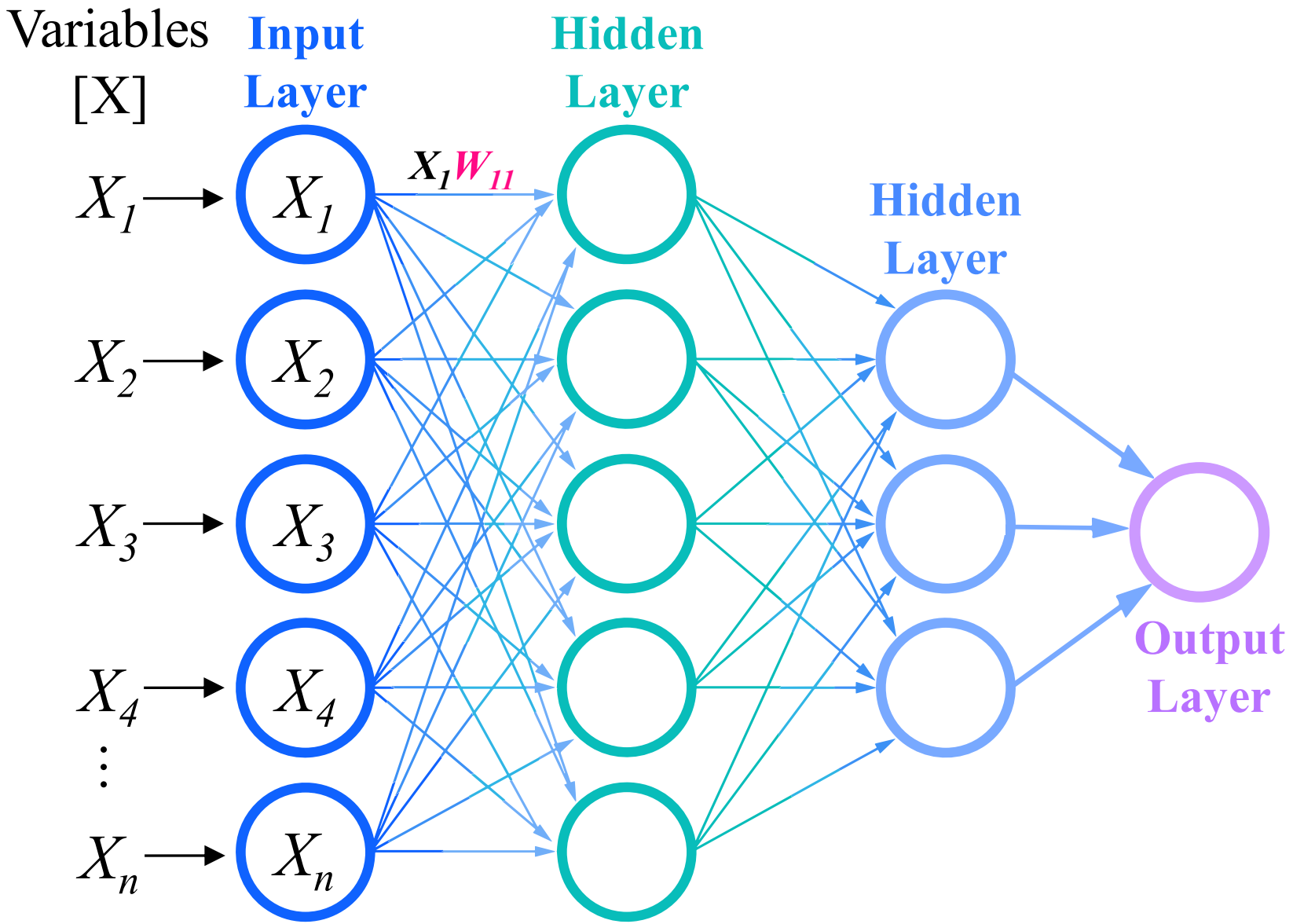
- Alta capacidade preditiva
- Poucos hiperparâmetros
- Realiza *feature importance*
- Lida bem com *datasets* grandes
- Gera métricas internas de erro
- Aleatoriedade combate o *overfitting*

Desvantagens

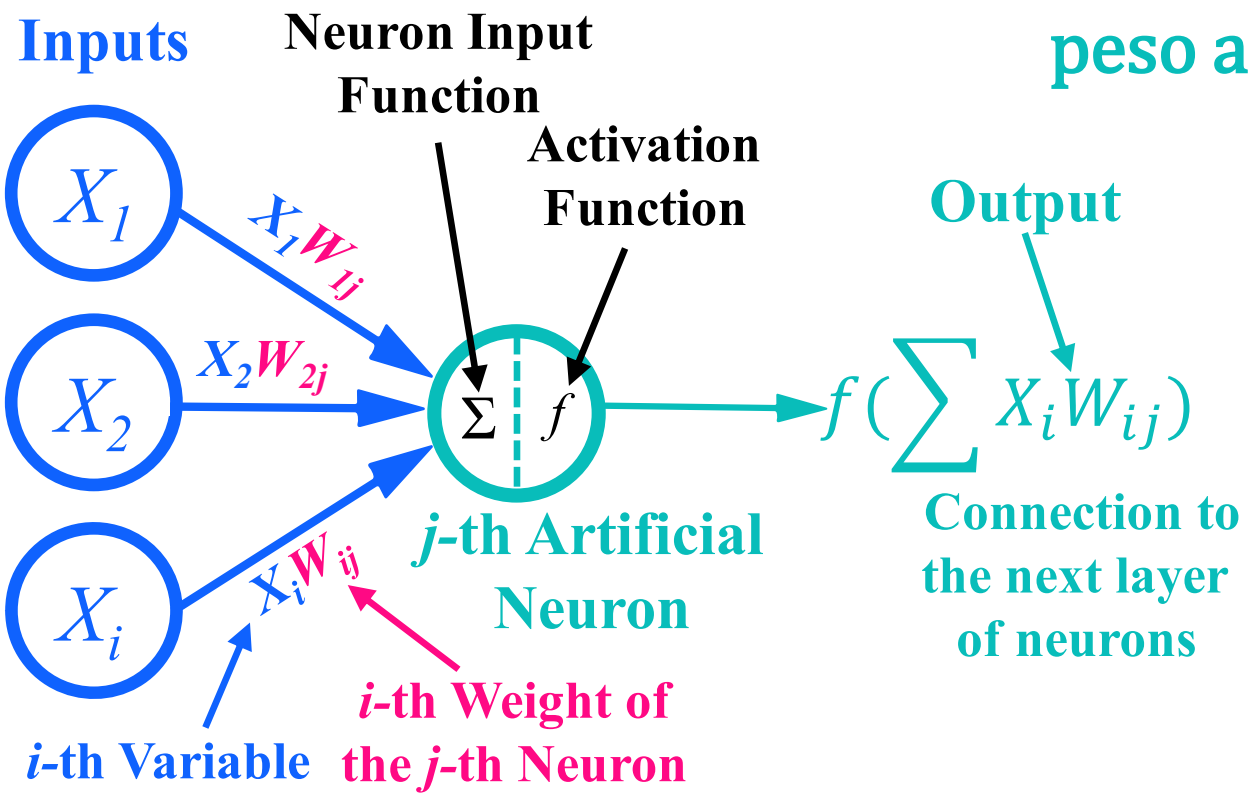
- Complexidade computacional
- Dependência dos hiperparametros
- Aleatoriedade
- Não lida bem imagem/som/texto

REDES NEURAIS CLÁSSICAS

REDE NEURAL – *MULTI LAYER PERCEPTRON*

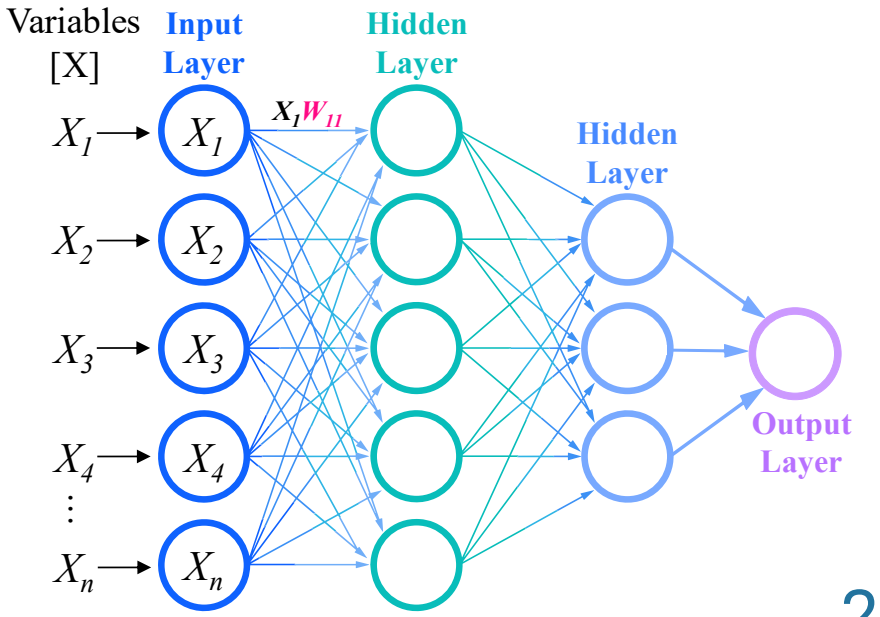


REDE NEURAL – *MULTI LAYER PERCEPTRON*

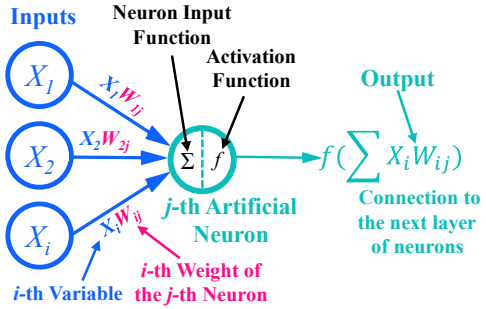


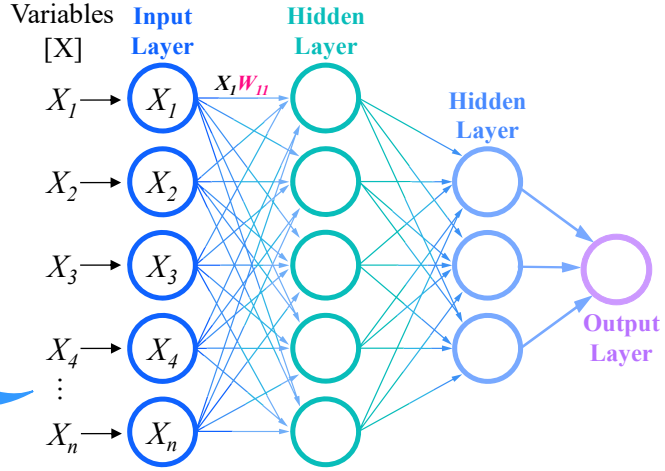
Cada neurônio associa um *i* - ésimo peso a *i* - ésima variável

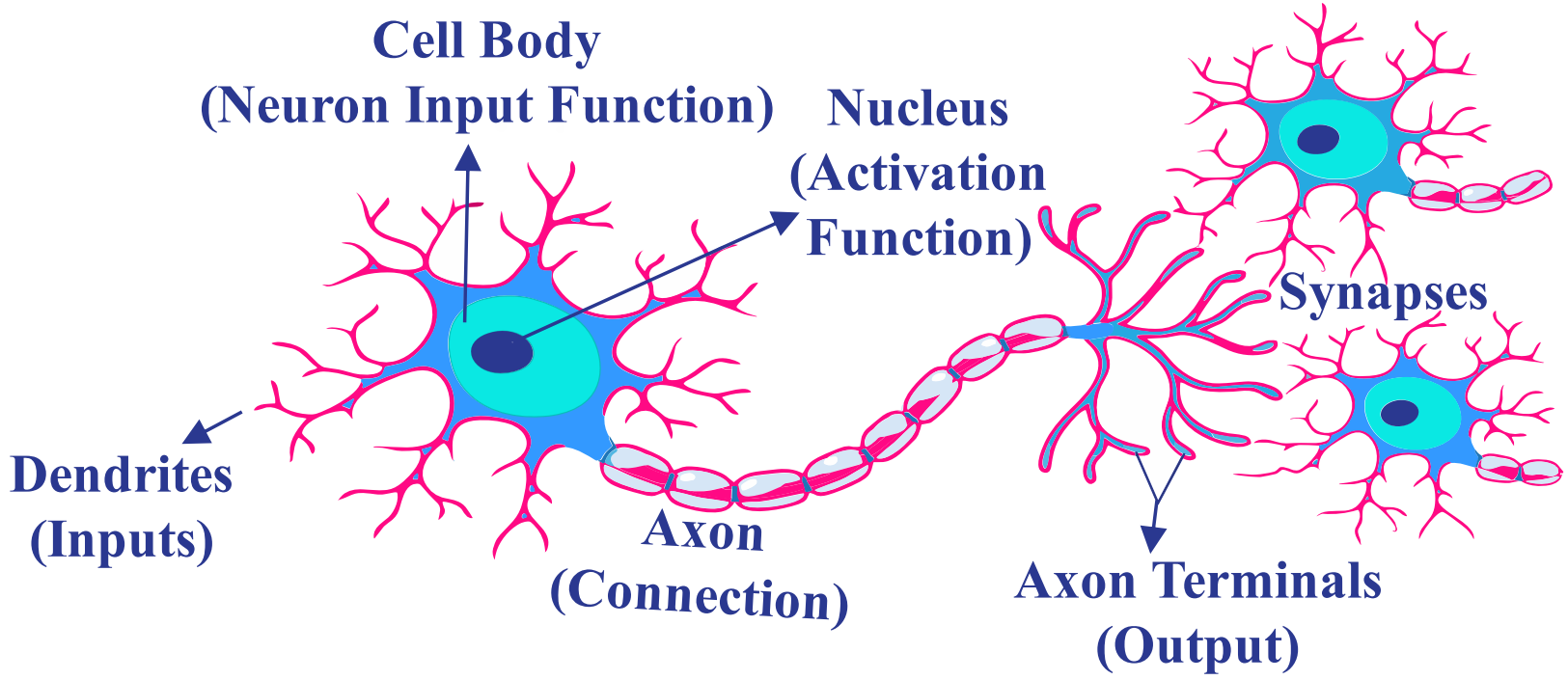
Em seguida, aplica uma função de ativação



REDE NEURAL – *MULTI LAYER PERCEPTRON*







REDE NEURAL – *MULTI LAYER PERCEPTRON*

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

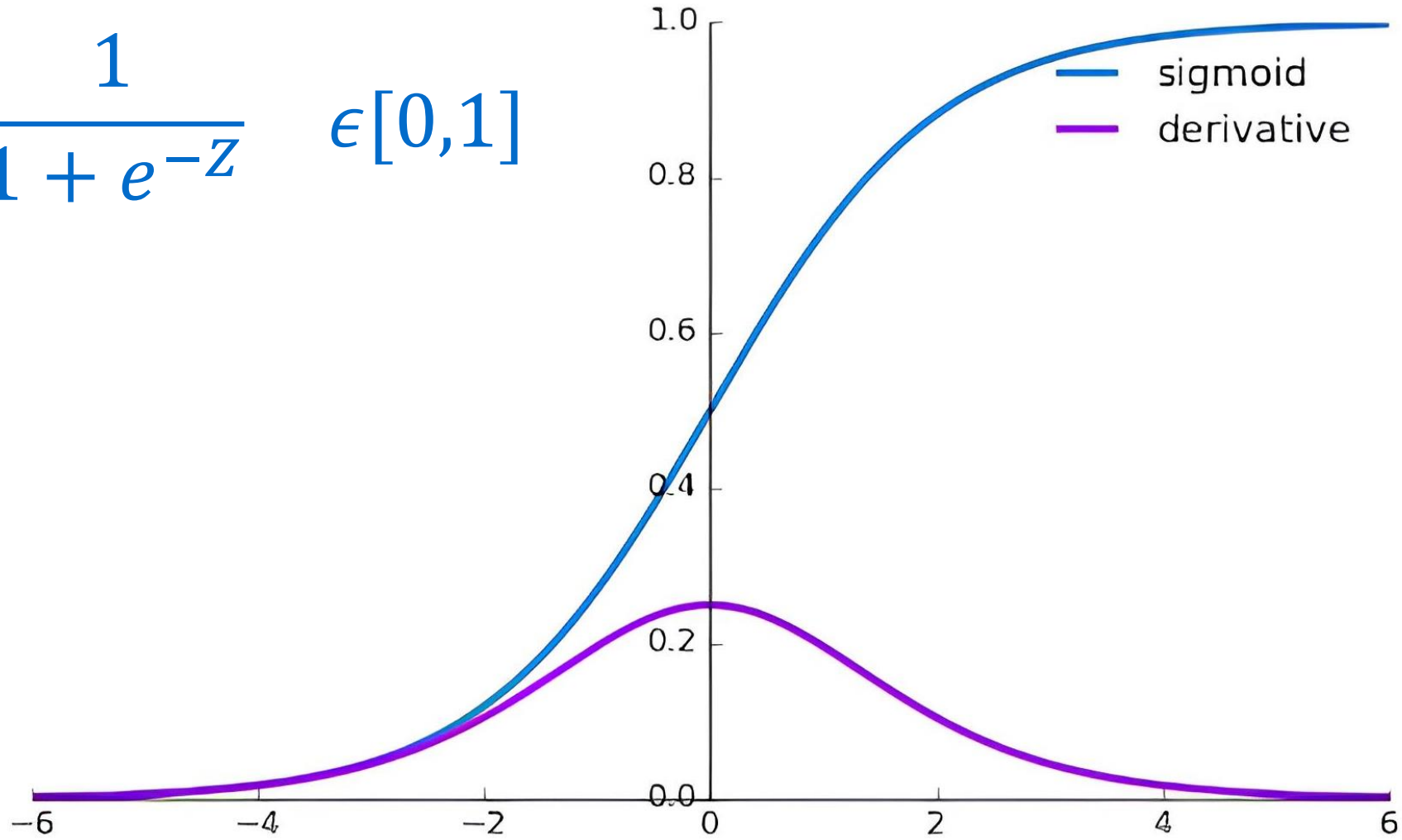
Principais objetivos

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

Principais tipos: sigmóide, tanh, ReLU, Softmáx

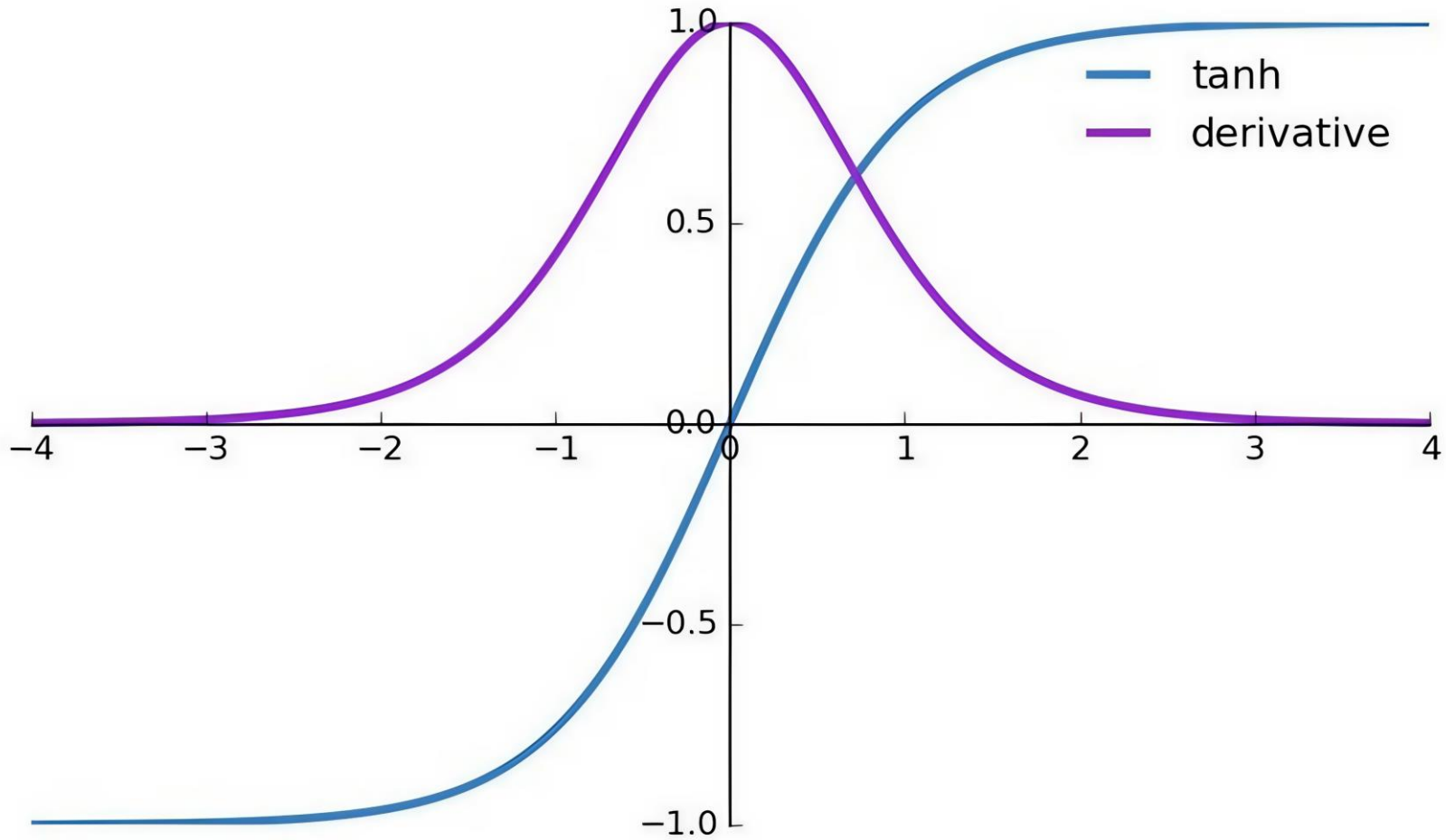
Funções de ativação: Sigmóide

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



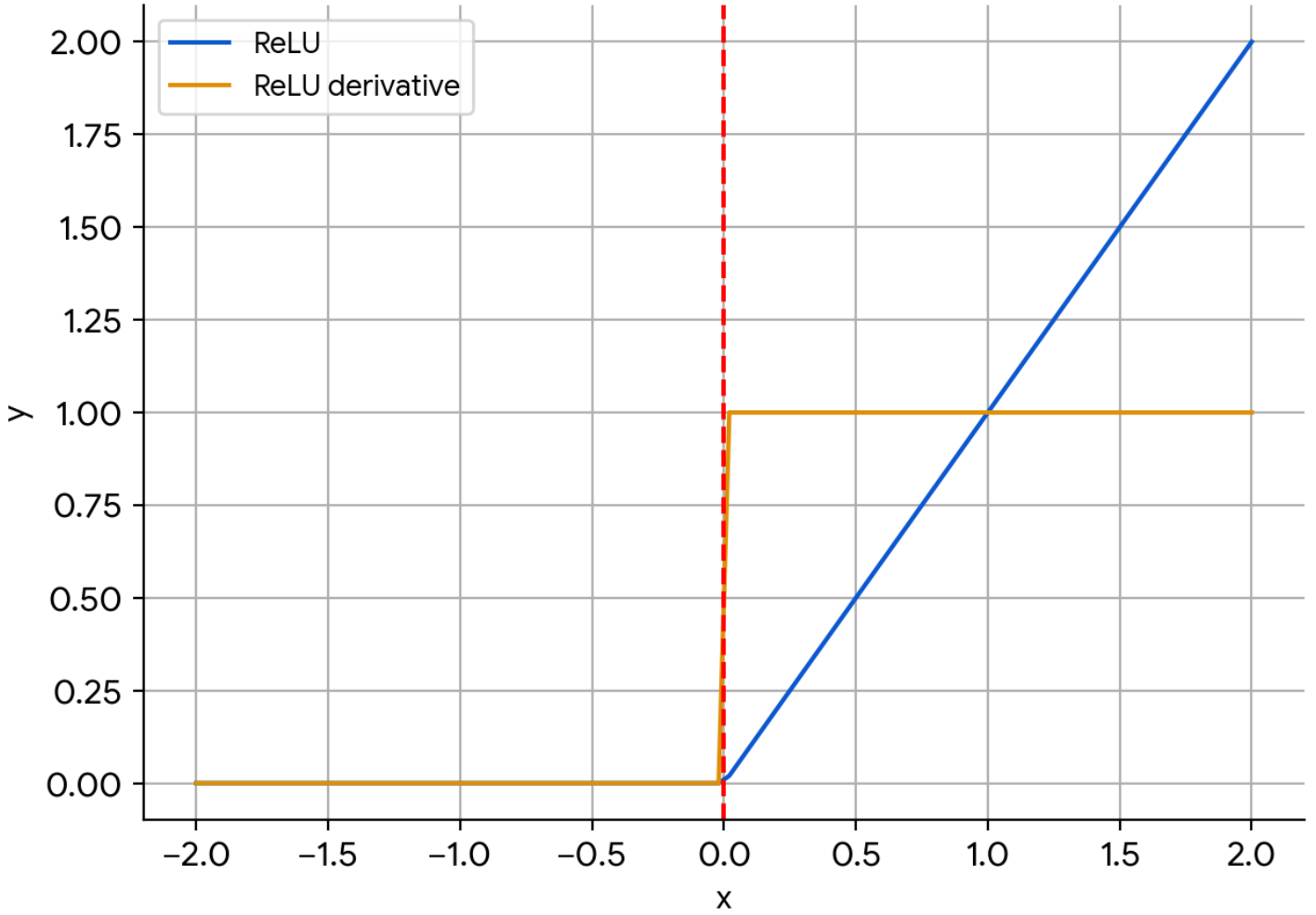
Funções de ativação: Tanh

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

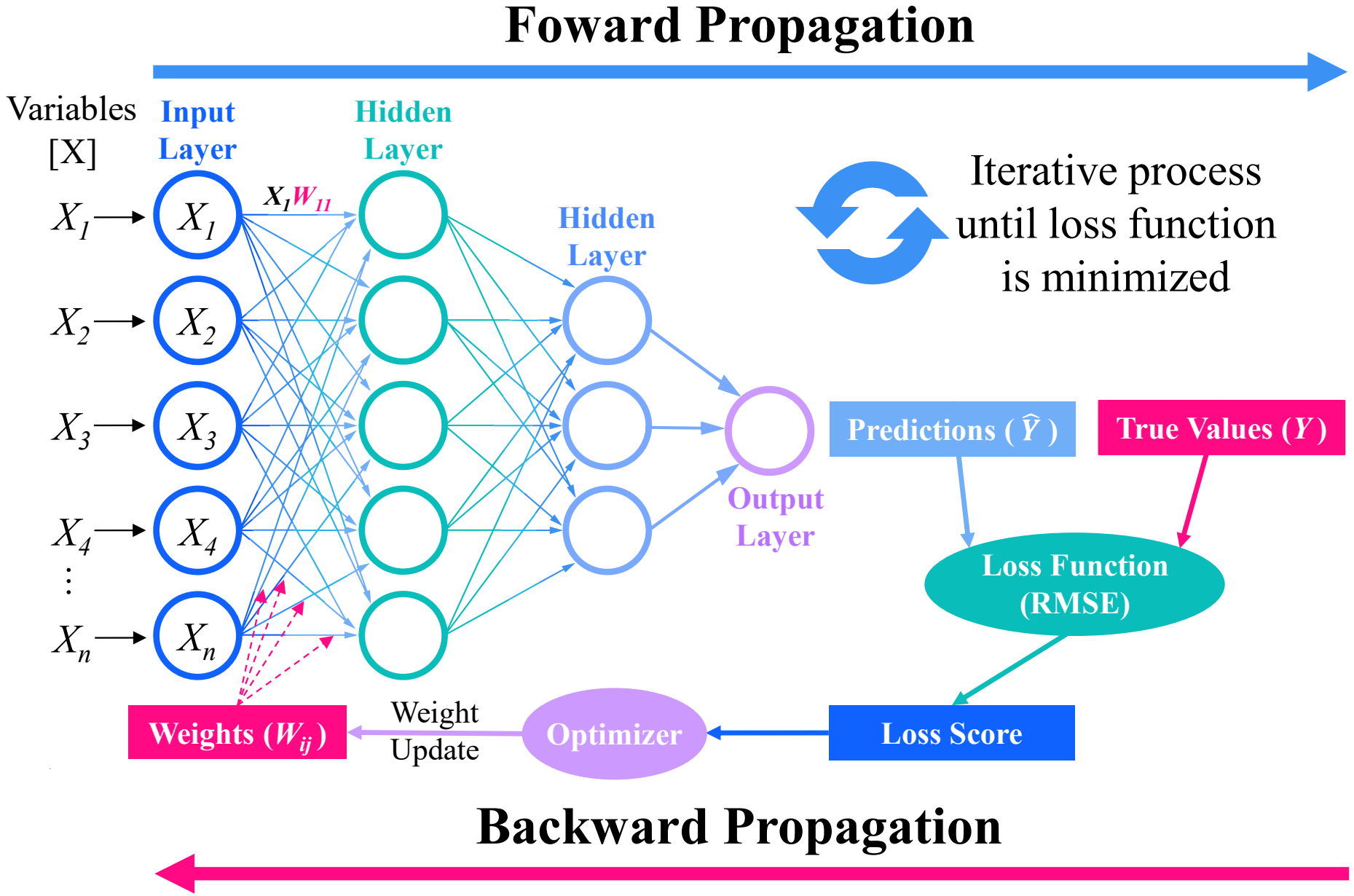


Funções de ativação: ReLU

$$ReLU(z) = 0\ se\ Z < 0$$
$$Z\ se\ Z > 0$$
$$\epsilon\ [0, \infty]$$



REDE NEURAL – *MULTI LAYER PERCEPTRON*



REDE NEURAL – *MULTI LAYER PERCEPTRON*

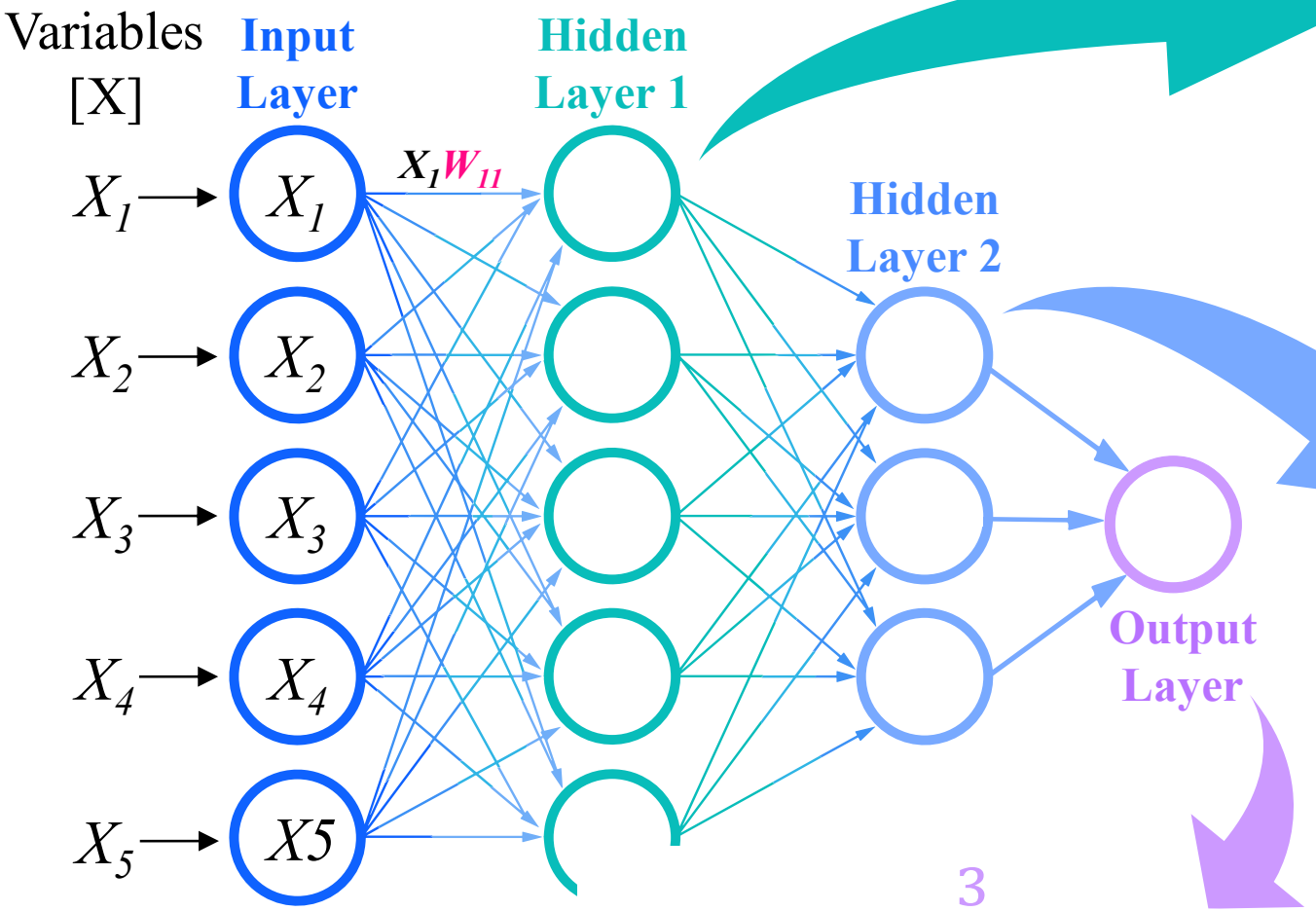
O objetivo da rede é gerar uma função com dependências em diversos parâmetros que, dado uma matriz X de variáveis de entrada, produz um vetor y com valores de saída

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



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

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

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

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

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

Nº pesos

5x5 + 3x5 + 3 = 43

REDE NEURAL – FASE *BACKWARD*

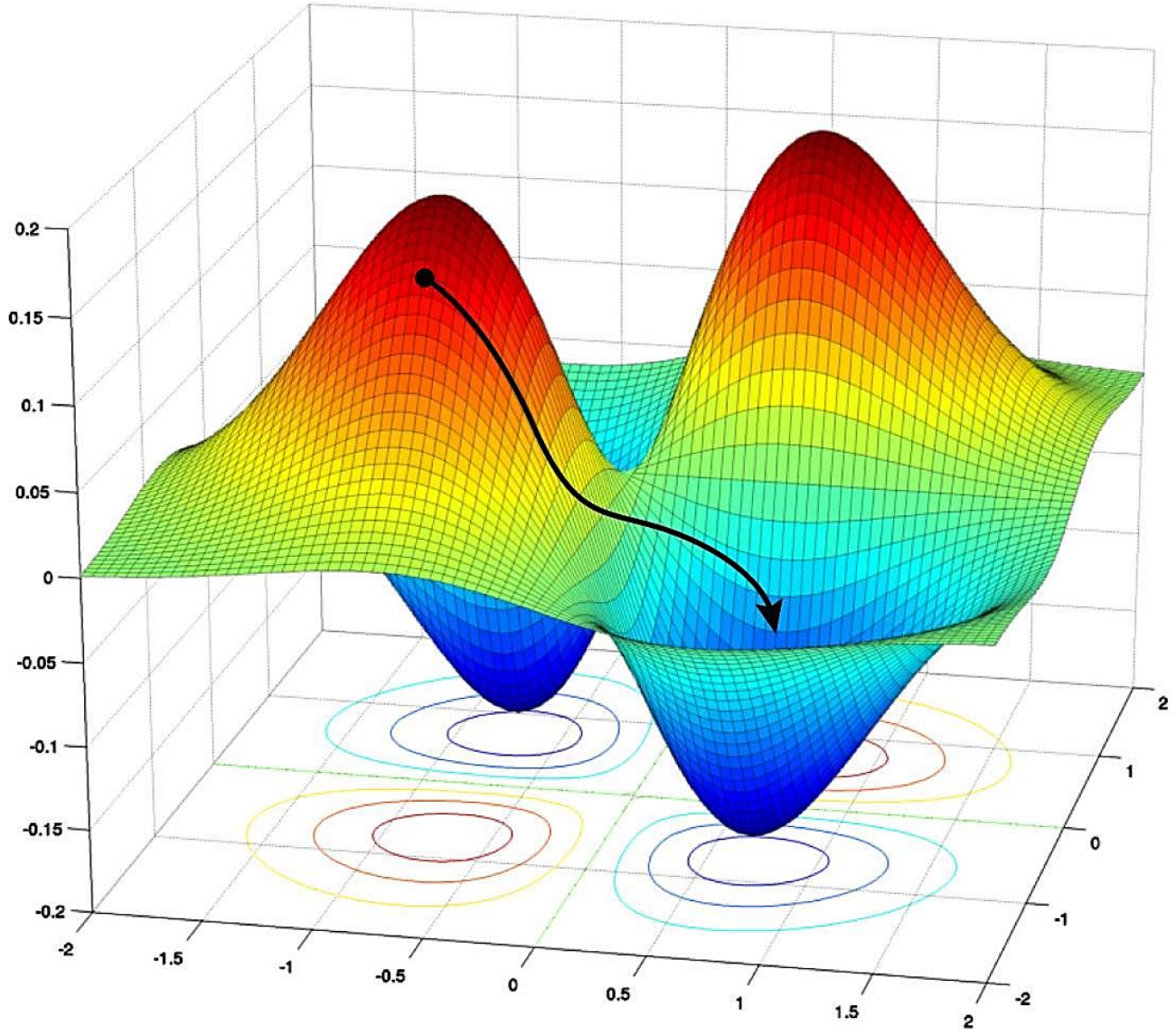
$$Loss = y - \hat{y} = L(\theta)$$

Queremos o conjunto de parâmetros θ^* que minimiza $L(\theta)$

Dado um conjunto de valores θ^t (dado pelas condições inicias), 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

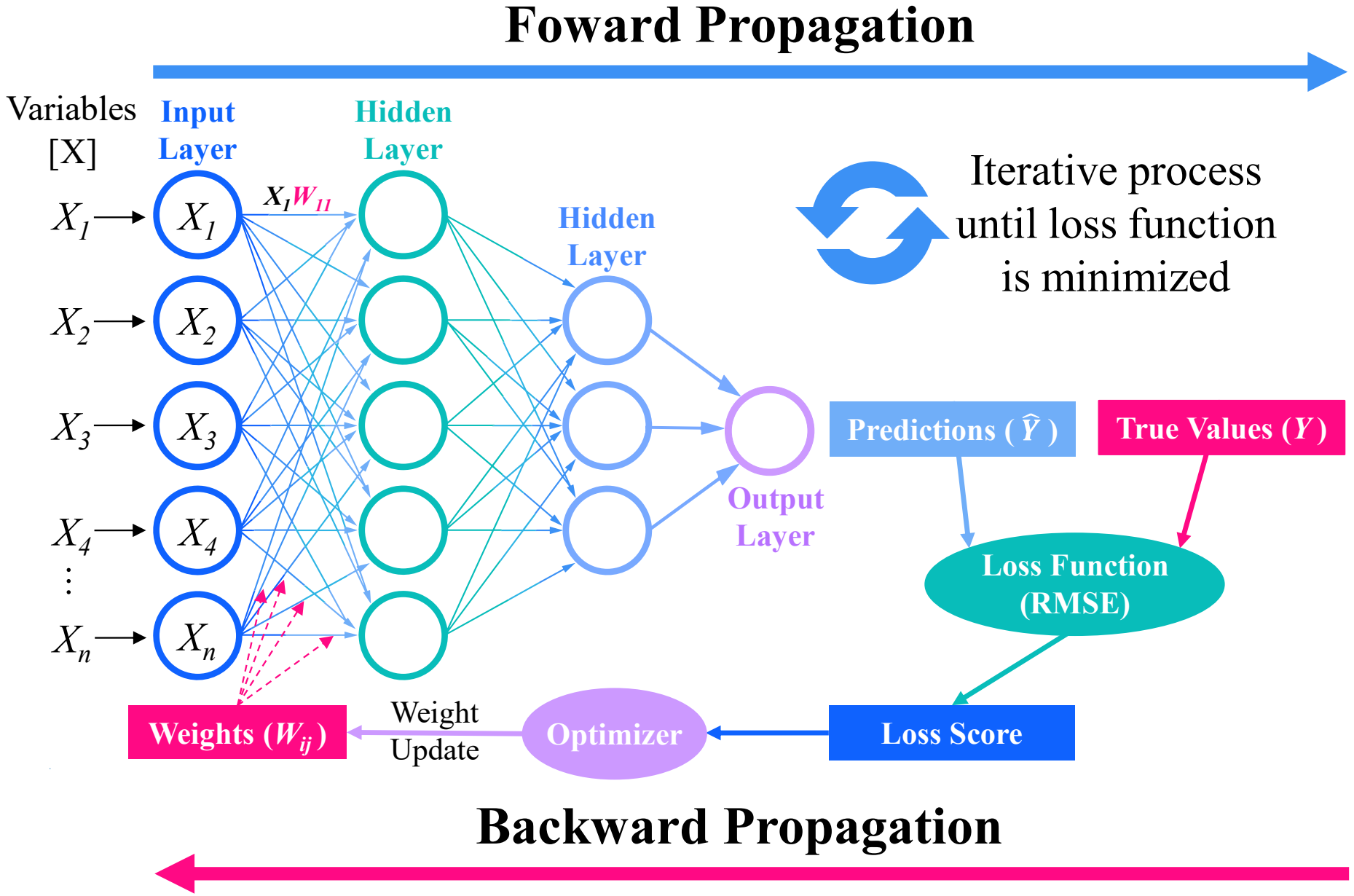
Na prática o cálculo é feito computacionalmente, por meio de algoritmos otimizados para o cálculo das derivadas

SGD (*Stochastic Gradient Descent* - 1950)

L-BFGS (*Limited-Memory* Broyden–Fletcher –Goldfarb–Shannon - 1989)

ADAM (*Adaptative Moment Estimation* - 2014)

REDE NEURAL – BACKPROPAGATION



REDE NEURAL – *MULTI LAYER PERCEPTRON*

Vantagens

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

Desvantagens

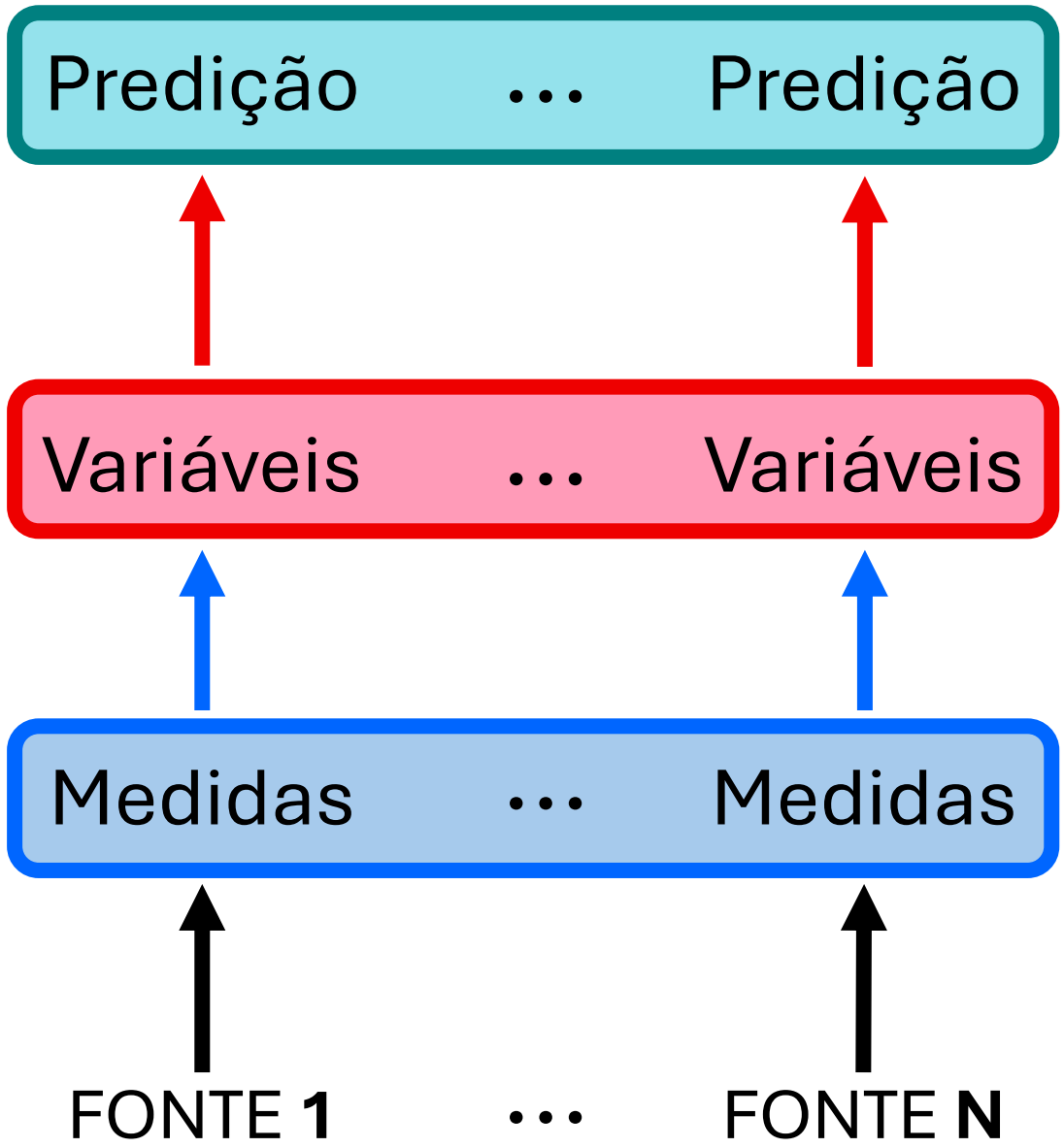
- Complexidade computacional
- Necessidade de muitos dados
- Propensão a *overfitting*
- Dificuldade de interpretar
- Complexidade no ajuste dos hiperparametros

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



Modelo → Predição

High-level fusion

Prediction level

Modelo → Predição

Mid-level fusion

Features level

Modelo → Predição

Low-level fusion

Measurements level

LOW-LEVEL FUSION

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

$$= \begin{pmatrix} X_{11}^{[1]} & \cdots & X_{1j}^{[1]} & \cdots & X_{11}^{[N]} & \cdots & X_{1j}^{[N]} \\ \vdots & & \vdots & & \vdots & & \vdots \\ X_{i1}^{[1]} & \cdots & X_{ij}^{[1]} & \cdots & X_{i1}^{[N]} & \cdots & X_{ij}^{[N]} \end{pmatrix}$$

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

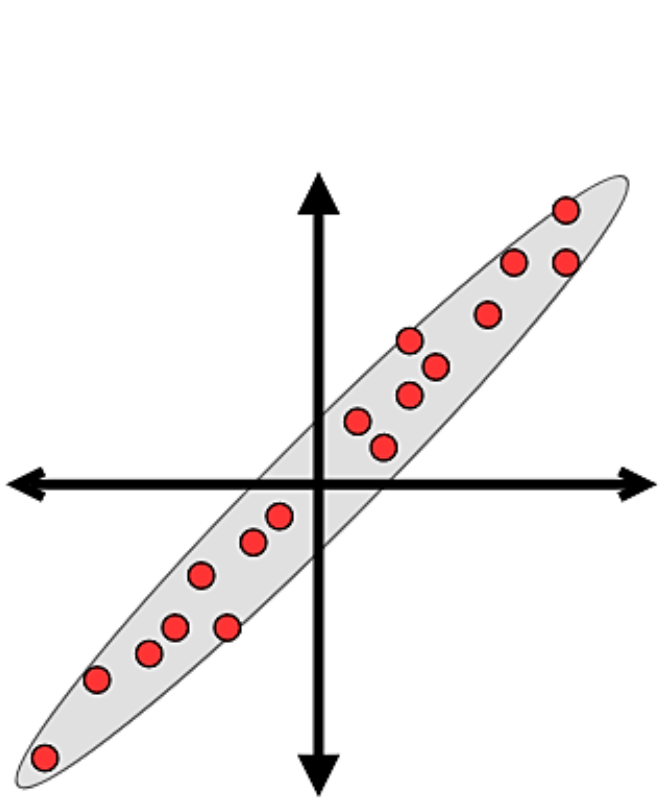
MID-LEVEL FUSION

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

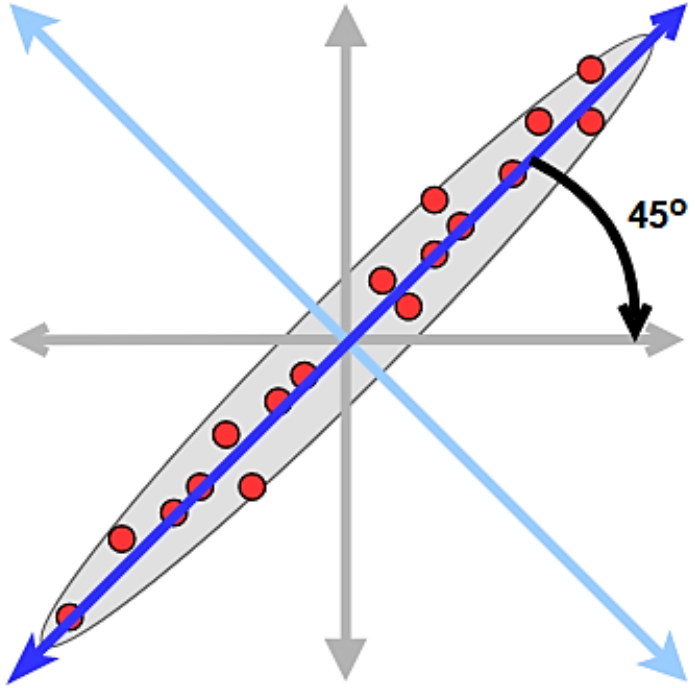
- Métodos para extração de *features complexas* são utilizados. Conhecido como *Features-level*
- Redução de dimensionalidade
- Seleção de variáveis

MID-LEVEL FUSION

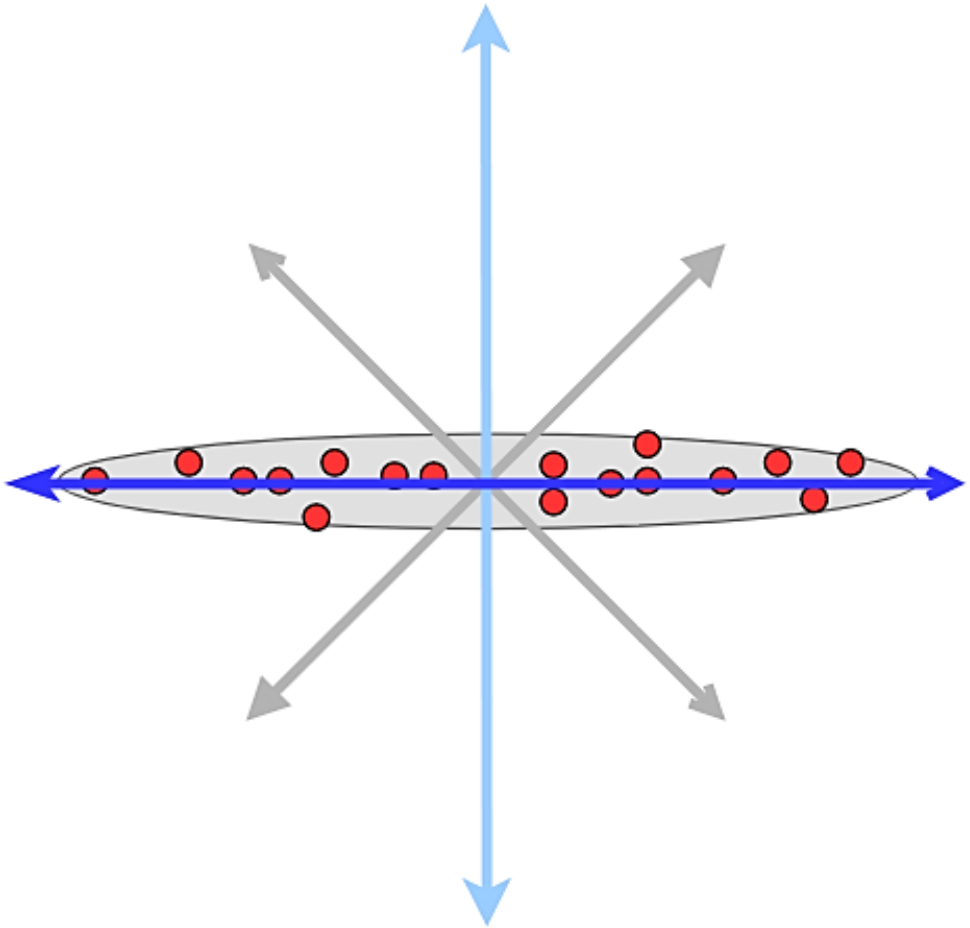
Exemplo: PCA



Original Coordinates



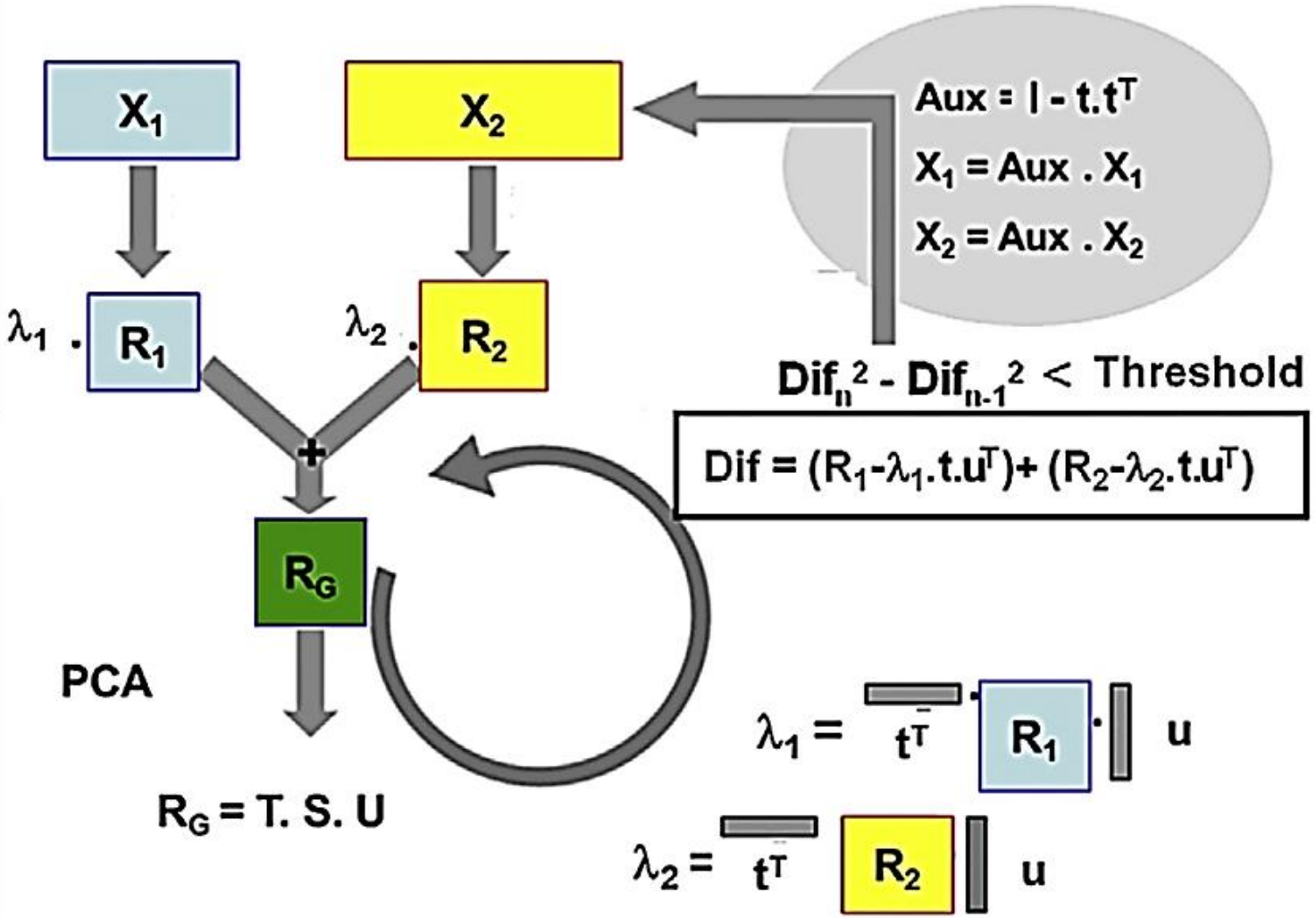
Principle Component



New Rotated Coordinates

MID-LEVEL FUSION

Exemplo: Análise de dimensões comuns (ComDim)



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

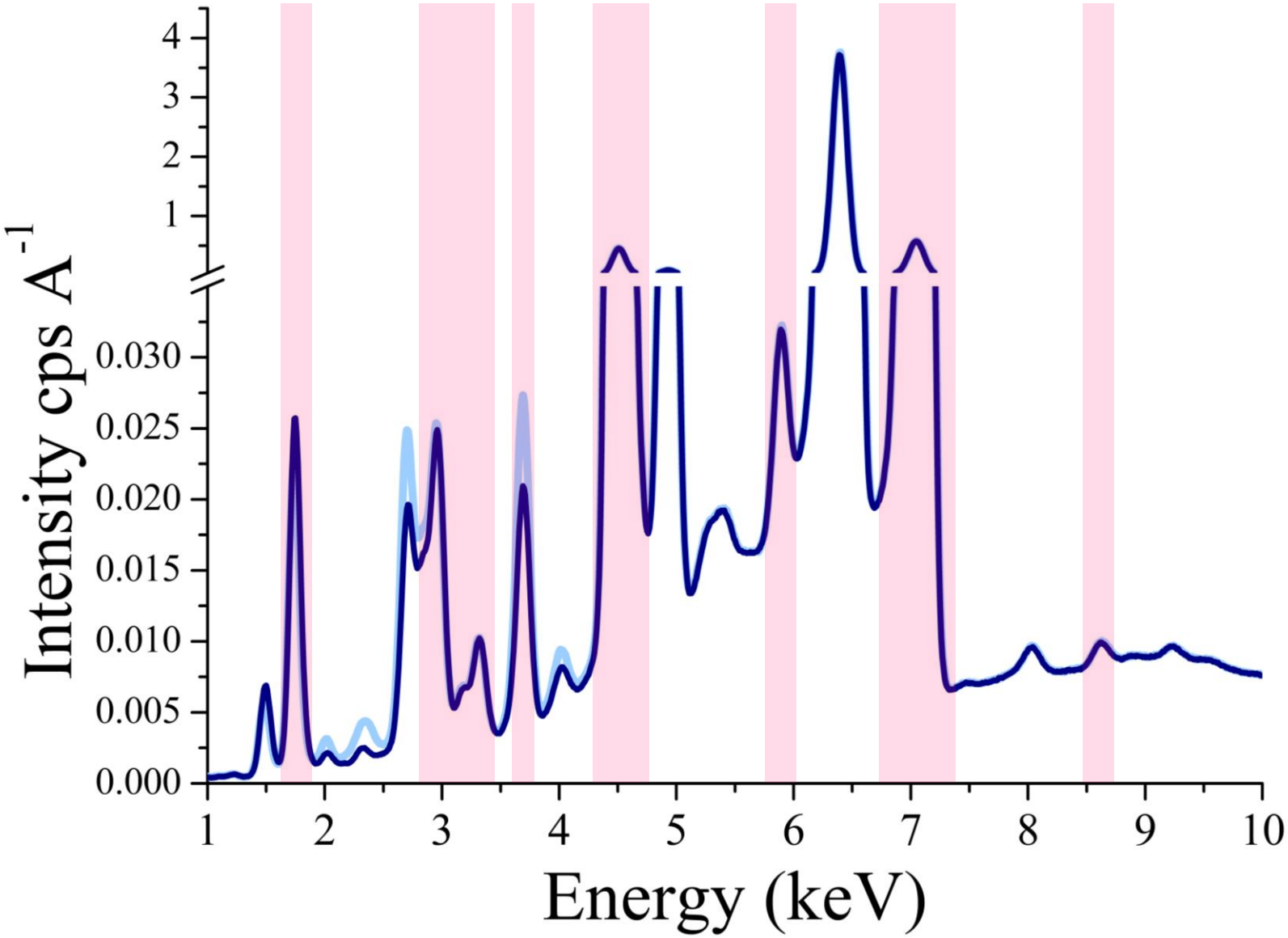
$i = 1, 2, \ldots n^o$ *de blocos*

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

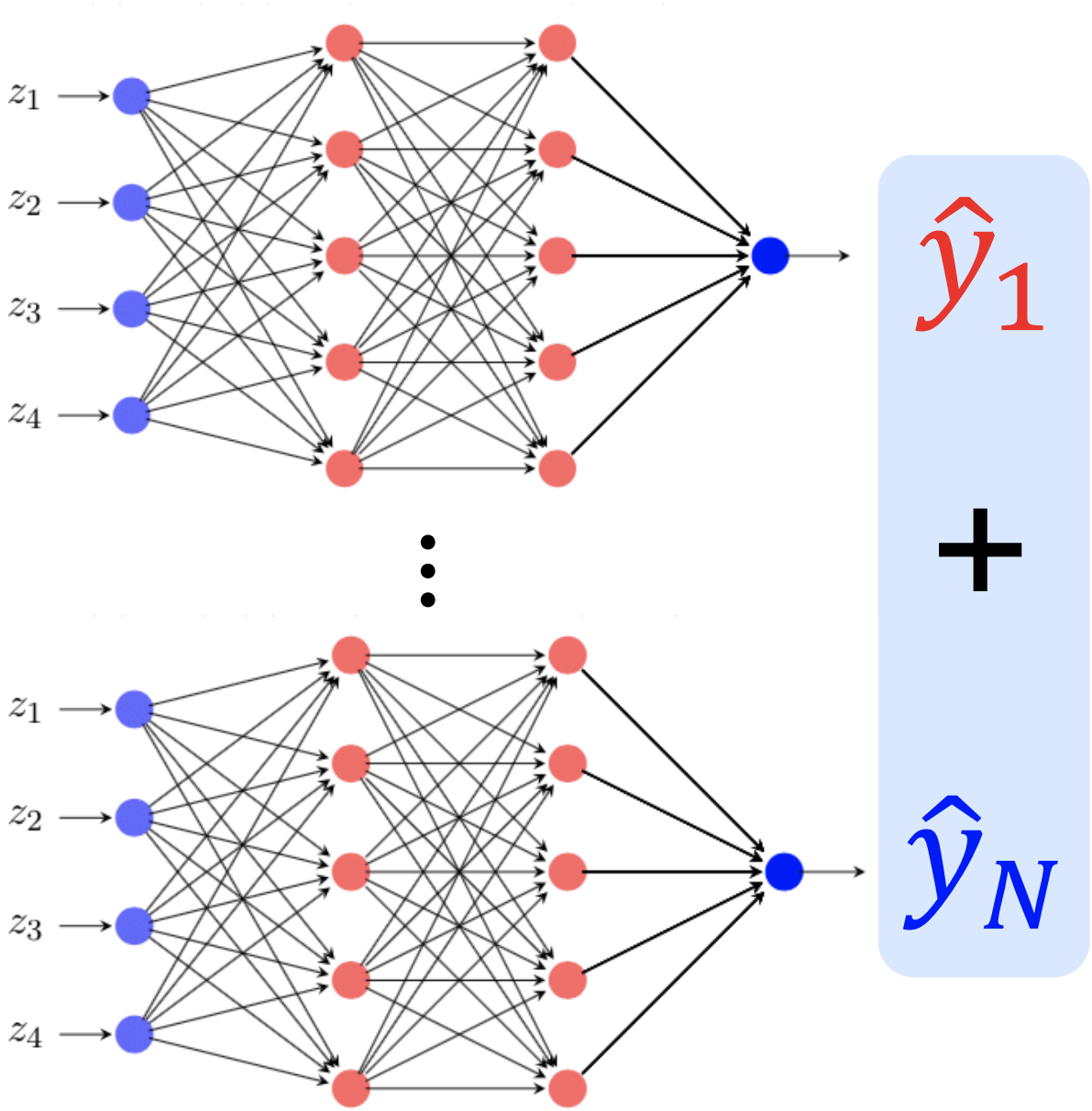
$$b = (T^t T)^{-1} X^t Y$$

MID-LEVEL FUSION

Ex: Seleção de variáveis



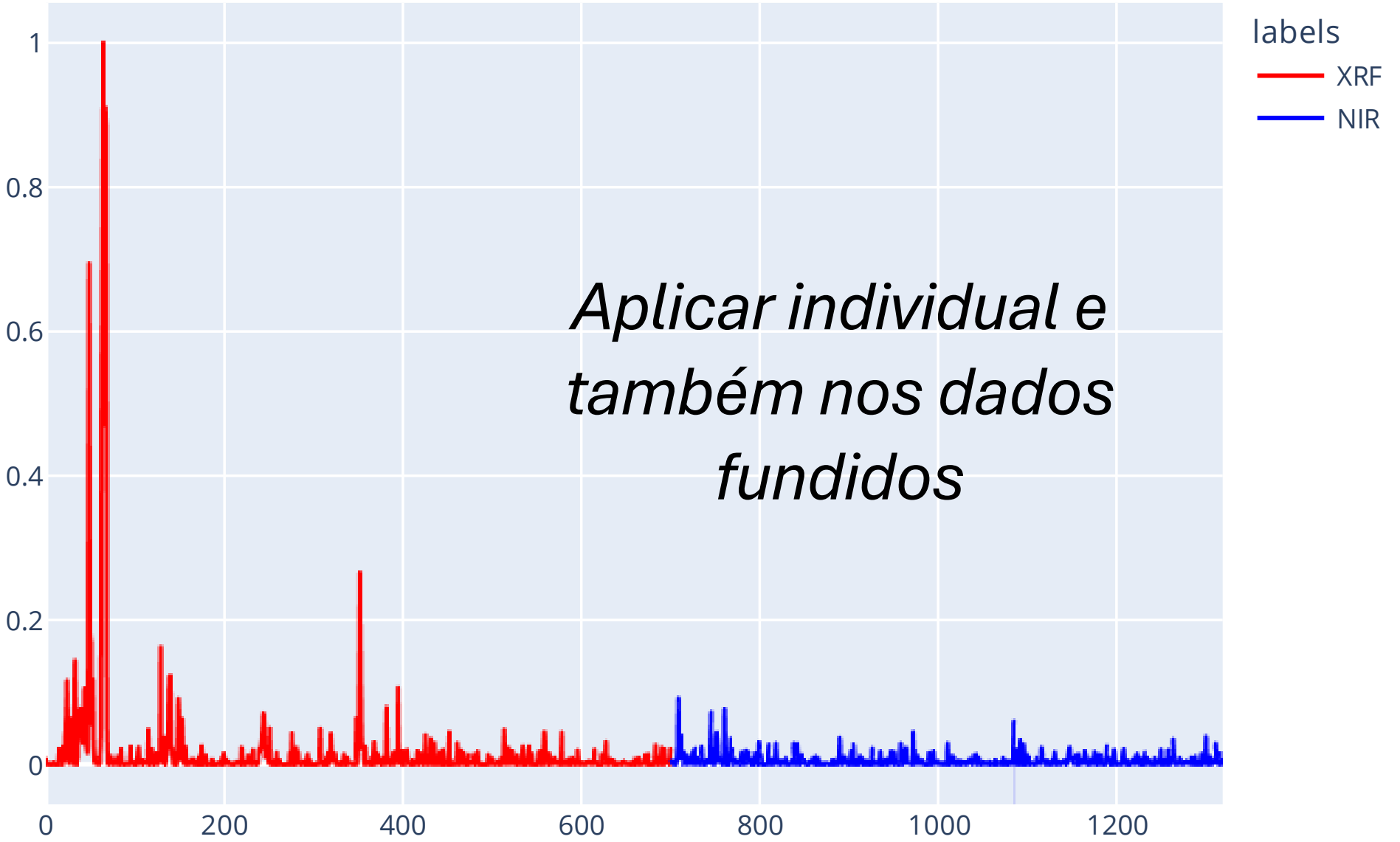
HIGH-LEVEL FUSION



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

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
- Necessário dados de mais de uma técnica
- *Sincronização e alinhamento de dados*
- Muitas variáveis (*overfitting*)

PRÁTICA – GOOGLE COLAB