Complexidade de Algoritmos Supervisionados

Machine Learning

Alexandro de Paula Charles Santos George Fabrício Pedro Brom

Projeto e Complexidade de Algoritmos (2025.1)

Planejamento

Dia 1 Será apresentado o **paradigma** da utilização de aprendizado de máquina em algoritmos supervisionados.

Dia 2 Serão apresentados **dois ou três algoritmos** com as respectivas complexidades e formulados matemáticos.

Dia 3 Será apresentado o **projeto** prático.

Sumário

Dia 1

Dia 2

Dia 3

Referências

Sumário — Dia 1

Contexto — Histórico

Áreas Cibernética e Ciência da Computação [Fradkov, 2020].

1957 O psicólogo **Frank Rosenblatt** identificou o uso de sinais analógicos e discretos para reconhecer letras, desenvolvendo o **perceptron**.

1960 **Vapnik**, **Chervonenkis**, **Yakubovich** e **Aizerman** contribuíram com teorias sobre a separação de classes usando hiperplanos lineares em espaços vetoriais.

Introdução — Vapnik & Chervonenkis

Teoria VC-dimensão e SVM linear: $f(x) = ax + b$.

Onde $W(x)$ é uma função de x valores de entrada ponderada pelos pesos w, e w_0 é bias e indica a posição da reta.

Kernel Aizerman inseriu a utilização do kernel através do $\phi(x)$ "phi de x", para que possa interpolar uma curva para separar as classes.

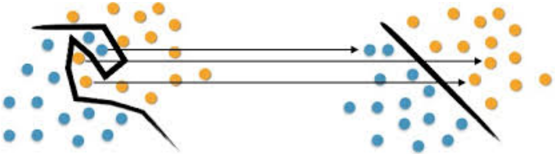


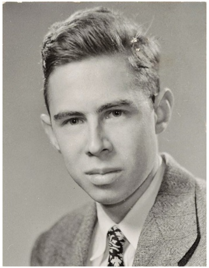
Figura: SVMs (lamfo-unb)

Introdução — Comparações

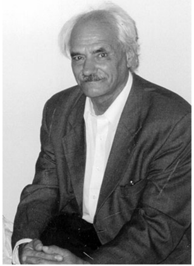
Ano	Modelo	Autor(es)	Característica Principal
1957	Perceptron	Rosenblatt	Separação linear simples
1962	Teorema do Perceptron	Novikoff	Prova de convergência
1963	Separação ótima	Vapnik	Hiperplano com margem máxima
1964	Formulação probabilística	Aizerman	Função potencial para classificação
1995	SVM Não Linear	Chervonenkis e Vapnik	Margem suave e kernel

Tabela: Primeiros modelos (adaptado de [Fradkov, 2020]).

Introdução — Autores



Frank Rosenblatt



Alexey Y. Chervonenkis

Vadim A. Trapeznikov

Vladimir Vapnik

Sumário — Dia 1

Fronteira — Paradigmas distintos

Embora diversos algoritmos de *Machine Learning* (**ML**) tenham origem em métodos estatísticos (**ME**), a aprendizagem de máquina (ML) não se limita a ser um desdobramento da estatística [Breiman, 2001]:

ME foco em inferência e explicação, com modelos probabilísticos bem definidos.

ML foco em desempenho preditivo e aproximação funcional.

Regressão linear sob a ótica da Estatística [Hastie et al., 2009]:

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top Y$$

$$\varepsilon \sim \mathcal{N}(0, \sigma^2), \quad \varepsilon_i \perp \varepsilon_j \text{ para } i \neq j.$$

Fronteira — Zona cinzenta (técnicas compartilhadas)

Regressão linear sob duas óticas [Hastie et al., 2009]:

Estatística

- Solução analítica.
- Pressupostos probabilísticos.
- Ênfase em inferência e causalidade.

Aprendizado de Máquina

- Otimização numérica (ex: gradiente descendente).
- Foco em desempenho preditivo.
- Menor ênfase em interpretabilidade.

- **A estatística clássica** se desenvolveu devido às restrições por limitações computacionais.
- **O predomínio** é de métodos analíticos fechados.
- O avanço computacional viabilizou:
  1. a inferência **Bayesiana**;
  2. a inferência **variacional**; e
  3. o Monte Carlo **Markov** Chain (MCMC)[1].

[1] https://github.com/pcbrom/bgumbel
[1] https://www.ajs.or.at/index.php/ajs/article/view/1392

Fronteira — O problema da alta dimensionalidade

- Quando $d > n$, a matriz $\mathbf{X}^\top\mathbf{X}$ torna-se singular [Wainwright, 2019].
- Os **métodos estatísticos clássicos perdem aplicabilidade**, uma vez que os pressupostos não estão satisfeitos.
- As técnicas heurísticas e de otimização **ganham protagonismo**.

Fronteira — Inferência causal: limites do ML

No *Machine Learning* [Pearl, 2009]:

- A **ênfase recai sobre a previsão** em vez da causalidade.
- Alguns otimizadores aprisionam soluções em **mínimos locais**.
- Há carência de **garantias** para a validade global.
- A **estatística** oferece estrutura probabilística para inferência causal.

Sumário — Dia 1

Formulação — Problema de classificação

Objetivo encontrar uma regra de decisão que aproxime uma função desconhecida f^* tal que

$$h(\mathbf{x}) \approx f^*(\mathbf{x}).$$

Dados

$\mathbf{x}_i \in \mathbb{R}^d$	vetor de entrada.
$z_i \in \{-1, 1\}$	rótulo.
$\mathbf{X} \in \mathbb{R}^{n \times d}$	matriz de entrada.
$S_n = \{(\mathbf{x}_i, z_i)\}_{i=1}^n$	conjunto de treinamento.

Risco verdadeiro,

$$L(h) = \mathbb{P}(h(\mathbf{X}) \neq f^*(\mathbf{X})).$$

Risco empírico,

$$R_n(h) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}[h(\mathbf{x}_i) \neq z_i].$$

Onde $\mathbb{I}[\cdot]$ é a função indicadora.

Formulação — Aprendizado empírico e hipóteses

ML O algoritmo A encontra h_n que minimiza o risco empírico:

$$h_n = A(S_n) \in \arg\min_{h \in HS} R_n(h),$$

onde HS é o conjunto de hipóteses (modelos admissíveis).

Formulação — Generalização do modelo

Objetivo Garantir que h_n tenha baixo erro em novos dados.

Depende do tamanho da amostra n, e
da complexidade de HS (medida pela dimensão VC).

Formulação — Desigualdade de generalização

Tem alta probabilidade

$$\mathbb{P}(L(h_n) \leq \varepsilon) \geq 1 - \delta$$

Se atende

$$n > \frac{VC(HS) + \log(1/\delta)}{\varepsilon^2}$$

Formulação — Ilustração geométrica de VC

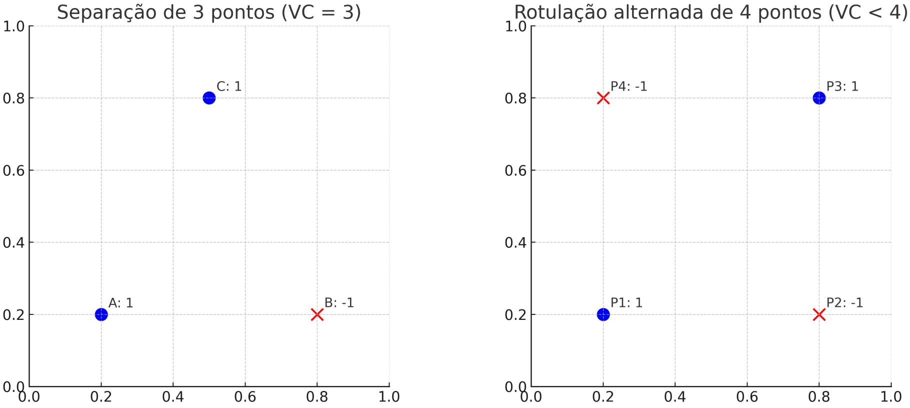


Figura: Exemplo geométrico da dimensão VC.
Esquerda: $\ell = 3$, separável. **Direita**: $\ell = 4$, não separável.

Formulação — Rotulações

Config.	z_1	z_2	z_3
(1)	-1	-1	-1
(2)	-1	-1	$+1$
(3)	-1	$+1$	-1
(4)	-1	$+1$	$+1$
(5)	$+1$	-1	-1
(6)	$+1$	-1	$+1$
(7)	$+1$	$+1$	-1
(8)	$+1$	$+1$	$+1$

Tabela: Todas as rotulações possíveis para $\ell = 3$.

Formulação — Exemplo numérico

Desejamos Cálcular amostra mínima de $\varepsilon = 0.1$, $\delta = 0.05$ e $VC = 3$.

$$n > \frac{3 + \log(1/0.05)}{(0.1)^2} = \frac{3 + \log(20)}{0.01} \approx \frac{5.9957}{0.01} = 599.6$$

Resultado São necessárias, ao menos, **600 observações**.

Formulação — Dimensão VC

Classe de hipóteses	**Dimensão VC**
Retas no plano	3
Hiperplanos em $\mathbb{R}^d$	$d+1$
Árvores de decisão com k folhas	$\mathcal{O}(k \log k)$
Redes neurais com W pesos (ReLU)	$\mathcal{O}(W \log W)$

Tabela: Dimensão VC para diferentes modelos.

Sumário — Dia 1

- Introdução
- Fronteira conceitual
- Formulação do ML
- Complexidade Computacional
- Otimização

Complexidade — Influência

- Quais fatores influenciam o **custo computacional** na utilização de aprendizado de máquina?

- Ultimamente, depende de:

  1. tempo e recursos de *hardware* necessários para treinar e aplicar seus modelos [Goodfellow et al., 2016]; e
  2. aplicações utilizadas.

Complexidade — Contexto de ML

Contexto O avanço da tecnologia, propiciado pela **Lei de Moore** e pelo desenvolvimento de **GPUs** e **TPUs**, possibilitou a utilização de modelos de complexidade crescente.

Conseq. A viabilidade de treinar esses modelos em ambientes com recursos escassos destaca a importância de administrar a complexidade computacional associada ao aprendizado de máquina.

Complexidade — Pontos positivos

1. Permitiu aprendizagem de padrões mais complexos e realização de tarefas de alta precisão.

   Exemplos: ResNet, ResNet-50 e ResNet-101.

2. Compreensão de linguagem natural com alta acurácia; previsão de estruturas com precisão sem precedentes; realização de raciocínios complexos que se aproximam do desempenho humano.

   Exemplo: AlphaGo (*deep reinforcement learning*).

Complexidade — Pontos negativos

1. Custo ambiental e financeiro.

   Projetos de predição de crimes ou doenças que utilizaram modelos excessivamente complexos, sem transparência, levando a resultados com alto viés, difíceis de interpretar e muitas vezes ineficazes.

   Exemplo: GPT-3 [Strubell et al., 2020].

   `https://huggingface.co/docs/hub/model-cards-co2`

Complexidade — Pontos negativos

2. Tempo de resposta e requisitos de latência.

   Edge computing (IoT): a inferência local de modelos complexos é inviável devido às restrições computacionais.

   Exemplo: Aplicações de tempo real.

Como? Reduzindo tamanho do modelo.

1. Quantização;
2. *Pruning*;
3. Algoritmos neuromórficos;
4. Hardware especializado;
5. Computação quântica; e
6. *Distillation* [Mullapudi et al., 2019].

Complexidade — *Model Distillation*

Como? Técnica de compressão onde **um modelo complexo** ("professor") **treina um modelo menor** ("aluno").

Profesor Modelo grande e bem treinado, como o GPT-3.

Aluno Modelo com menos parâmetros, treinado para aprender a partir das previsões do professor ao invés de usar dados rotulados tradicionais. **Ou seja, aprende-se a imitar o comportamento do professor**.

Complexidade — *Model Distillation*

Como? Técnica de compressão onde **um modelo complexo** ("professor") **treina um modelo menor** ("aluno").

Training Na destilação, o modelo aluno recebe igual conjunto de dados que do modelo professor, mas em vez de ler resposta correta diretamente, apenas aproxima com as previsões deste.

Loss Envolve o uso de uma função de perda que mede a diferença entre as saídas do aluno e do professor.

Complexidade — *Model Distillation*

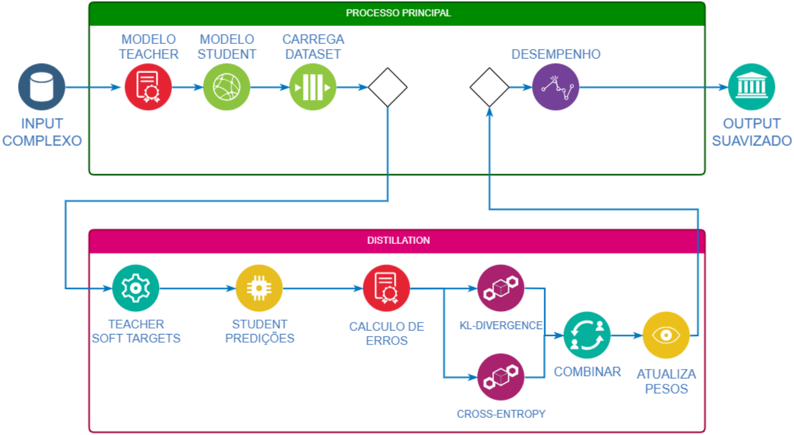


Figura: Processo de destilação (adaptado de [Polino et al., 2018]).

Sumário — Dia 1

- Introdução
- Fronteira conceitual
- Formulação do ML
- Complexidade Computacional
- Otimização

Otimização

Objetivo **Encontrar os melhores parâmetros** θ de um modelo que minimizam (ou maximizam) uma função de custo $\mathcal{L}(\theta)$.

1. **Problema central** em aprendizagem de máquina.
2. Envolve **técnicas iterativas**.
3. O método mais popular: **descida do gradiente**.

Otimização — Três estratégias de descida

BGD Descida do Gradiente **Clássica**: Calcula o gradiente usando **todo o conjunto** de dados em cada iteração.

SGD Descida do Gradiente **Estocástica**: Determina o gradiente considerando exclusivamente uma **amostra** por iteração.

MBGD Descida do Gradiente ***Mini-Batch***: Abordagem **de meio termo**, com pequenos lotes de dados em cada iteração.

Otimização — BGD

Função Os parâmetros são atualizados por

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}(\theta),$$

onde θ são os parâmetros (pesos/*bias*), η é taxa de aprendizagem, e $\mathcal{L}(\theta)$ é perda total sobre todo o *dataset*.

Pontos + Caminho mais **estável** até o mínimo e convergência **suave**.

Pontos – **Lento** com muitos dados e **alto custo** computacional.

Otimização — SGD

Função Os parâmetros são atualizados por

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}^{(i)}(\theta).$$

Perda sobre uma única amostra: $\mathcal{L}^{(i)}$.

Pontos + Atualizações **rápidas**, pode escapar de **mínimos locais**, ideal para **grandes volumes** de dados ou treinamento *online*.

Pontos − Caminho **ruidoso** com oscilações e pode **não convergir** bem sem ajustes (ex. *momentum*).

Otimização — MBGD

Função Os parâmetros são atualizados por pequenos lotes de dados[1], dado por

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}_{\text{batch}}(\theta).$$

Pontos + Mais eficiente que **BGD**, mais estável que **SGD** e facilita paralelização com **GPUs**.

Pontos − Escolha do tamanho do *batch* pode **afetar desempenho**.

[1] Tipicamente 32 a 128 amostras.

Otimização — Comparação

Critério	BGD	SGD	MBGD
Velocidade	Lento	Muito rápido	Rápido
Estabilidade	Alta	Baixa	Média/Alta
Requisitos computacionais	Altos	Baixos	Moderados
Ideal para	Dados pequenos	Dados massivos	Casos gerais

Tabela: Comparativo entre BGD, SGD e MBGD.

Sumário — Dia 2

- Otimização
- Perceptron
- Multilayer Perceptron
- Regressão logística

Otimização

Objetivo **Encontrar a melhor estimativa dos parâmetros** θ de um modelo, que minimizam, uma função de custo $\mathcal{L}(\theta)$.

1. **Problema central** em aprendizagem de máquina.
2. Envolve **técnicas iterativas**.
3. O método mais popular: **descida do gradiente**.

Otimização

Motivação Para projetar otimizadores eficientes, precisamos entender as propriedades das funções de perda. Algumas propriedades matemáticas controlam:

1. A existência de **mínimos globais**.
2. A **estabilidade numérica**.
3. A **taxa de convergência** dos métodos iterativos.

Otimização — Convexidade

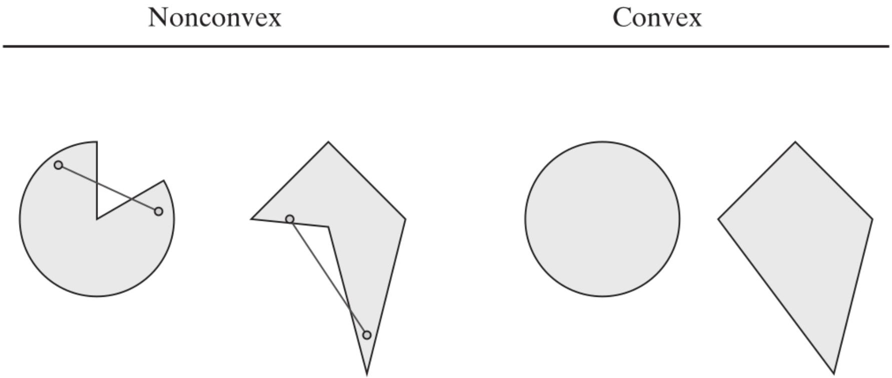


Figura: Formas convexas e não convexas [Shalev-Shwartz and Ben-David, 2014].

Seja $C \subset \mathbb{R}^d$ um conjunto convexo. Dizemos que uma função $f : C \to \mathbb{R}$ é **convexa** se, para todo $\mathbf{u}, \mathbf{v} \in C$ e $\alpha \in [0, 1]$:

$$f(\alpha\mathbf{u} + (1-\alpha)\mathbf{v}) \leq \alpha f(\mathbf{u}) + (1-\alpha) f(\mathbf{v})$$

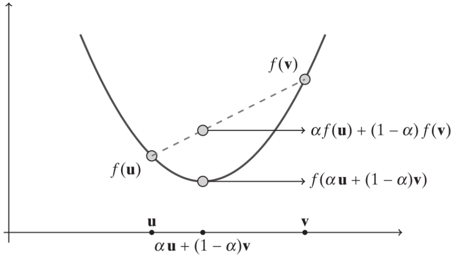


Figura: Função convexa [Shalev-Shwartz and Ben-David, 2014].

Otimização — Continuidade Lipschitz

Uma função $f : C \to \mathbb{R}$ é L-Lipschitz se:

$$|f(\mathbf{u}) - f(\mathbf{v})| \leq L\|\mathbf{u} - \mathbf{v}\|, \quad \forall \mathbf{u}, \mathbf{v} \in C$$

Intuição A função f não varia abruptamente, ou seja tem crescimento controlado.

Conseq. Limita a diferença entre saídas com entradas próximas, importante para estabilidade e generalização.

Otimização — Continuidade Lipschitz

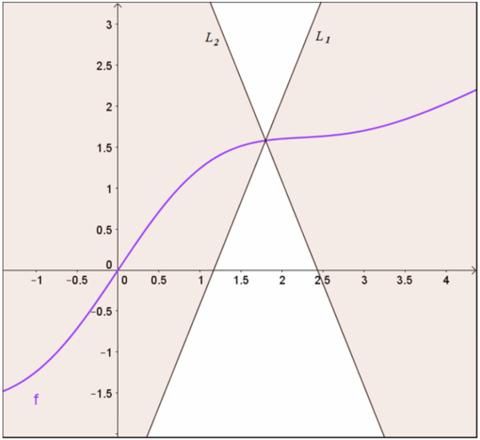


Figura: Continuidade Lipschitz [Statistics How To, 2023].

Otimização — Suavidade

Definição Uma função diferenciável $f : C \to \mathbb{R}$ é β-suave se:

$$\|\nabla f(\mathbf{u}) - \nabla f(\mathbf{v})\| \leq \beta \|\mathbf{u} - \mathbf{v}\|, \quad \forall \mathbf{u}, \mathbf{v} \in C$$

Relevância Controla a curvatura local de f e garante convergência de algoritmos como o gradiente descendente.

Otimização — Suavidade

Valor de β	**Interpretação**
$\beta = 0$	O gradiente é constante. Ex.: $f(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b \implies \nabla f(\mathbf{x}) = \mathbf{a}$; $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c \implies \nabla f(\mathbf{x}) = A\mathbf{x} + \mathbf{b}$.
$0 < \beta < \infty$	Gradiente varia de forma controlada, função suave. Ex.: $f(x) = \log(1 + e^{-x})$
$\beta \to \infty$	Gradiente pode variar abruptamente, função não suave. Ex.: ReLU, $f(x) = \max(0, x)$. Não é suave em $x = 0$, pois o gradiente muda abruptamente de 0 para 1.

Tabela: Valores admissíveis.

Otimização — Estratégias

Hipótese teórica	**Problemas reais**	**Impacto**	**Abordagens de contorno**
Convexidade global	Superfície não convexo, saddle points frequentes	Convergência apenas local	Momentum, reinícios cíclicos, métodos de segunda ordem aproximados
L-Lipschitz conhecido	Constante efetiva desconhecida ou alta	Saltos irregulares, explosão de gradiente	Clipping, normalização de camadas, restrição espectral
β-suavidade	Discontinuidades locais (ReLU, ℓ_1)	Gradiente imprevisível	Subgradientes, regularização suave, otimização proximal
Gradiente exato	Estimativa estocástica via minibatch	Variância alta e ruído	Adam, RMSProp

Tabela: Quadro comparativo.

Def. A função de perda $\mathcal{L}(\boldsymbol{\theta})$ expressa o erro agregado de um modelo parametrizado por $\boldsymbol{\theta}$ sobre um conjunto de exemplos $(\mathbf{x}_i, y_i)$.

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$$

Onde $\boldsymbol{\theta}$ são os parâmetros do modelo, $f_{\boldsymbol{\theta}}$ é a função preditora e $\ell(\cdot, \cdot)$ é a perda individual [Shalev-Shwartz and Ben-David, 2014].

Otimização — $\mathcal{L}(\boldsymbol{\theta})$ mais usadas

Forma da perda	**Quando utilizar**
$\lVert h_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i \rVert_2^2$	Regressão
$\lVert h_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i \rVert_1$	Regressão robusta a *outliers*
$\lVert h_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i \rVert_\infty$	Problemas com limite máximo de erro permitido
$-\sum_{i=1}^{n} y_i \log h_{\boldsymbol{\theta}}(\mathbf{x}_i)$	Classificação

Sumário — Dia 2

- Otimização
- Perceptron
- Multilayer Perceptron
- Regressão logística

Perceptron — Contexto histórico

Objetivo Simular o processo de aprendizagem humana e realizar tarefas de reconhecimento de padrões [Rosenblatt, 1958].

Problemas Limitações matemáticas do Perceptron de camada única. Incapacidade de resolver o problema complexos [Block, 1962].

Perceptron — Funcionamento básico

Função Classificador binário linear.

$$w_i \leftarrow w_i + \alpha(y - \hat{y})x_i$$

Entrada Recebe um vetor de características $[x_{i_1}, x_{i_2}, \ldots, x_{i_d}]$, $i = 1, 2, 3, \ldots, n$.

Pesos Cada entrada é associada a um peso $[w_1, w_2, \ldots, w_n]$.

Obs Soma ponderada e definição de viés. Função de ativação. **Aprendizado** — os pesos são atualizados com base no erro da previsão: taxa de aprendizado, rótulo verdadeiro e saída prevista.

Perceptron — Função OR/AND

Problema A saída de um *perceptron* binário é definida por:

$$y = \text{step}(w^\top x), \textbf{onde}$$

$$x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}, \quad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

$$\phi(z) = \begin{cases} 1, & \text{se } z \geq 0 \\ 0, & \text{se } z < 0 \end{cases}$$

Perceptron — Função OR/AND

Logo, o *perceptron* **depende da** separabilidade das entradas.

$$y = \phi(w_0 + w_1 x_1 + w_2 x_2)$$

Fronteira de decisão:

$$w_0 + w_1 x_1 + w_2 x_2 = 0 \quad \text{e} \quad x_2 = \frac{-w_0 - w_1 x_1}{w_2}$$

Essa eq. define uma linha no plano (x_1, x_2), separando as regiões onde o *perceptron* gera saída 0 ou 1.

Perceptron — Função OR/AND

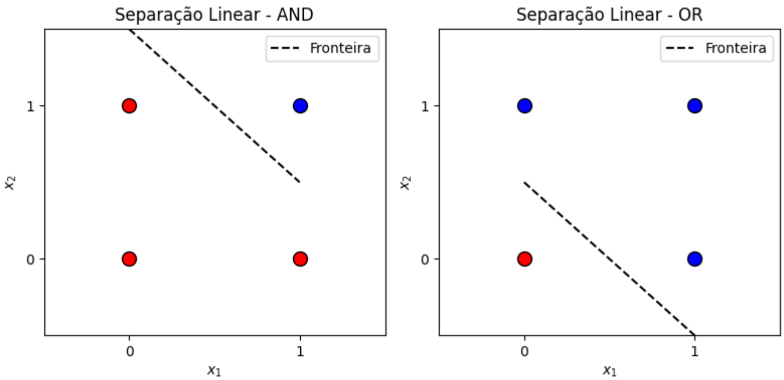


Figura: Retas de separação do **AND** e **OR**.

Perceptron — Problema do XOR

$$XOR(x_1, x_2) = \begin{cases} 1, & \text{se } x_1 \neq x_2 \\ 0, & \text{se } x_1 = x_2 \end{cases}$$

No XOR O conjunto de pontos não é linearmente separável.

Nenhuma reta consegue dividir as classes 0 e 1 no plano (x_1, x_2).

Um único neurônio (*perceptron* simples) não é suficiente.

MLP — Solução de problemas complexos

01. Uma ou mais camadas intermediárias com pelo menos dois neurônios.

02. Uma função de ativação não linear ϕ aplicada na camada oculta.

03. Um neurônio final que combina os resultados da camada intermediária.

» **Correção de erro por ajuste de pesos.**

MLP — Correção de erro

Def. Para cada amostra (x, y), com $x \in \mathbb{R}^n$ e $y \in \{0, 1\}$:

$$\hat{y} = \phi(w^\top x)$$

$$\delta = y - \hat{y}$$

$$w \leftarrow w + \eta \cdot \phi \cdot x$$

$\eta > 0$ taxa de aprendizado.

δ erro de predição.

A correção é proporcional ao erro e às entradas.

MLP — Duas camadas ocultas para XOR

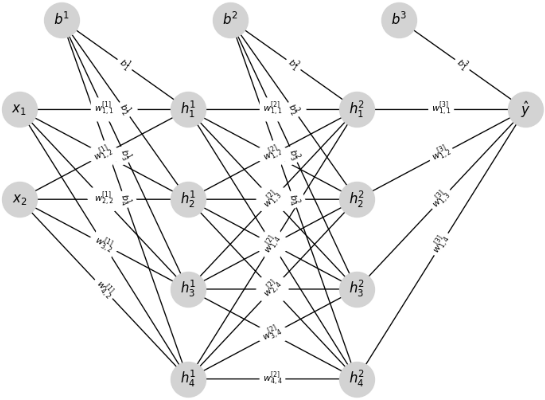


Figura: MLP com duas camadas ocultas.

MLP — Função XOR

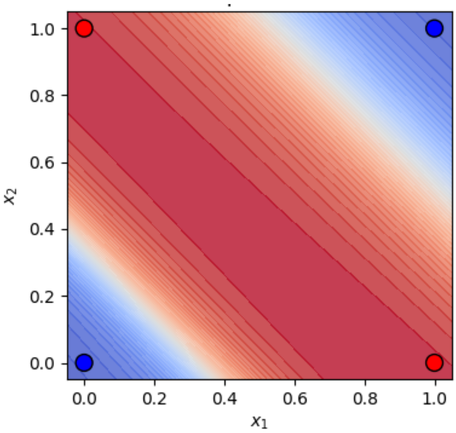


Figura: Camadas de separação do XOR.

Sumário — Dia 2

Multilayer Perceptron — Arquitetura

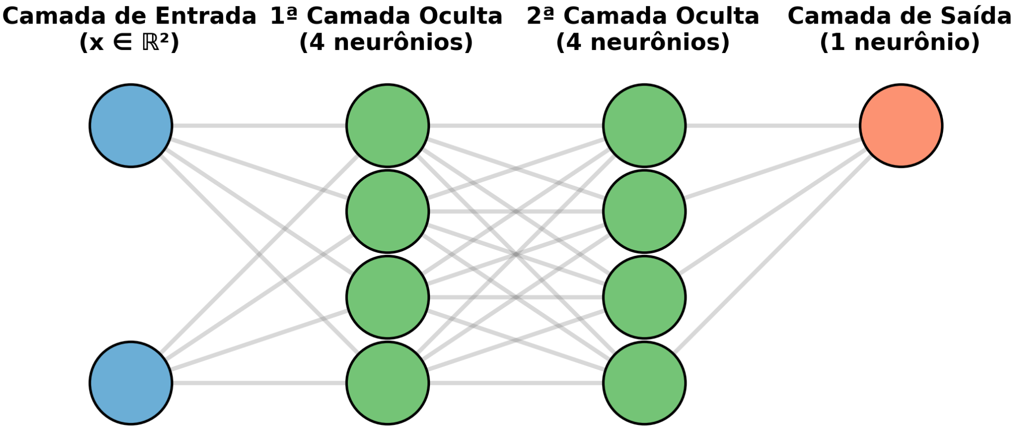


Figura: $W_1 \in \mathbb{R}^{2\times4}, b_1 \in \mathbb{R}^{1\times4}, W_2 \in \mathbb{R}^{4\times4}, b_2 \in \mathbb{R}^{1\times4}, W_3 \in \mathbb{R}^{4\times1}, b_3 \in \mathbb{R}^{1\times1}$

Multilayer Perceptron — Backpropagation

Forward pass O cálculo da saída $\hat{y}$ da rede é realizado em três etapas:

$$Z_1 = xW_1 + b_1 \qquad A_1 = \phi(Z_1)$$

$$Z_2 = A_1W_2 + b_2 \qquad A_2 = \phi(Z_2)$$

$$Z_3 = A_2W_3 + b_3 \qquad \hat{y} = \phi(Z_3)$$

onde $\phi(z) = \frac{1}{1+e^z}$ é a função sigmoide.

MSE ***Mean Squared Error***

$$\mathcal{L} = \frac{1}{2}(y - \hat{y})^2$$

» O gradiente da perda em relação à saída é dado por

$$\frac{\delta \mathcal{L}}{\delta Z_3} = (\hat{y} - y) \cdot \phi'(Z_3)$$

Backward pass o erro é propagado da saída para as camadas ocultas.

Camada **3** (saída)

$$\delta_3 = (\hat{y} - y) \cdot \phi'(Z_3) \qquad \nabla_W 3 = A_1^\top \delta_3 \qquad \nabla_{b_3} = \delta_3$$

Camada **2**

$$\delta_2 = (\delta_3 W_3) \cdot \phi'(Z_2) \qquad \nabla_W 2 = A_1^\top \delta_2 \qquad \nabla_{b_2} = \delta_2$$

Camada **1**

$$\delta_1 = (\delta_2 W_2) \cdot \phi'(Z_1) \qquad \nabla_W 1 = x^\top \delta_1 \qquad \nabla_{b_1} = \delta_1$$

Atualização dos pesos

$$W_i \leftarrow W_i - \eta \nabla_{W_i} \qquad b_i \leftarrow b_i - \eta \nabla_{b_i}$$

para $i = 1, 2, 3$, onde η é a taxa de aprendizado.

Esse processo é repetido para cada época até que o erro da rede seja suficientemente pequeno ou a acurácia atinja o valor desejado.

Sumário — Dia 2

- Otimização
- Perceptron
- Multilayer Perceptron
- Regressão logística

Regressão Logística

Introd. Em algumas situações a variável passa a ter o comportamento:

Qualitativo » Categórica » Classificação

Muitas técnicas de classificação estimam a probabilidade.

Exemplo: Regressão Logística.

Pode Transcender a Lógica Binária

- Uma pessoa chega ao pronto socorro com sintomas que podem ser atribuídos a três condições médicas.
- Qual das três condições o indivíduo possui?

Sintomas Variáveis explicativas.

- Dar as variáveis explicativas uma das três condições.

Pressupostos

1. A variável resposta precisa ser qualitativa (dicotômica ou binária).
2. As varáveis preditoras podem ser quantitativas ou categóricas.
3. As observações são independentes (uma não afeta a outra).
4. A principal ideia da Regressão Logística é de estimar uma probabilidade.

Regressão Logística *vs.* Regressão Linear

Linear Qual o valor de uma casa, dadas suas **características**?

Logística O cliente vai pagar uma dívida? Sim $(= 1)$ ou não $(= 0)$.

A premissa da Regressão linear é que exista uma relação linear entre a variável resposta e as variáveis explicativas.

Para o caso de variáveis binárias essa premissa será **violada**.

Os valores projetados por um modelo de **regressão linear podem ser superiores a 1 ou inferiores a 0**.

Regressão Logística *vs.* Regressão Linear

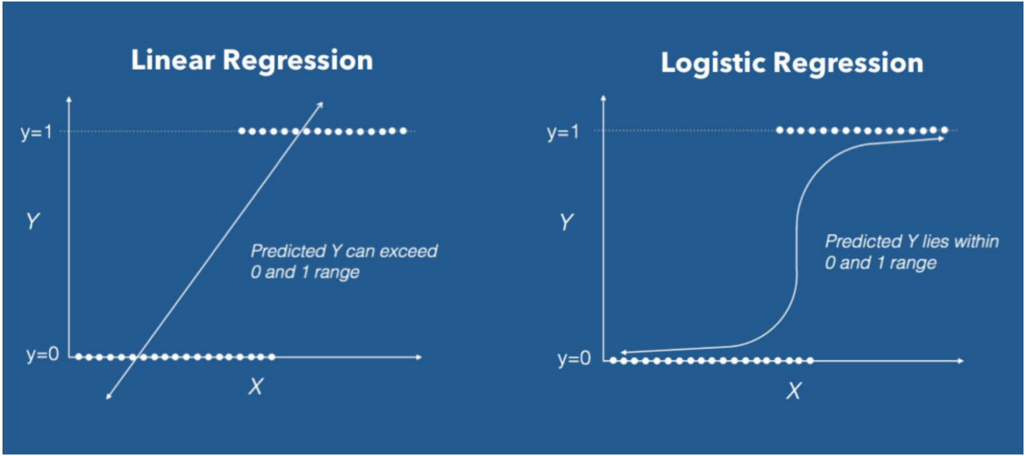


Figura: Regressão Linear *vs.* Regressão Logística [1]

Regressão Logística — Teoria

R. Linear $\hat{y} = wx + b$, onde

- w coeficiente angular.
- x variável preditora.
- b constante.

Restrição $g()$: $\hat{y} = g(wx + b)$

Calcula-se a probabilidade de se ter um resultado positivo (y) dividido pela probabilidade de se ter um resultado negativo $(1 - y)$, tratando-se da "Relação das Probabilidades".

$$\frac{y}{1 - y} \rightarrow (y : 1 - y)$$

Regressão Logística — Teoria

1. A Relação das Probabilidades é a probabilidade de positivo (1) dividido pela probabilidade de negativo (0).

   Podemos encontrar a equação final da Função Logística.

   Função linear dos preditos (*logodds*) | (Sigmoide).

$$\log\left(\frac{\hat{y}}{1-\hat{y}}\right) = wx + b \rightarrow \hat{y} = \frac{1}{1+e^{-(wx+b)}}$$

Regressão Logística — Teoria

2. Para transformarmos na **Regressão Logística**, pegamos a função linear e encaixamos no denominador.

   A equação é 1 dividido por 1 mais e elevado a menos a função linear $wx + b$.

   A grosso modo, esprememos uma Regressão Linear em uma Regressão Logística.

Questão Como, matematicamente, a fórmula consegue manter os números entre zero e um? Seja o resultado intermediário da Regressão Linear $= z$. Logo, podemos dizer que $z = wx + b$.

$$\hat{y} = \frac{1}{1 + e^{-z}}$$

Suponhamos $z = 20$.

$$\hat{y} = \frac{1}{1 + e^{-20}}$$

Regressão Logística — Aplicação

- Sabemos que, independente do valor da constante e, quando elevado a -20 terá um valor desprezável. Assim,

$$\hat{y} = \frac{1}{1 + e^{-20}} \rightarrow \frac{1}{1 + 0}$$

$$\hat{y} = \frac{1}{1} = 1$$

Se z é muito pequeno, por exemplo $z = -200$, a equação irá agir de maneira oposta ($\hat{y}$ tende a 0).

Detecção de *SPAM*

- A saída representa a probabilidade de que $y = 1$ dado uma entrada x.
- Exemplo: se $\hat{y} = 0{,}7$, há 70% de chance de ser *spam*.
- Utilizamos um **limiar** para tomada de decisão. Acima do limiar, classifica-se como **positivo**; abaixo, **negativo**.
- O padrão é $0,5$, mas pode ser ajustado (por exemplo, $0,9$ para ser conservador; $0,3$ para ser agressivo).

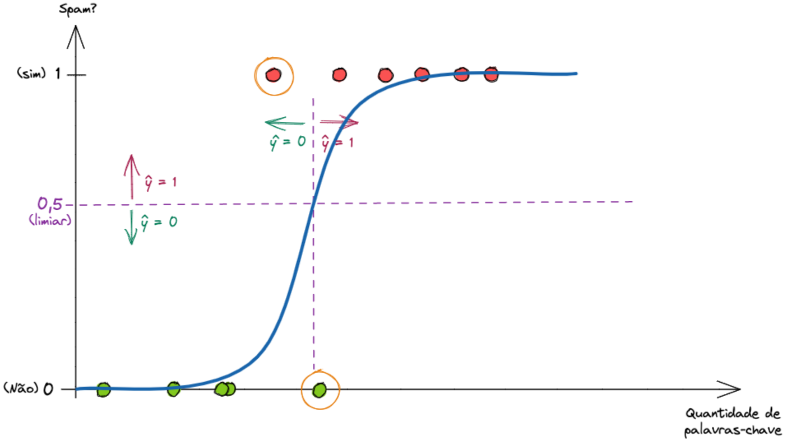


Figura: Limiar de $0,5$ [Fonte: Brains].

Regressão Logística — Influência do limiar

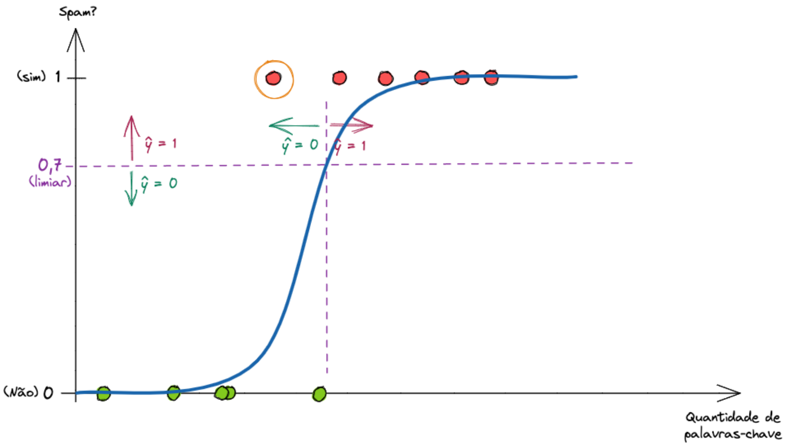


Figura: Limiar de $0,7$ [Fonte: Brains].

Regressão Logística — Influência do limiar

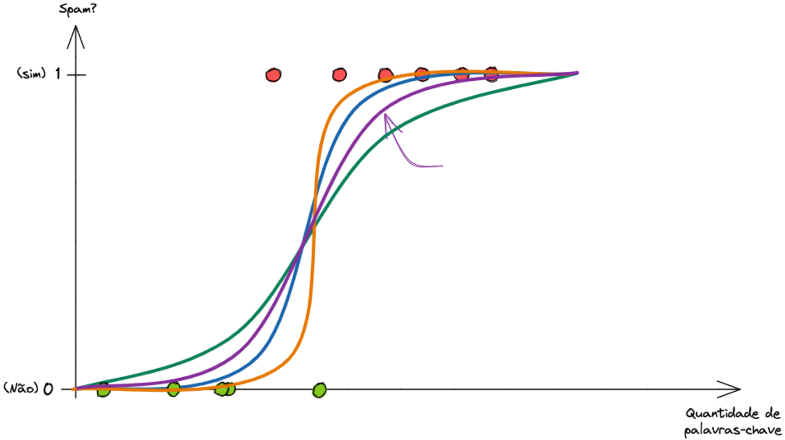


Figura: Qual o caption? [Fonte: Brains].

Log Loss Logarítmo de perda. A equação se ajusta de acordo com o valor real de y.

Interpretação condicional:

$$Log\ Loss = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$$

Se $y = 0$: O erro será maior quanto maior for a probabilidade prevista $\hat{y}$

$$Log\ Loss = -\log(1-\hat{y})$$

Log Loss Logarítmo de perda.

Interpretação condicional:

$$Log\ Loss = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$$

Se $y = 1$: o segundo termo será cancelado. Iremos multiplicar $\log(1-\hat{y})$ por $(1-1)$, ou seja, por 0. Neste caso, a equação fica.

$$Log\ Loss = -\log(\hat{y})$$

Sumário — Dia 3

Contexto — Objetivo do Estudo

Avaliar O *Multilayer Perceptron*.

A Regressão Logística (visão do *Machine Learning*).

A Regressão Logística (visão da Estatística).

Aplicados À predição de **risco de *diabetes mellitus*** com base em dados clínicos simples[1].

[1] https://github.com/pcbrom/perceptron-mlp-cnn

Contexto — Base de dados

Dados 768 observações e 9 variáveis clínicas do repositório UCI.

Problemas Colunas zeradas em variáveis que não deveriam ser nulas.

Colunas `Glucose` $(n = 5)$, `BloodPressure` $(n = 35)$, `SkinThickness` $(n = 227)$, `Insulin` $(n = 374)$, `BMI` $(n = 11)$.

Ação Remoção de observações com inconsistências. O conjunto resultante teve 392 observações válidas.

Objetivo Garantir integridade estatística sem imputação *ad hoc*.

Análise da densidade

- Todas as variáveis apresentaram distribuições significativamente diferentes entre os grupos $p < 0{,}001$: teste de Mann–Whitney, usando a correção de Benjamini-Hochberg.
- As variáveis `Glucose`, `Insulin` e `BMI` mostraram maior separação visual e deslocamento nas medianas.

Contexto — Exploração inicial

Análise da correlação e multicolinearidade

- Correlação alta entre `Age` e `Pregnancies` ($r = 0{,}68$).
- Correlação moderada entre `BMI` e `SkinThickness` ($r = 0{,}66$).
- Correlação esperada entre `Glucose` e `Insulin` ($r = 0{,}58$).
- Estrutura de *cluster* identificada por análise hierárquica de correlações.

Contexto — Exploração inicial

Análise das implicações

- Exclusão de variáveis redundantes ou instáveis em relação à escala.
- Foco em preditores com boa separação e menor colinearidade.

Contexto — Exploração inicial

Análise da Linearidade (teste de Box–Tidwell)

Objetivo Verificar a adequação da escala original dos preditores ao modelo logístico.

Resultados Evidência de não linearidade forte ou extrema em `BMI`, `Insulin`, `SkinThickness`, e `BloodPressure`.

`Glucose` apresentou padrão sublinear ($\lambda \approx 0,70$), mas ainda interpretável sem transformação.

Contexto — Exploração inicial

Análise **da Linearidade (teste de Box–Tidwell)**

Decisões Variáveis com distorção severa foram descartadas (ex. `BMI` não transformado).

Foi mantida a escala original para `Glucose`, `BMI`, `Age` e `Pregnancies`:

- Facilita a interpretação clínica e decisões baseadas em pontos de corte.
- Permite uso direto em contextos operacionais (triagem, protocolos de enfermagem).

Contexto — Preditores mantidos

Glucose

- Forte capacidade discriminatória (AUC $= 0{,}806$), amplamente validado.
- Coleta padronizada e acessível no SUS (R$ 1,85).
- Já utilizado em protocolos clínicos.

BMI

- VIF alto, mas custo marginal e interpretabilidade imediata.
- Forte associação com risco metabólico.
- Recomendado por diretrizes nutricionais.

Contexto — Preditores mantidos

Age (idade)

- Preditor estável, colinearidade baixa e de fácil obtenção.
- Captura progressão basal do risco metabólico com o envelhecimento.

Pregnancies (número de gestações)

- Específico para população feminina.
- AUC $= 0{,}651$, efeito moderado.
- Correlação com idade reprodutiva.
- Relevante para triagem em saúde pública.

Contexto — Generalização e dimensão VC

Dimensão VC estimada

- Regressão Logística (ML): VC $\approx d + 1 = 4$.
- Multilayer Perceptron: VC $\approx O(W \log W) \approx 60$.

Referência do curso

- Note que, no **Dia 1**, para VC $= 3$, $\varepsilon = 0{,}1$, e $\delta = 0{,}05$, são necessários $n = 600$ para garantir 95% de cobertura e erro de 10%.
- Nosso conjunto está abaixo desse limite: $n = 392$.

Contexto — Generalização e dimensão VC

Implicação prática

- Modelos com alta complexidade (ex.: MLP) exigem controle adicional (regularização, validação cruzada).
- A Regressão Logística Estatística oferece maior garantia de generalização, dada a limitação amostral.

Contexto — Propriedades da otimização

Propriedade	MLP	Reg. Log. (ML)	Reg. Log. (Estat.)
Convexidade	Não convexa; risco de mínimos locais. Requer boas estratégias de inicialização e otimização.	Convexa no espaço dos parâmetros, se sem regularização não complexa.	Convexa com solução global garantida. Estável para pequena dimensão VC.
Lipschitz	Nem sempre garantido (ex.: ReLU). Sigmoide melhora a continuidade.	Pode ser garantido com regularização e escolha de penalização (ex: Ridge).	Garantido devido à forma analítica e escala controlada dos dados.
β**-suavidade**	Depende da função de ativação e arquitetura. Possui regiões de alta curvatura.	Suavidade moderada com penalizações. Aprendizado mais controlado.	Suave. Gradientes estáveis, mesmo sem regularização.

Sumário — Dia 3

» Contexto do Projeto

» Multilayer Perceptron

» Modelo Regressão Logística (ML)

» Modelo Regressão Logística (Estatística)

MLP Multilayer Perceptron

- Arquitetura com camadas ocultas e ativação não linear.
- Capaz de modelar relações complexas e interações.
- Otimizado por retro propagação e descida do gradiente.

MLP — Variáveis de entrada

Objetivo Treinar uma MLP para prever a presença de diabetes a partir de atributos fisiológicos e laboratoriais.

Entradas

Pregnancies	Número de gravidez	Discreta
Glucose	Teste oral de duas horas	Contínua
BloodPressure	Pressão arterial $(mmHg)$	Contínua
SkinThickness	Espessura dobra do tríceps (mm)	Contínua
Insulin	Insulina sérica 2 horas $(\mu U/mL)$	Contínua
BMI	Índice de Massa Corporal (kg/m^2)	Contínua
DiabetesPedigreeFunction	Histórico familiar de diabetes	Contínua
Age	Idade (anos)	Discreta

MLP — Arquitetura

Objetivo Treinar uma MLP para prever a presença de diabetes a partir de atributos fisiológicos e laboratoriais.

Arquitet. **8** variáveis de **entrada** (padronizadas).

1ª camada oculta » 6 neurônios com ativação $\phi(z)$.

2ª camada oculta » 3 neurônios com ativação $\phi(z)$.

Saída » 2 neurônios com ativação Softmax.

($0 =$ não diabético, $1 =$ diabético).

MLP — Arquitetura

Ativação camada oculta

» ReLU

$$\phi(z) = \max(0, z)$$

Ativação camada saída

» Softmax

$$\text{softmax}(z_j) = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

Função de custo

» *Cross-entropy* categórica

$$\mathcal{L}(y, \hat{y}) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{2} y_{ij} \log(\hat{y}_{ij})$$

Otimização

» Usou-se o algoritmo de descida do gradiente clássico (*batch*), calculados por *backpropagation* e atualizados com base no erro de todo o conjunto de treino, usando uma taxa de aprendizado $\eta = 0{,}0001$ em 1.500 épocas.

MLP — Arquitetura

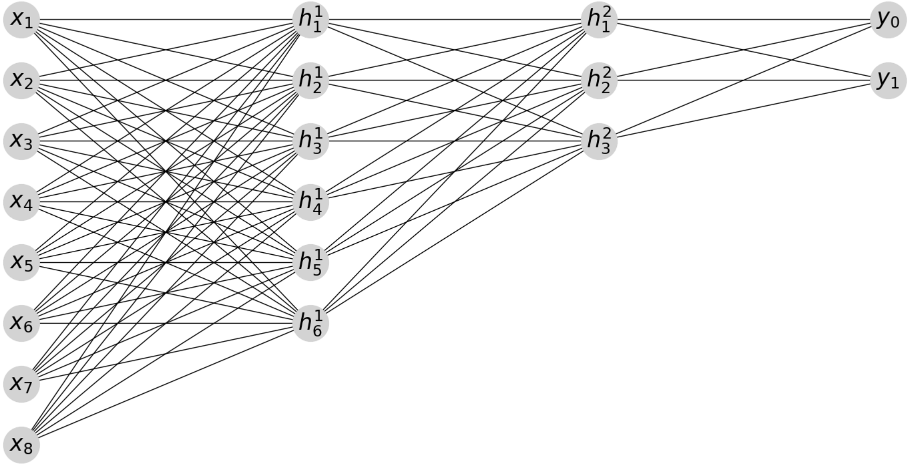
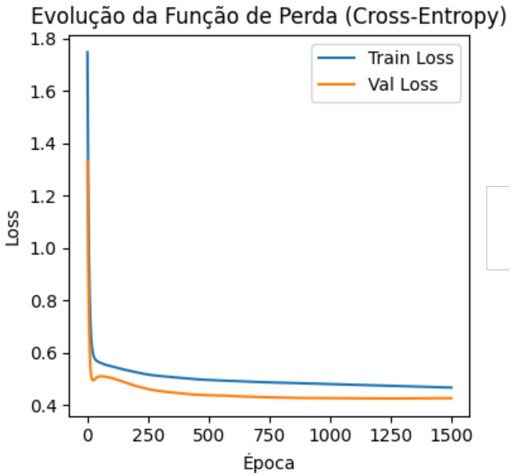


Figura: Entrada (8) → Oculta1 (6, ReLU) → Oculta2 (3, ReLU) → Saída (2, Softmax).

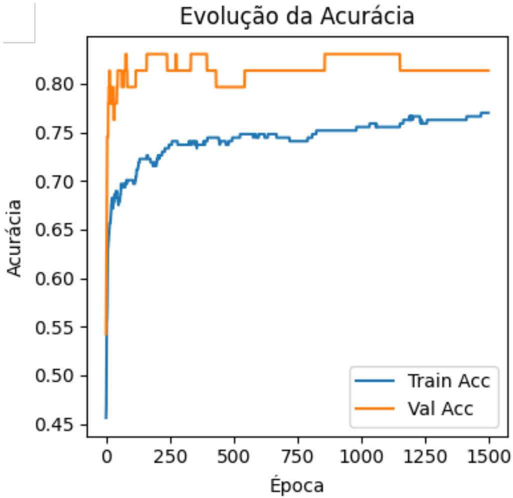
MLP — Resultados



Função de Perda

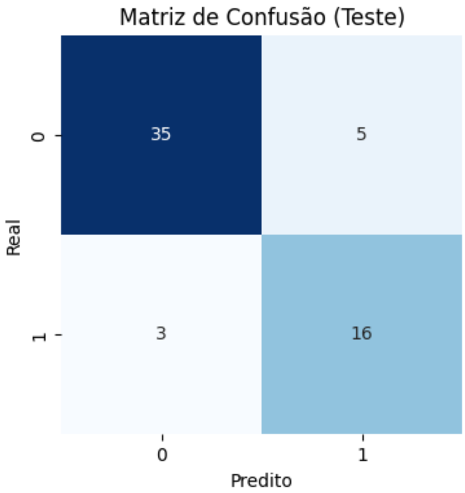
- A perda de treino (azul) e de validação (laranja) diminuem ao longo do tempo.
- A curva estabiliza após cerca de 500 épocas.
- A perda de validação é ligeiramente menor, indicando boa generalização.

MLP — Resultados



Acurácia

- A acurácia de treino cresce de forma contínua.
- A acurácia de validação atinge rapidamente cerca de 80% e se mantém estável.
- A diferença entre as curvas indica bom desempenho sem sinais de *overfitting*.



Matriz de confusão

- Boa capacidade de acerto, especialmente para a classe 0 (não diabetes).
- Há poucos falsos positivos (5) e falsos negativos (3).
- O modelo está equilibrado e apresenta bom desempenho em ambas as classes.

MLP — Métricas (classe 1)

Acurácia

$$\frac{TP+TN}{TP+TN+FP+FN} \approx \mathbf{0{,}864}$$

Revocação (*Recall*)

$$\frac{TP}{TP+FN} \approx \mathbf{0{,}842}$$

Precisão

$$\frac{TP}{TP+FP} \approx \mathbf{0{,}762}$$

F1-Score

$$2 \cdot \frac{\text{Precisão}\times\text{Recall}}{\text{Precisão}+\text{Recall}} \approx \mathbf{0{,}800}$$

MLP — Avaliação do modelo

» O modelo apresentou bom equilíbrio entre sensibilidade e precisão, especialmente para a classe 0.

Classe	**Precisão**	**Revocação**	**F1-score**	**Suporte**
1	0,7619	0,8421	0,8000	19
0	0,9211	0,8750	0,8974	40
Acurácia			**0,8644**	59
Média macro	0,8415	0,8586	0,8487	59
Média ponderada	0,8698	0,8644	0,8661	59

Tabela: Relatório de Classificação.

MLP — Complexidade computacional

Algoritmo do *backpropagation*

- Eficiente, mas exige custo quadrático ou pior no número de conexões. **O custo de cada época** de treinamento é dado por

$$\mathcal{O}(n \cdot d \cdot L),$$

onde: $n =$ exemplos, $d =$ é entradas e $L =$ camadas.

- Logo, o treinamento depende de **GPUs/TPUs**, **paralelismo** e **distribuição** para ser viável.

MLP — Complexidade computacional

Dados **Exige muitos dados** para generalizar bem:

Alta capacidade (dimensão VC) $\to$ risco de *overfitting* se n pequeno.

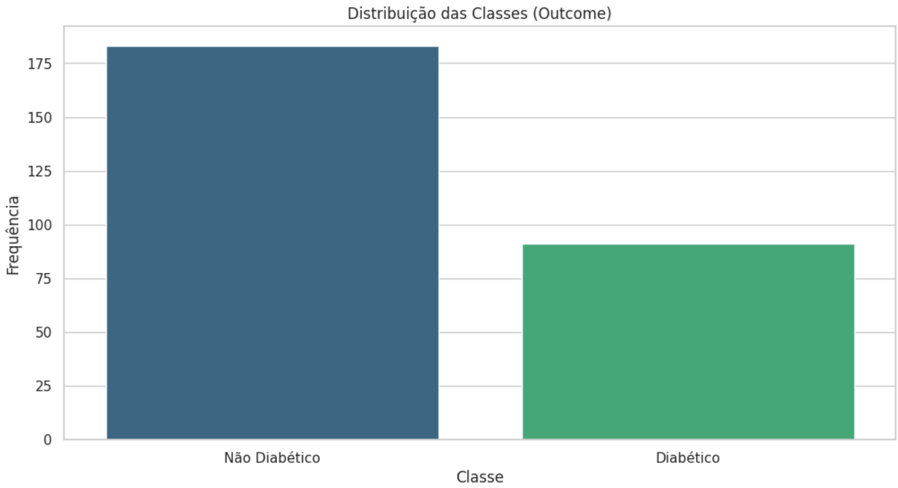
Dados devem ser **representativos** e **preprocessados**.

Memória Armazenamento de pesos, gradientes e ativações podem exigir **muita memória**.

Sumário — Dia 3

- Contexto do Projeto
- Multilayer Perceptron
- Modelo Regressão Logística (ML)
- Modelo Regressão Logística (Estatística)

Regressão Logística — ML



» Glucose, Age, BMI são as características com maior correlação com o desfecho.

```
Correlação ponto-biserial entre as variáveis contínuas e a variável binária 'Outcome':
```

	Variável	Correlação_Ponto_Biserial
1	Glucose	0.466036
7	Age	0.353429
5	BMI	0.295597
3	SkinThickness	0.262339
4	Insulin	0.261645
0	Pregnancies	0.228685
6	DiabetesPedigreeFunction	0.194173
2	BloodPressure	0.111044

Regressão Logística — ML

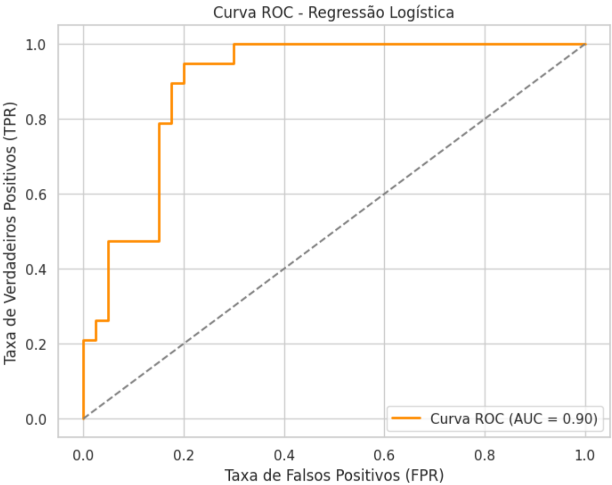
	Limiar	Precisão	Recall	F1-Score
0	0.10	0.413043	1.000000	0.584615
1	0.15	0.441860	1.000000	0.612903
2	0.20	0.487179	1.000000	0.655172
3	0.25	0.513514	1.000000	0.678571
4	0.30	0.558824	1.000000	0.716981
5	0.35	0.612903	1.000000	0.760000
6	0.40	0.666667	0.947368	0.782609
7	0.45	0.680000	0.894737	0.772727
8	0.50	0.666667	0.736842	0.700000
9	0.55	0.666667	0.631579	0.648649
10	0.60	0.647059	0.578947	0.611111
11	0.65	0.666667	0.526316	0.588235
12	0.70	0.818182	0.473684	0.600000
13	0.75	0.800000	0.421053	0.551724
14	0.80	0.777778	0.368421	0.500000
15	0.85	0.833333	0.263158	0.400000
16	0.90	1.000000	0.210526	0.347826

- TPR (Sensibilidade ou Recall): $\dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$
- FPR (Especificidade invertida): $\dfrac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$
- Para cada limiar t, calcula-se o par $(\mathrm{FPR}(t), \mathrm{TPR}(t))$.
- Esses pontos são plotados na Curva ROC:

  Eixo X: FPR | Eixo Y: TPR

  AUC: Mede a capacidade discriminativa global do modelo.

Regressão Logística — ML



Sumário — Dia 3

- Contexto do Projeto
- Multilayer Perceptron
- Modelo Regressão Logística (ML)
- Modelo Regressão Logística (Estatística)

RL **Regressão Logística** (estatística clássica)

- Ênfase em inferência estatística.
- Hipóteses explícitas e coeficientes interpretáveis.
- Adequada para modelos explicativos e comunicáveis.

RL — Modelagem 1 (clássica)

Especificação

$$\text{logito}(P(\text{diabetes})) = \beta_0 + \beta_1 \cdot \texttt{Glucose} + \beta_2 \cdot \texttt{BMI} + \beta_3 \cdot \texttt{Age}$$

Resultados principais

- Todos os coeficientes com $p < 0{,}001$ (alta significância estatística).
- Pseudo-R^2 de McFadden $= 0{,}289$.
- $AIC = 362, 37$ (modelo parcimonioso e estável).

RL — Modelagem 1 (clássica)

Nota `Pregnancies` foi testado, mas excluído por não significância ($p > 0{,}05$).

Escala **Justificativa da original**

- Alinhamento com protocolos clínicos baseados em cortes absolutos.
- Facilita treinamento de equipes de enfermagem e aplicação prática em UBS.

RL — Recalibração do Modelo

Motivação

- Teste de Hosmer–Lemeshow indicou viés sistemático na calibração inicial.

Estratégia

- Ajuste de intercepto (γ_0) e inclinação (γ_1) sobre o logito original.
- Estimado por novo modelo: `Outcome` $\sim$ `logit_orig`.

RL — Recalibração do Modelo

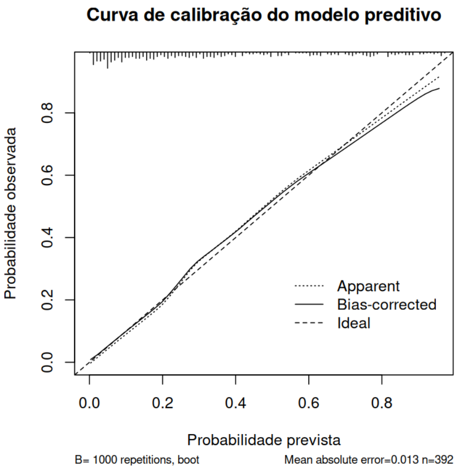
Vantagens

- Corrige o desvio médio sem modificar a estrutura do modelo.
- AUC permanece constante (ranking preservado).
- Consome apenas 2 graus de liberdade adicionais.

Resultados

- Erro absoluto médio pós-calibração: MAE $= 0{,}013$.
- Curva de calibração próxima da ideal (linha 45°).

RL — Recalibração do Modelo



Objetivo Investigar possíveis efeitos não lineares em `Glucose` e `BMI`.

Formulação `rcs(Glucose, 4)` $+$ `rcs(BMI, 4)` $+$ `Age`.
Ajuste via função `lrm()` do pacote `rms`.

Validação Calibração por bootstrap ($B = 1000$ amostras).
Erro absoluto médio ($MAE = 0{,}013$).
Curva de calibração muito próxima da ideal (45^o).

Avaliação dos termos não lineares (Wald)

- $\chi^2 = 5{,}67$, $df = 4$, $p = 0{,}225$.
- Glucose: $p = 0{,}864$ (sem efeito não linear).
- BMI: $p = 0{,}069$ (marginal, não significativo).

Implicações

- Não há evidência estatística de ganho relevante ao flexibilizar a forma funcional.
- O modelo linear permanece parcimonioso e clinicamente interpretável.

RL — Coeficientes e interpretação clínica

Estimativas do modelo logístico linear

Intercepto	$-9{,}68$
`Glucose`	$+0{,}036$
`BMI`	$+0{,}078$
`Age`	$+0{,}054$

RL — Interpretação (indivíduo mediano)

Perfil	**Glicose** (mg/dL)	**IMC** (kg/m^2)	**Idade** (anos)	**Odds Ratio**	**Probabilidade**
Geral	119	33,2	27	0,2597	0,2061
Grupo sem diabetes	107,25	31,22	25	0,1323	0,1169
Grupo com diabetes	144,5	34,6	33	1,0028	0,5007

RL — Coeficientes e interpretação clínica

Aplicabilidade

- Coeficientes em unidades clínicas facilitam integração em protocolos e triagem populacional.
- Modelo robusto, interpretável e calibrado.
- **E a capacidade preditiva?**

Desempenho por partição

Partição	Acurácia	AUC	Sensibilidade	Especificidade
Treinamento	0,79	0,83	0,90	0,57
Teste	0,80	0,89	0,90	0,58
Validação	0,83	0,86	0,87	0,75

RL — Capacidade preditiva

Erro tipo I (falso positivo)

- Especificidade entre 58 e 75% $\rightarrow$ de 25 a 42% dos indivíduos sem diabetes são classificados como positivos.

Implicação

- Sobreencaminhamentos com custo operacional, mas sem prejuízo clínico direto.

Erro tipo II (falso negativo)

- Sensibilidade $\geq 87\% \rightarrow$ apenas 10 a 13% dos diabéticos deixam de ser detectados.

Implicação

- risco de atraso diagnóstico, com impacto clínico relevante.

Complexidade de Algoritmos Supervisionados

Machine Learning

Alexandro de Paula   Charles Santos   George Fabrício   Pedro Brom

Dúvidas? Obrigados!

Referencial bibliográfico

Block, H.-D. (1962).
The perceptron: A model for brain functioning. i.
Reviews of Modern Physics, 34(1):123.

Breiman, L. (2001).
Statistical modeling: The two cultures.
Statistical Science, 16(3):199–231.

Fradkov, A. L. (2020).
Early history of machine learning.
IFAC PapersOnLine.

Goodfellow, I., Bengio, Y., and Courville, A. (2016).
Deep Learning.
MIT Press.

Hastie, T., Tibshirani, R., and Friedman, J. (2009).
The Elements of Statistical Learning: Data Mining, Inference, and Prediction.
Springer Science & Business Media.

Mullapudi, R. T., Chen, S., Zhang, K., Ramanan, D., and Fatahalian, K. (2019).
Online model distillation for efficient video inference.
In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 3573–3582.

Pearl, J. (2009).
Causality: Models, Reasoning and Inference.
Cambridge University Press.

Polino, A., Pascanu, R., and Alistarh, D. (2018).
Model compression via distillation and quantization.
arXiv preprint arXiv:1802.05668.

Robert, C. P. and Casella, G. (2004).
Monte Carlo Statistical Methods.
Springer Science & Business Media.

Rosenblatt, F. (1958).
The perceptron: A probabilistic model for information storage and organization in the brain.
Psychological Review, 65(6):386.

Shalev-Shwartz, S. and Ben-David, S. (2014).
Understanding Machine Learning: From Theory to Algorithms.
Cambridge University Press.

Statistics How To (2023).
Lipschitz function: Definition, examples.
Acesso em: 7 jun. 2025.

Strubell, E., Ganesh, A., and McCallum, A. (2020).
Energy and policy considerations for deep learning in nlp.
Psychological Review, 34(9):13693–13696.

Wainwright, M. J. (2019).
High-Dimensional Statistics: A Non-Asymptotic Viewpoint.
Cambridge University Press.