Aula 03

Gradiente Descendente e Otimização de Funções

Agenda

01

Recapitulação:
Função de Custo

02

Minimização da
Função de Custo

03

Gradiente
Descendente

04

Equação Normal

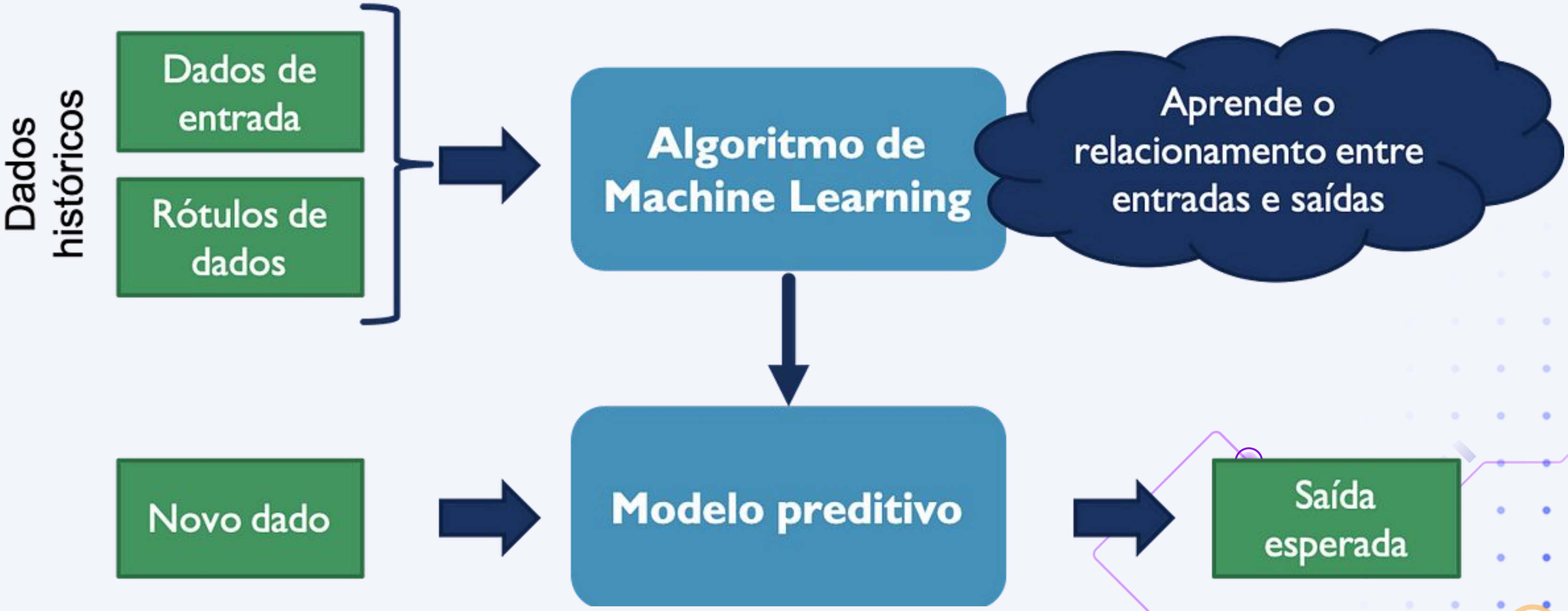
05

Implementando
os algoritmos

01

Recapitulação: Função de Custo

Modelando um fenômeno



Retirado de: https://tatianaesc.medium.com/machine-learning-conceitos-e-modelos-f0373bf4f445

Quão bom é o nosso modelo?

Como esse ajuste é calculado?

Podemos metrificar o quão bom o nosso modelo é?

Qual é o **erro** dele?

O erro do modelo é uma espécie de diferença entre o chute do modelo e a resposta correta.

Quão bom é o nosso modelo?

Como esse ajuste é calculado?

Vamos pensar no nosso erro como **um custo** ou **uma perda**.

Queremos minimizar o custo!

Objetivo: minimizar uma função que representa o custo do nosso modelo.

O que é uma Regressão Linear?

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

Uma função F é linear se:

$F(x + y) = F(x) + F(y)$ e $c^*F(x) = F(c^*x)$

Pode ser representada por uma reta, um plano, . . .

Qual a Função de Custo?

A nossa métrica de erro será a *Mean Squared Error*:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

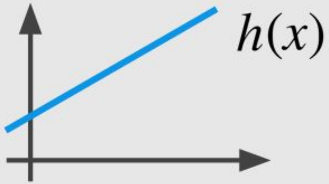
Função de Custo

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Parameters:

$$\theta_0 , \theta_1$$



Cost Function:

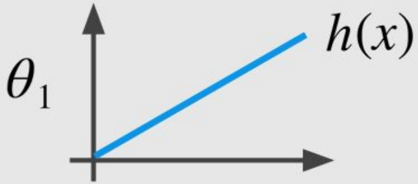
$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Goal:

$$\underset{\theta_0,\theta_1}{\text{minimize}}\ J(\theta_0,\theta_1)$$

<u>Simplified</u>

$$h_\theta(x) = \theta_1 x$$



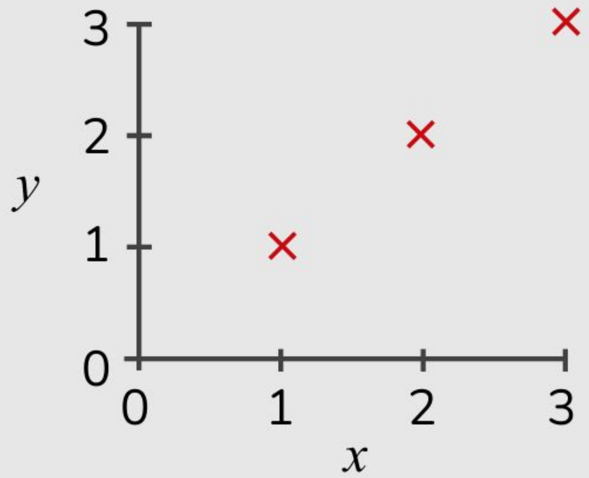
$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$$\underset{\theta_1}{\text{minimize}}\ J(\theta_1)$$

Função de Custo

$h_\theta(x)$

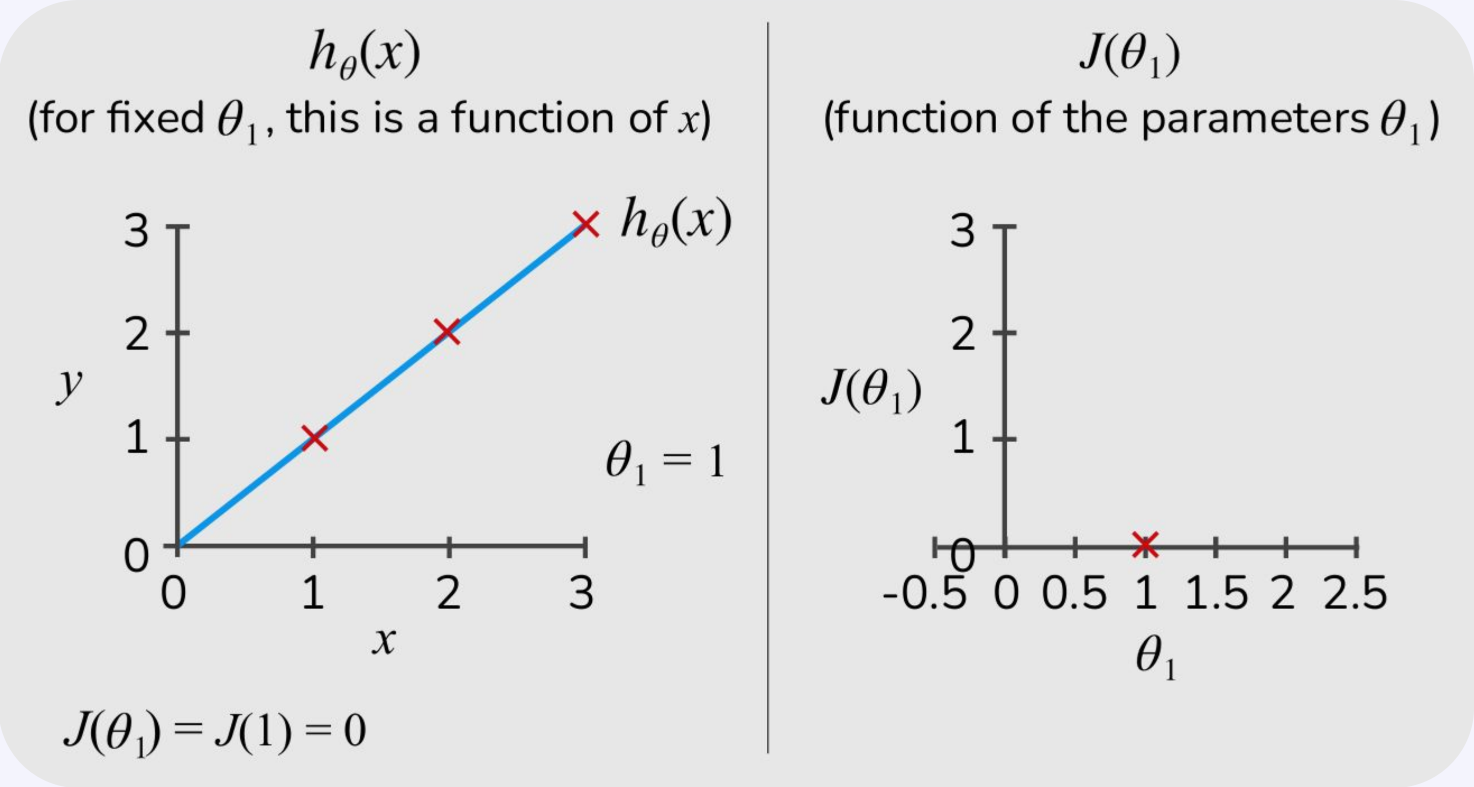
(for fixed θ_1, this is a function of x)



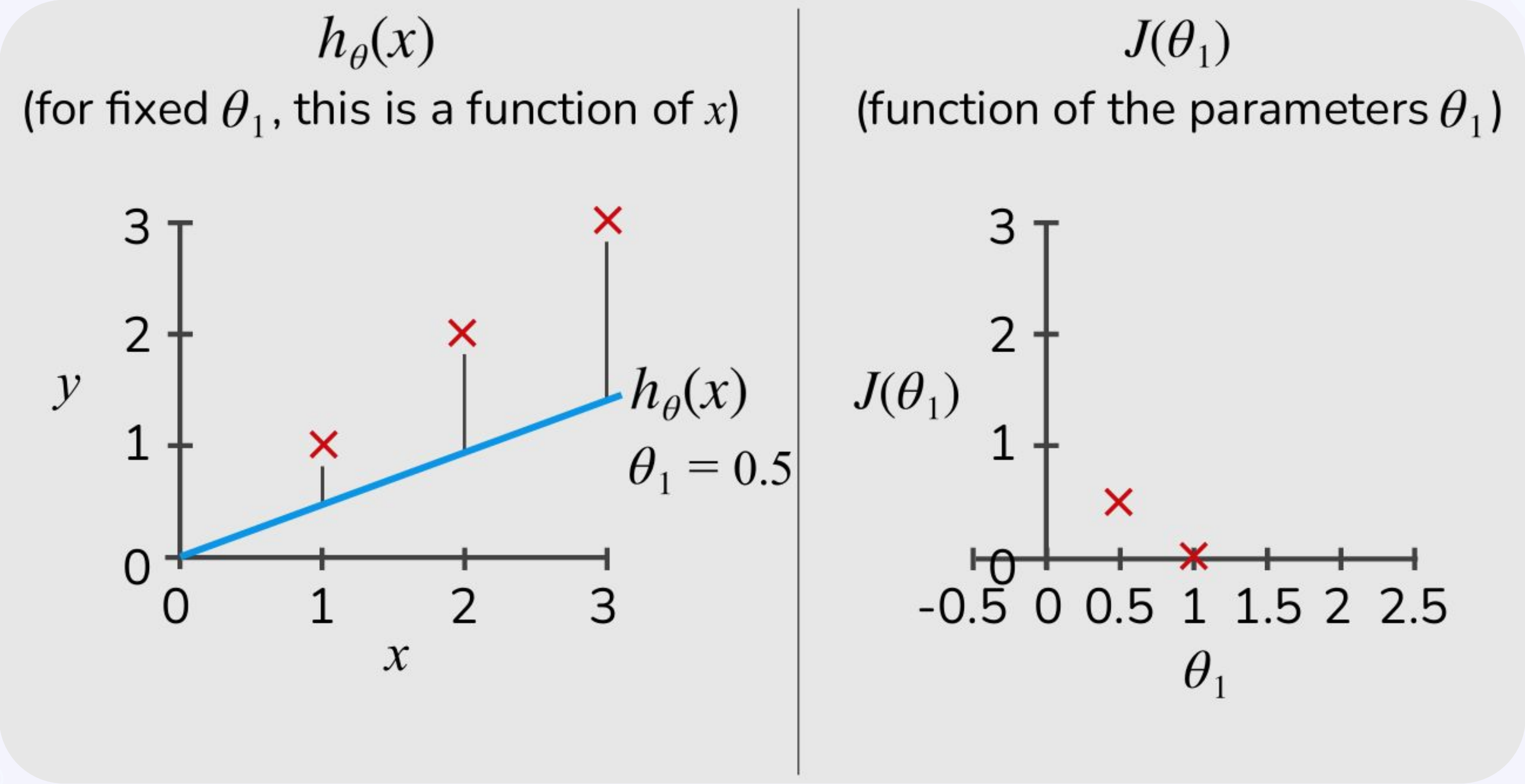
$J(\theta_1)$

(function of the parameters θ_1)

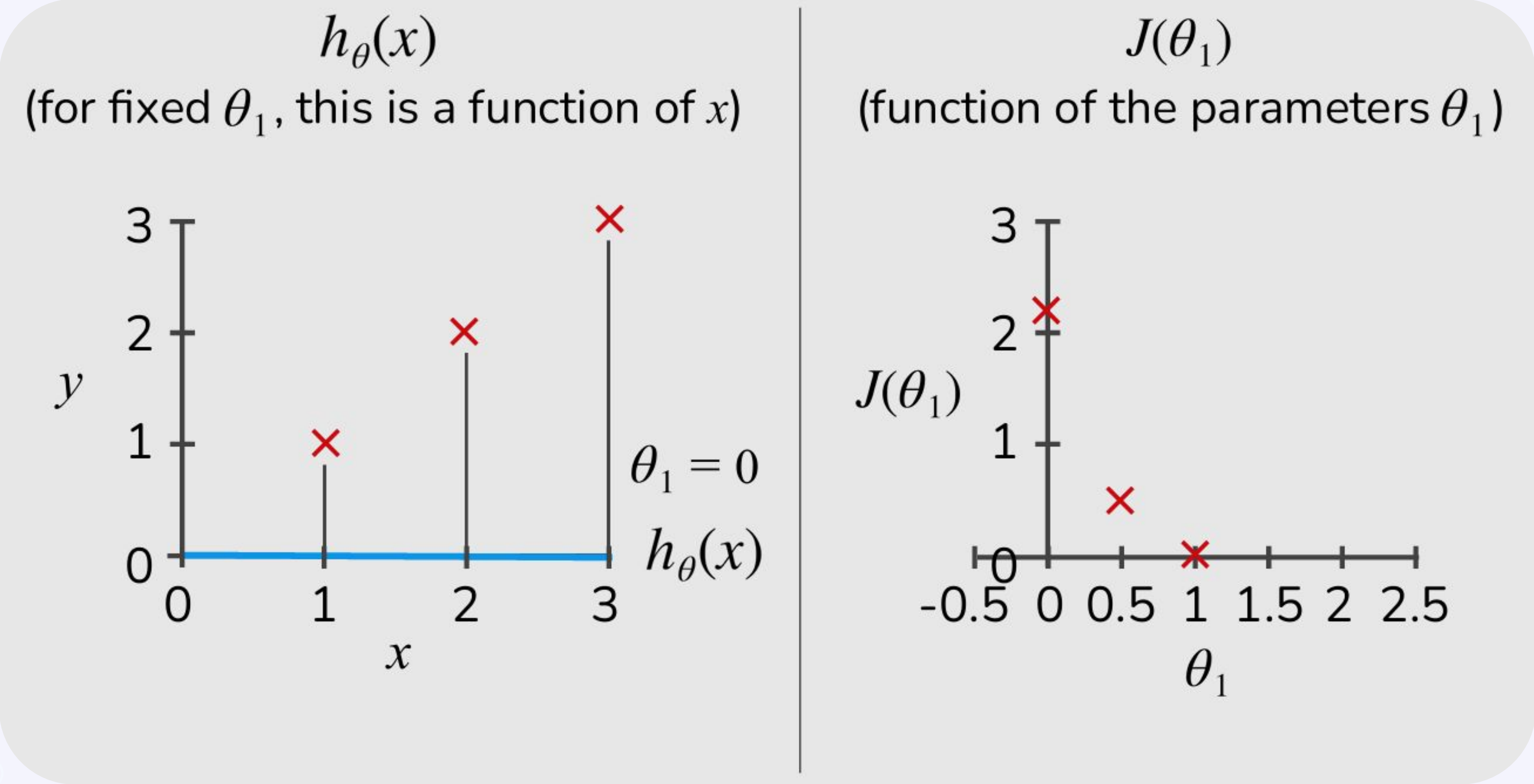
Função de Custo



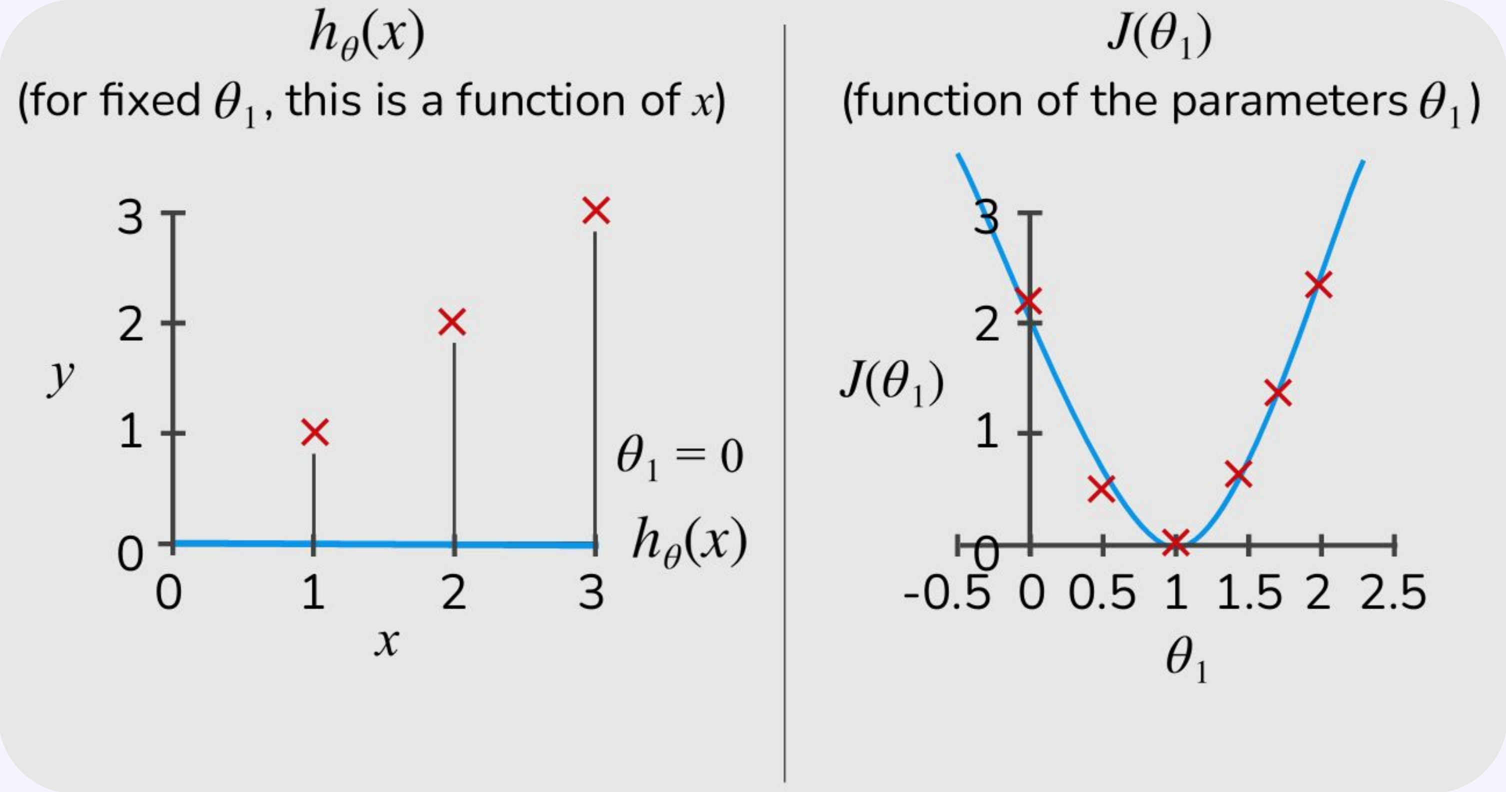
Função de Custo



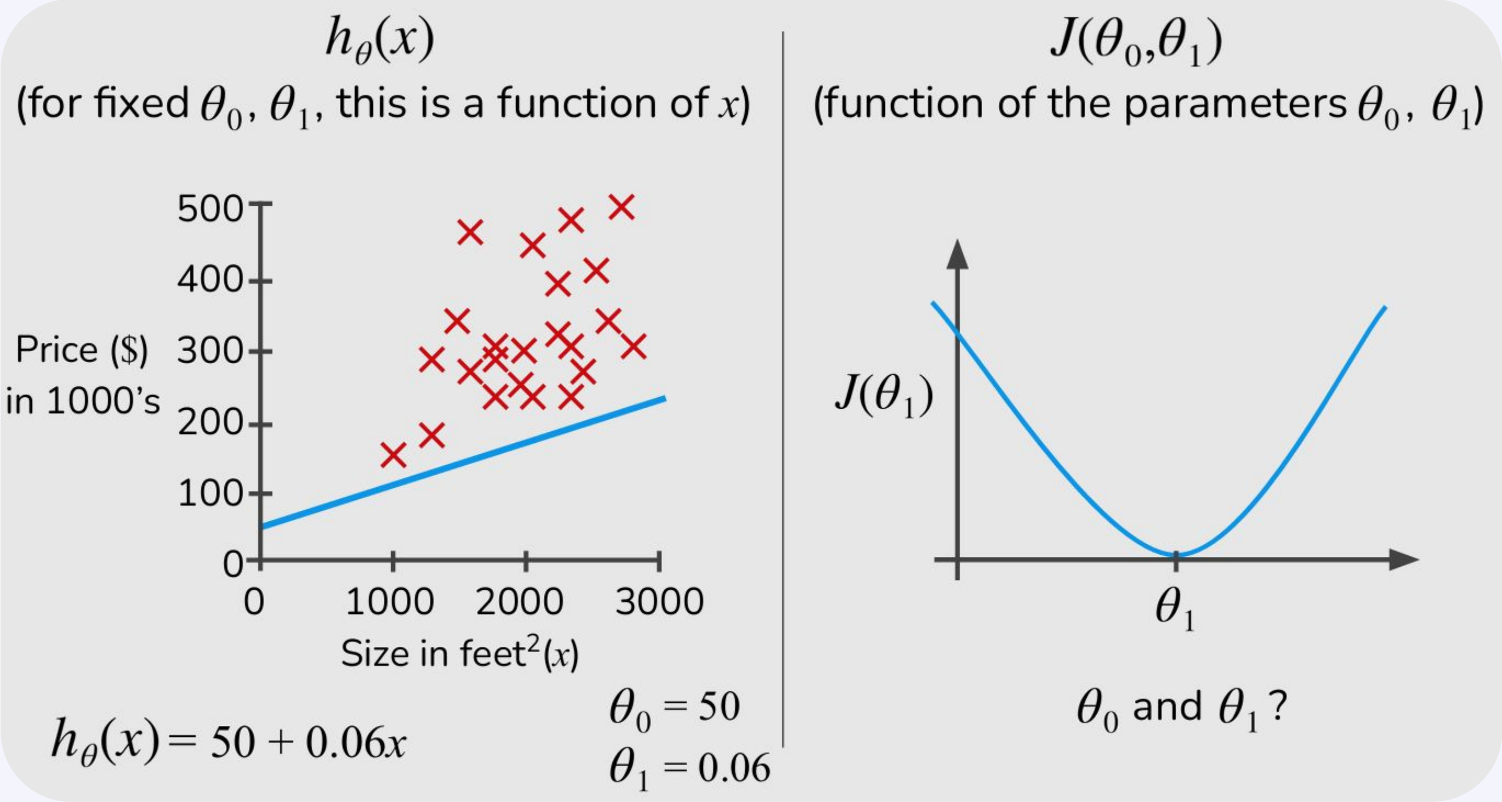
Função de Custo



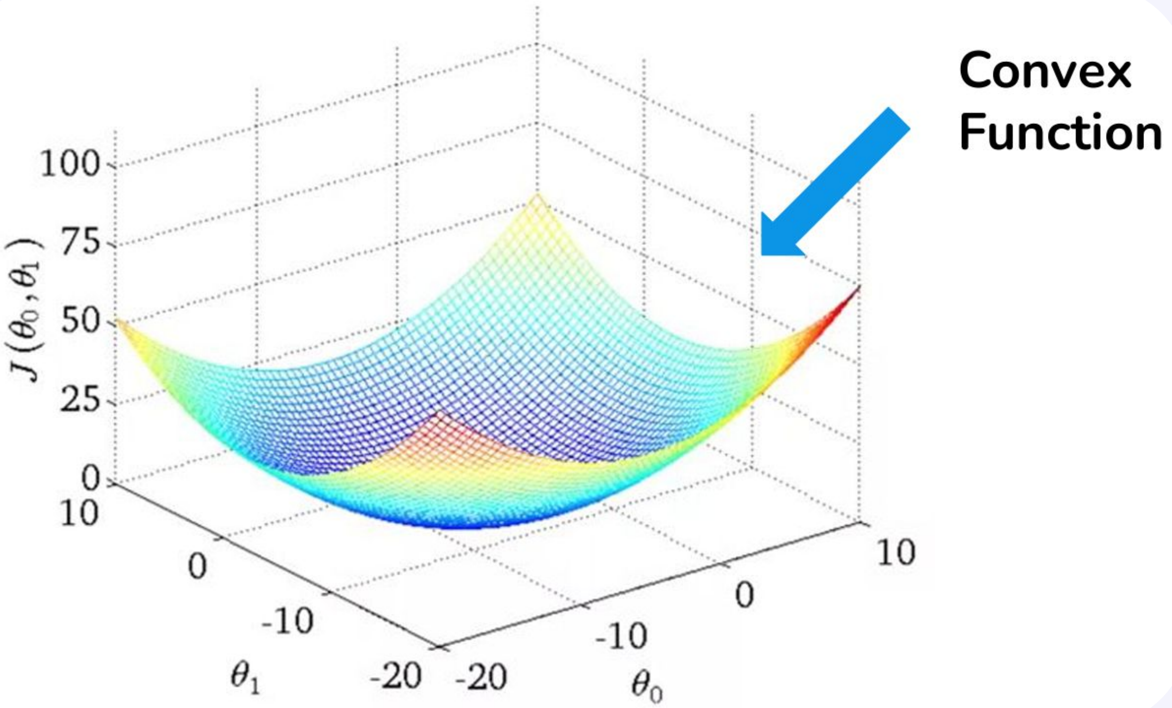
Função de Custo



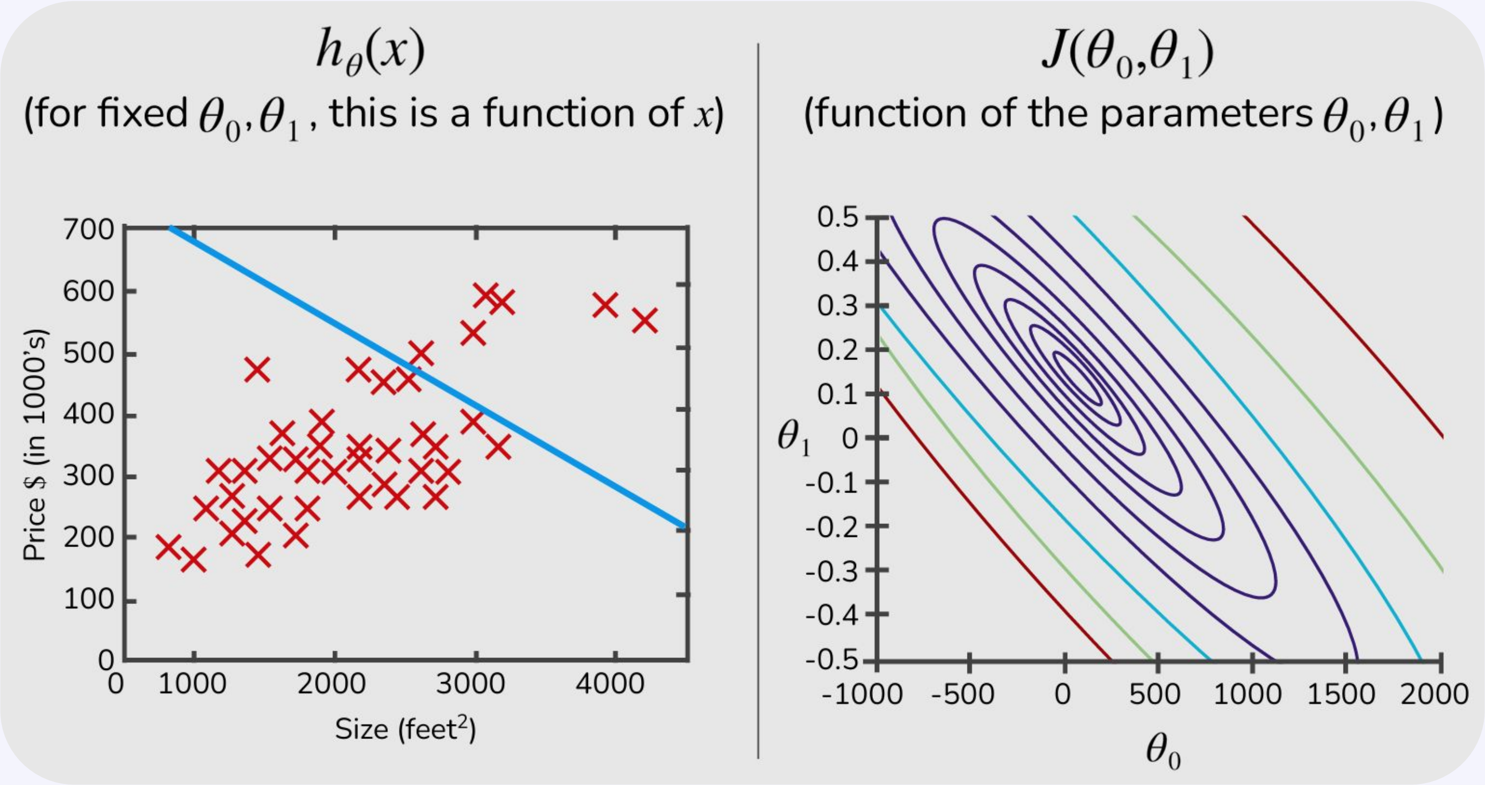
Função de Custo



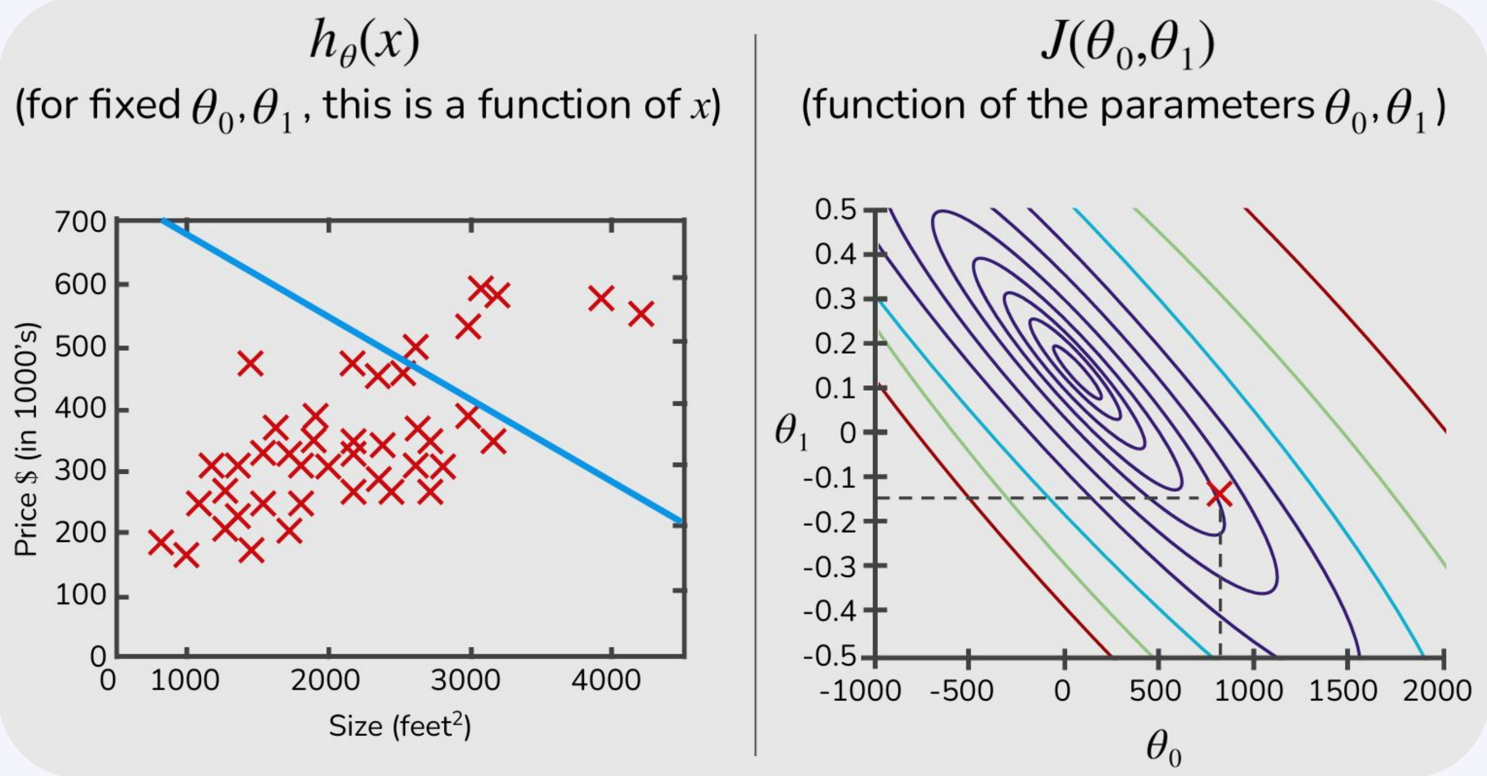
Função de Custo



Função de Custo



Função de Custo



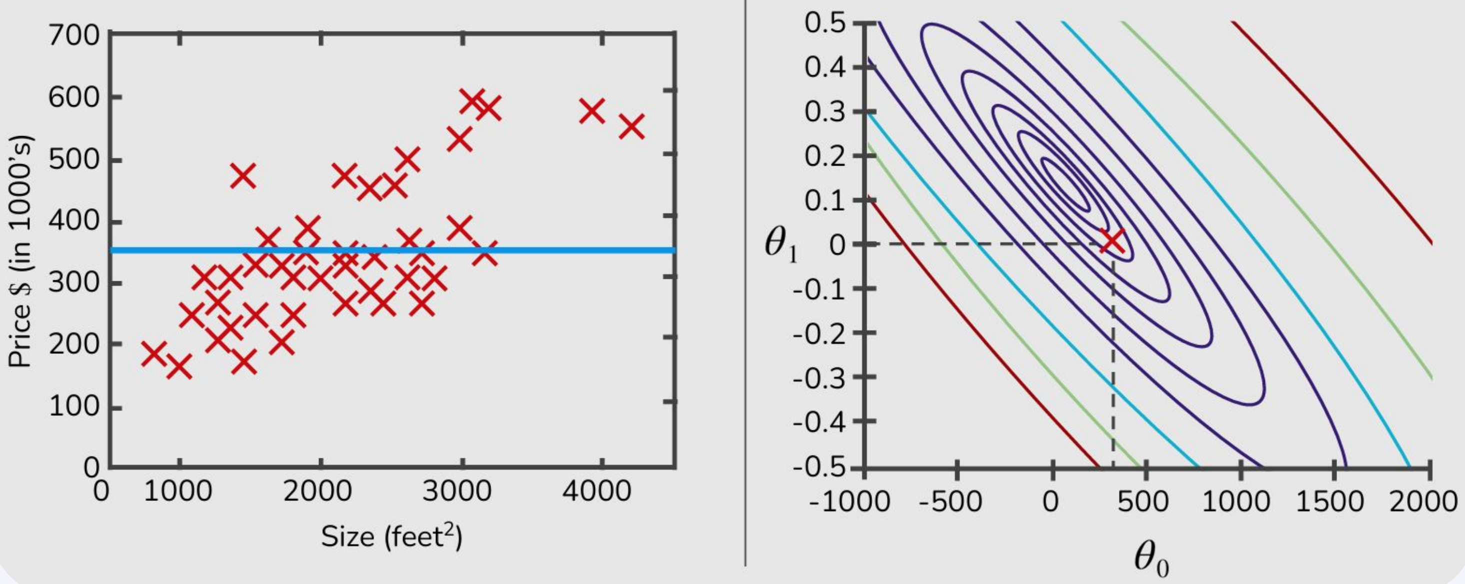
Função de Custo

$h_\theta(x)$

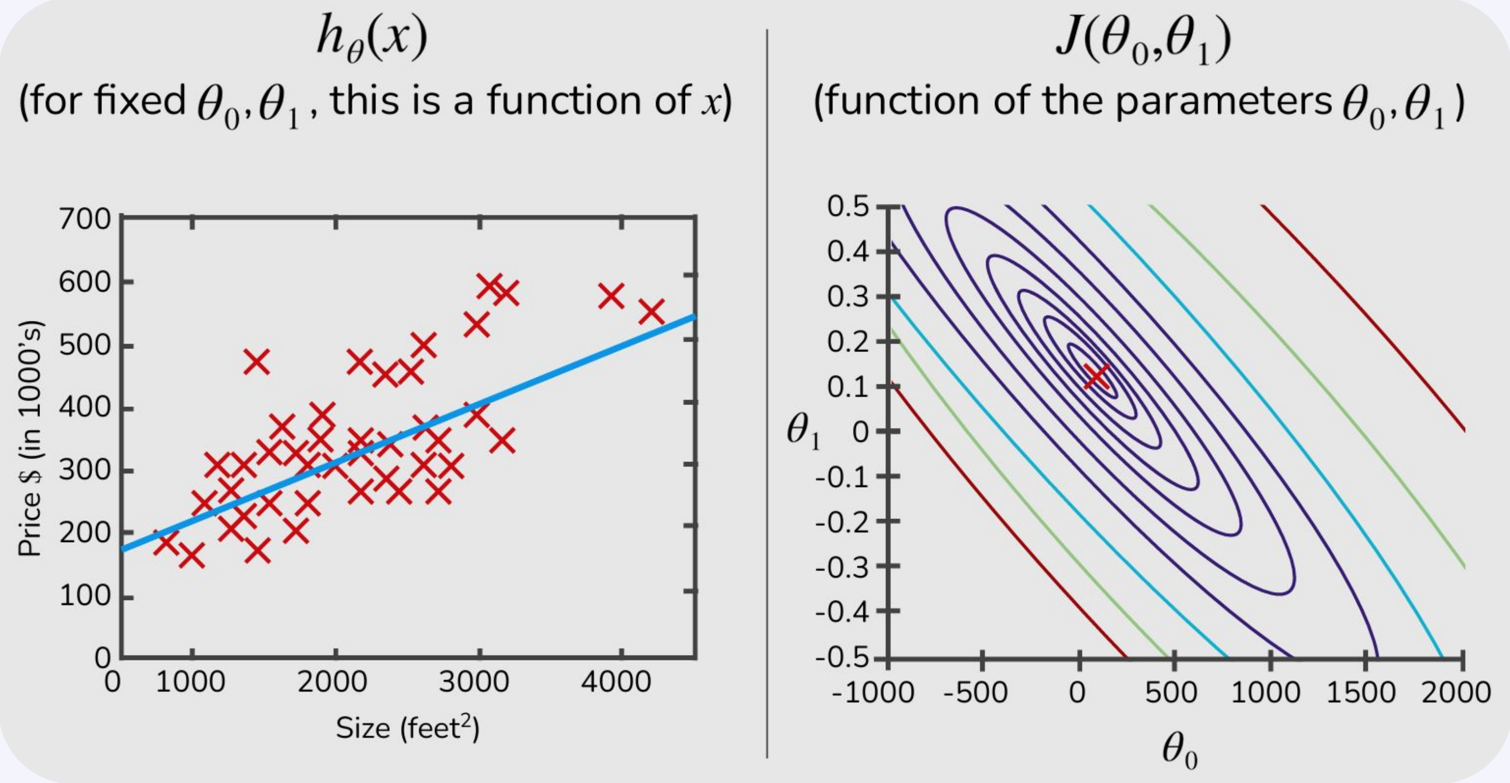
(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



Função de Custo



02

Minimização da Função de Custo

O que é minimizar uma função?

Minimizar uma função é encontrar os valores de entrada que retornam **o menor valor possível** para aquela função.

Este valor também é conhecido como o **mínimo global**.

Achar o mínimo global **nem sempre é possível**.

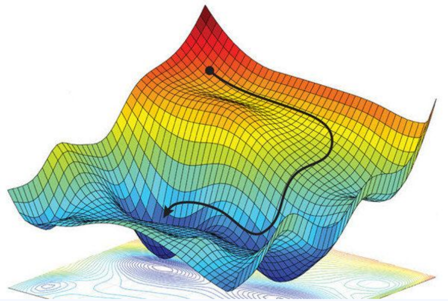
Às vezes, é possível encontrar apenas **mínimos locais**.

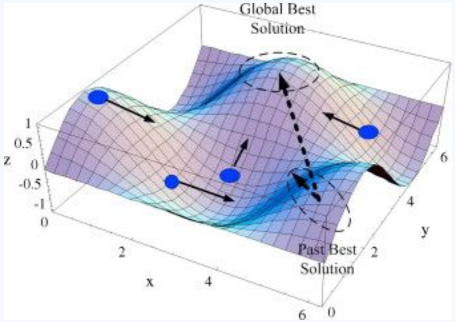
Encontrar **o mínimo** ou **o máximo** de funções são **problemas de otimização**.

Alguns algoritmos de otimização

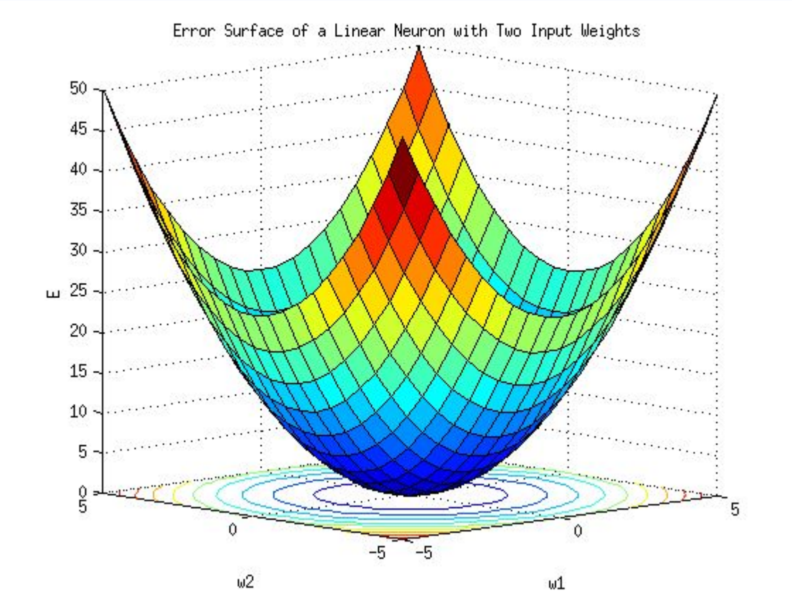
- MMQ (*OLS*) → Equação Normal (*Normal Equation*)
- **Gradiente Descendente (*Gradient Descent*)**
- Otimização por Enxame de Partículas (*Particle Swarm Optimization*)
- Algoritmo Genético (*Genetic Algorithm*)

$$\hat{\beta} = \left(X^1 X\right)^{-1} X^1 Y$$



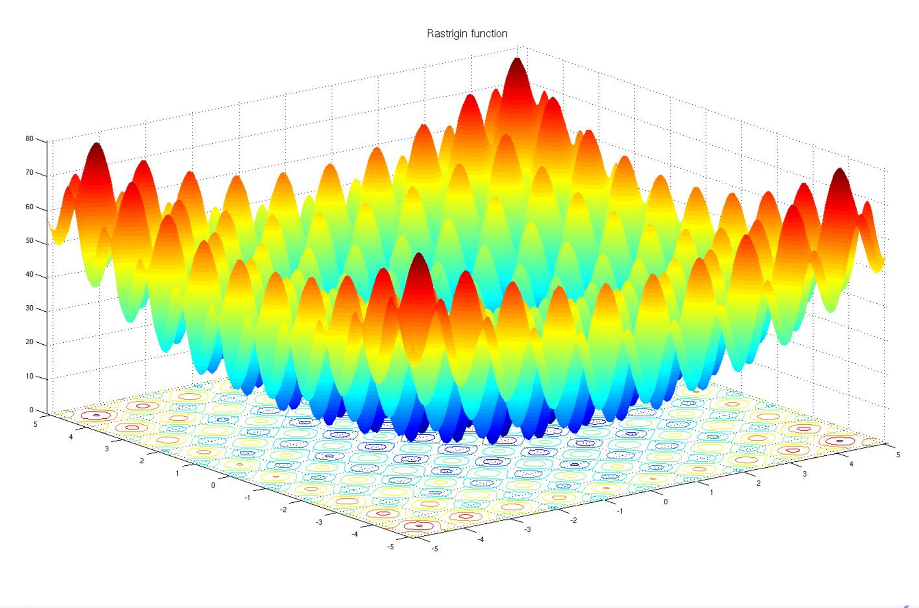


Alguns algoritmos de otimização



Função de Custo em forma de paraboloide
→ **MSE** usado em Regressão Linear

Função de benchmark (Rastrigin) de proc. otimização



Alguns algoritmos de otimização

Exemplo: a classe `LinearRegression` do *sklearn*.

API Reference > sklearn.linear_model > LinearRegression

LinearRegression

class sklearn.linear_model.**LinearRegression**(*, *fit_intercept*=True, *copy_X*=True, *n_jobs*=None, *positive*=False) [source]

Ordinary least squares Linear Regression.

LinearRegression fits a linear model with coefficients w = (w1, ..., wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

Parameters:

fit_intercept : *bool, default=True*

Whether to calculate the intercept for this model. If set to False, no intercept will be used in calculations (i.e. data is expected to be centered).

copy_X : *bool, default=True*

Notes

From the implementation point of view, this is just plain Ordinary Least Squares (scipy.linalg.lstsq) or Non Negative Least Squares (scipy.optimize.nnls) wrapped as a predictor object.

Examples

```
>>> import numpy as np
>>> from sklearn.linear_model import LinearRegression
>>> X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
>>> # y = 1 * x_0 + 2 * x_1 + 3
>>> y = np.dot(X, np.array([1, 2])) + 3
>>> reg = LinearRegression().fit(X, y)
>>> reg.score(X, y)
1.0
>>> reg.coef_
array([1., 2.])
>>> reg.intercept_
3.0...
>>> reg.predict(np.array([[3, 5]]))
array([16.])
```

fit(*X*, *y*, *sample_weight*=None) [source]

Fit linear model.

Parameters:

Alguns algoritmos de otimização

Exemplo: a classe `SGDRegressor` do *sklearn*.

API Reference > sklearn.linear_model > SGDRegressor

SGDRegressor

```
class sklearn.linear_model.SGDRegressor(loss='squared_error', *,
penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_intercept=True, max_iter=1000,
tol=0.001, shuffle=True, verbose=0, epsilon=0.1, random_state=None,
learning_rate='invscaling', eta0=0.01, power_t=0.25, early_stopping=False,
validation_fraction=0.1, n_iter_no_change=5, warm_start=False,
average=False)
```
[source]

Linear model fitted by minimizing a regularized empirical loss with SGD.

SGD stands for Stochastic Gradient Descent: the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule (aka learning rate).

The regularizer is a penalty added to the loss function that shrinks model parameters towards the zero vector using either the squared euclidean norm L2 or the absolute norm L1 or a combination of both (Elastic Net). If the parameter update crosses the 0.0 value because of the regularizer, the update is truncated to 0.0 to allow for learning sparse models and achieve online feature selection.

This implementation works with data represented as dense numpy arrays of floating point values for the features.

```
fit(X, y, coef_init=None, intercept_init=None, sample_weight=None)
```
[source]

Fit linear model with Stochastic Gradient Descent.

Parameters:

X : *{array-like, sparse matrix}, shape (n_samples, n_features)*

Training data.

y : *ndarray of shape (n_samples,)*

Target values.

coef_init : *ndarray of shape (n_features,), default=None*

The initial coefficients to warm-start the optimization.

intercept_init : *ndarray of shape (1,), default=None*

The initial intercept to warm-start the optimization.

sample_weight : *array-like, shape (n_samples,), default=None*

Weights applied to individual samples (1. for unweighted).

Returns:

self : *object*

Fitted `SGDRegressor` estimator.

Alguns algoritmos de otimização

Exemplo: a classe `LogisticRegression` do *sklearn*.

scikit learn

API Reference > sklearn.linear_model > LogisticRegression

LogisticRegression

```
class sklearn.linear_model.LogisticRegression(penalty='l2', *, dual=False, tol=0.0001,
C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None,
solver='lbfgs', max_iter=100, multi_class='deprecated', verbose=0, warm_start=False,
n_jobs=None, l1_ratio=None)
```
[source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

Read more in the User Guide.

Parameters:

solver : *{'lbfgs', 'liblinear', 'newton-cg', 'newton-cholesky', 'sag', 'saga'}, default='lbfgs'*

Algorithm to use in the optimization problem. Default is 'lbfgs'. To choose a solver, you might want to consider the following aspects:

- For small datasets, 'liblinear' is a good choice, whereas 'sag' and 'saga' are faster for large ones;
- For multiclass problems, only 'newton-cg', 'sag', 'saga' and 'lbfgs' handle multinomial loss;
- 'liblinear' and 'newton-cholesky' can only handle binary classification by default. To apply a one-versus-rest scheme for the multiclass setting one can wrapt it with the `OneVsRestClassifier`.
- 'newton-cholesky' is a good choice for `n_samples` >> `n_features`, especially with one-hot encoded categorical features with rare categories. Be aware that the memory usage of this solver has a quadratic dependency on `n_features` because it explicitly computes the Hessian matrix.

Warning

The choice of the algorithm depends on the penalty chosen and on (multinomial) multiclass support:

solver	penalty	multinomial multiclass
'lbfgs'	'l2', None	yes
'liblinear'	'l1', 'l2'	no
'newton-cg'	'l2', None	yes
'newton-cholesky'	'l2', None	no
'sag'	'l2', None	yes
'saga'	'elasticnet', 'l1', 'l2', None	yes

Note

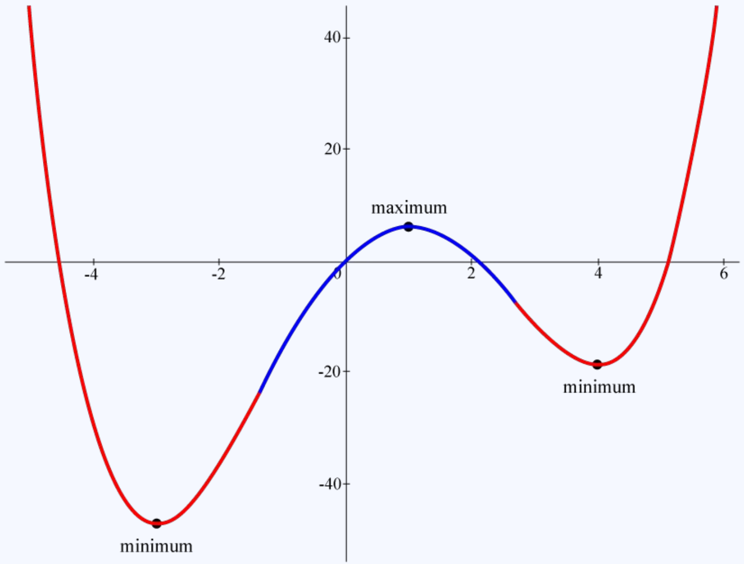
'sag' and 'saga' fast convergence is only guaranteed on features with approximately the same scale. You can preprocess the data with a scaler from

03

Gradiente Descendente

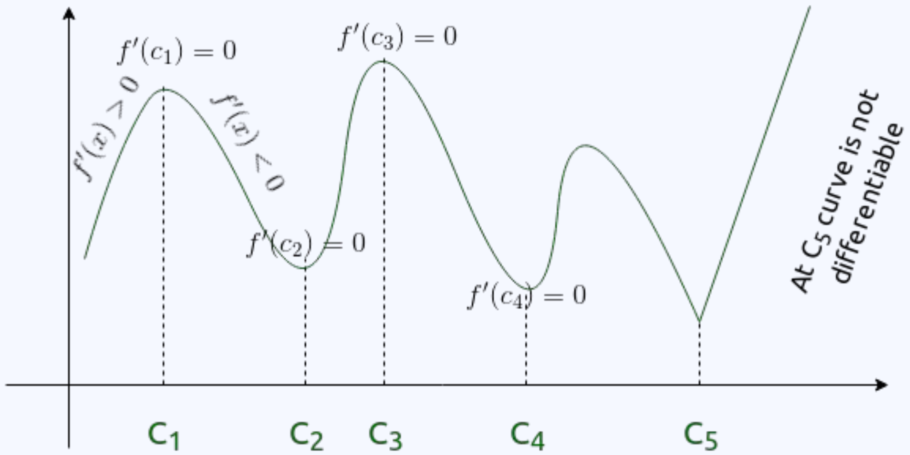
Relembrando derivadas

Como minimizamos funções em Cálculo Diferencial?



Calcworkshop.com

Relembrando derivadas



Derivada:

"Taxa" de variação de uma função em certo ponto.

Mínimo local:

Derivada de primeira ordem é igual a 0.

Testes de derivada de primeira e segunda ordem.

Relembrando derivadas

Vetor Gradiente:

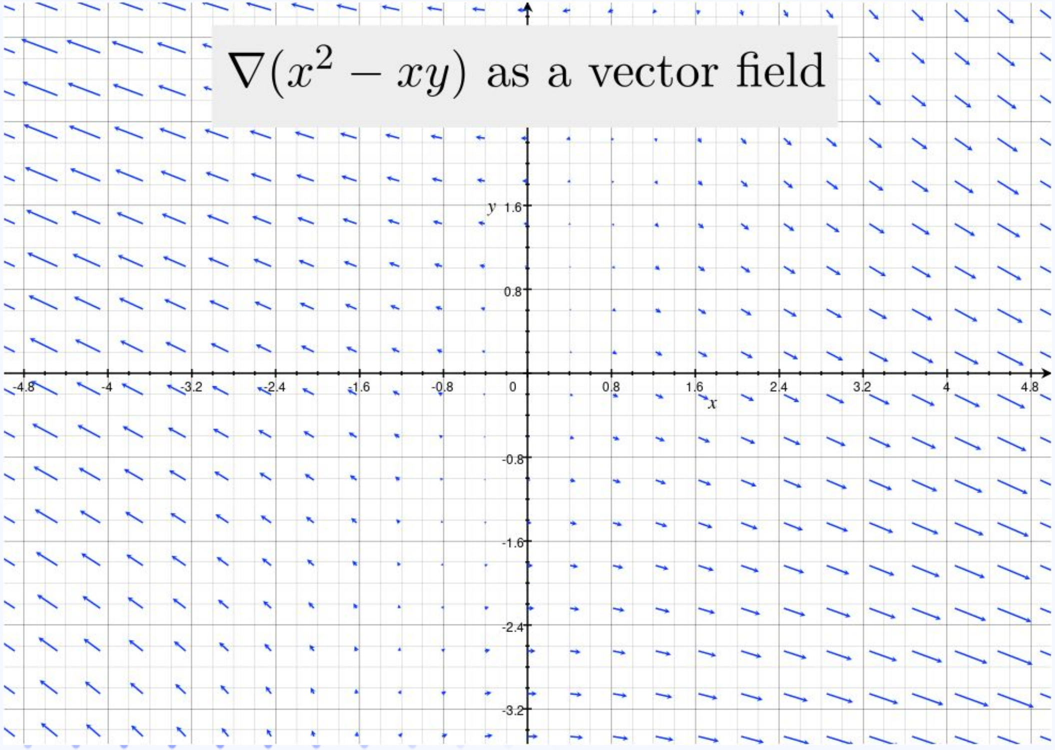
$$\nabla f = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z})$$

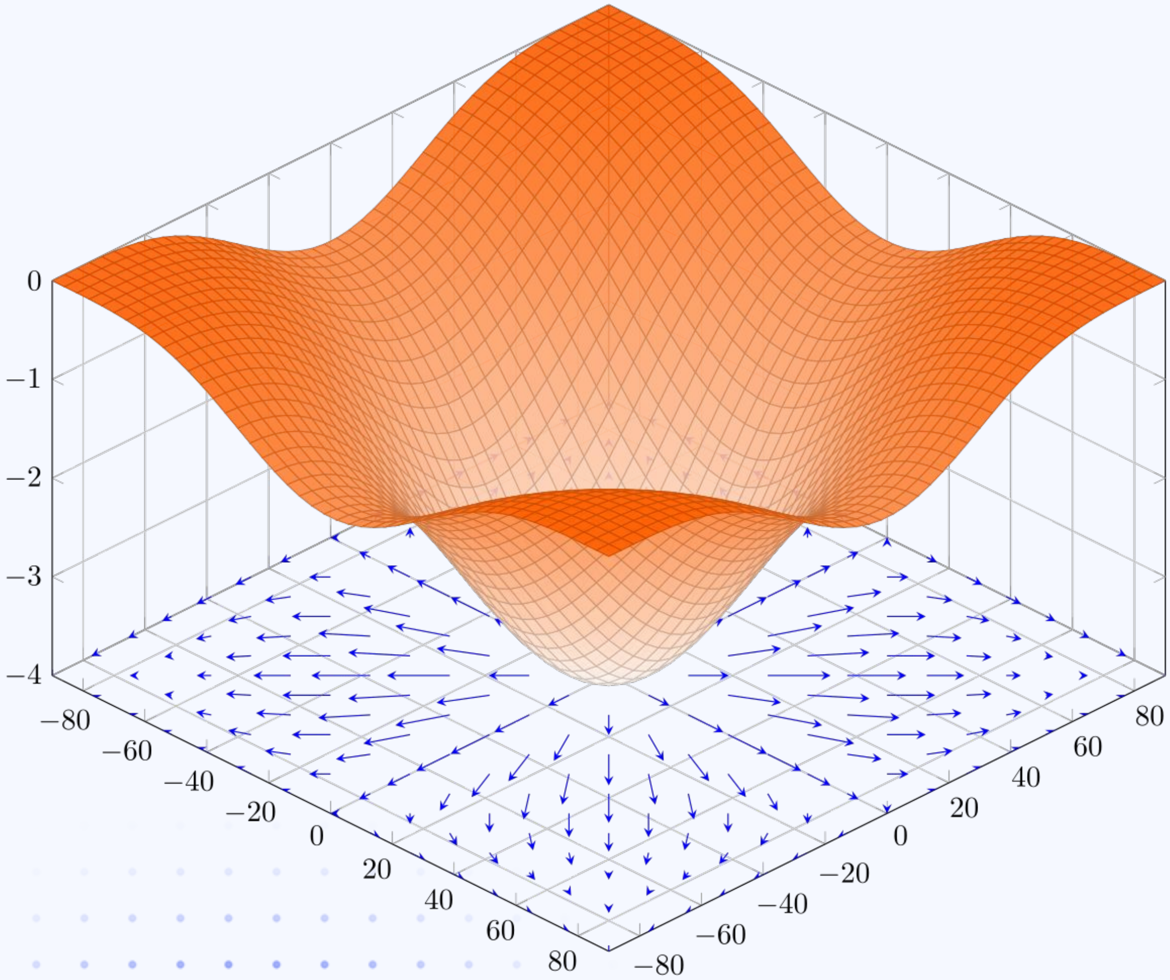
$$F(x, y, z) = x + y^2 + z^3$$

$$\nabla F(x, y, z) = (\frac{dF}{dx}, \frac{dF}{dy}, \frac{dF}{dz}) = (1, 2y, 3z^2)$$

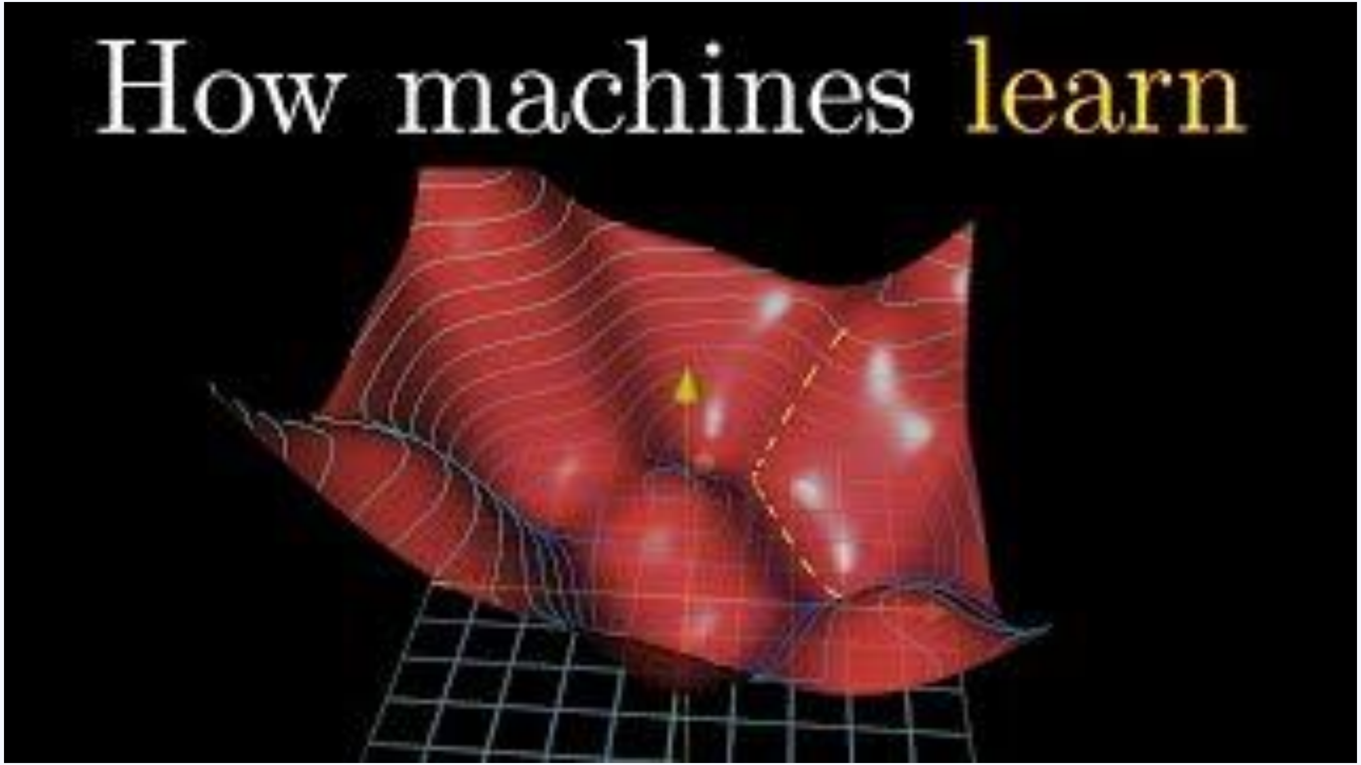
Name	Symbol	Example
Derivative	$\frac{d}{dx}$	$\frac{d}{dx}(x^2) = 2x$
Partial derivative	$\frac{\partial}{\partial x}$	$\frac{\partial}{\partial x}(x^2 - xy) = 2x - y$
Gradient	∇	$\nabla(x^2 - xy) = \begin{bmatrix} 2x - y \\ -x \end{bmatrix}$

Relembrando derivadas

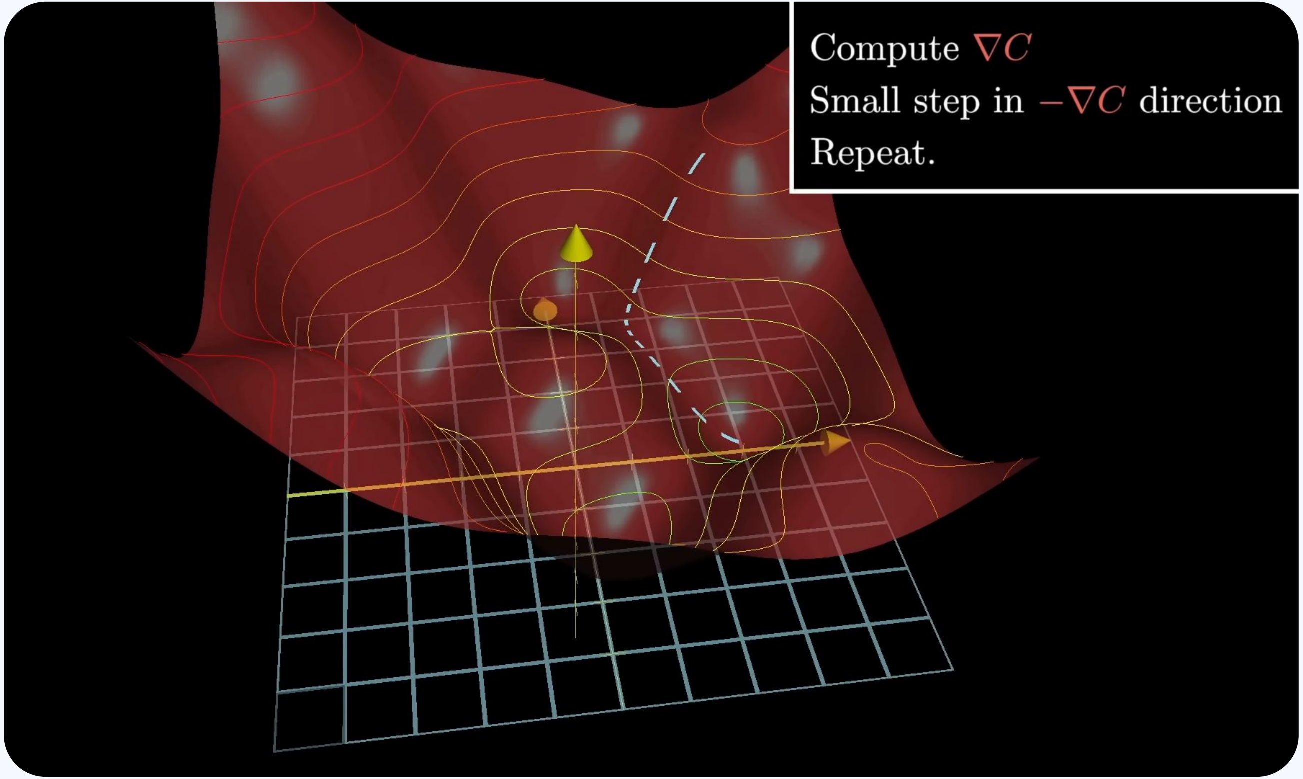



0
−1
−2
−3
−4
−80
−60
−40
−20
0
20
40
60
80
80
60
40
20
0
−20
−40
−60
−80

A descida do gradiente



A descida do gradiente



A descida do gradiente

$$\vec{\mathbf{W}} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{13,000} \\ w_{13,001} \\ w_{13,002} \end{bmatrix}$$

$$-\nabla C(\vec{\mathbf{W}}) = \begin{bmatrix} 0.31 \\ 0.03 \\ -1.25 \\ \vdots \\ 0.78 \\ -0.37 \\ 0.16 \end{bmatrix}$$

w_0 should increase somewhat
w_1 should increase a little
w_2 should decrease a lot

$w_{13,000}$ should increase a lot
$w_{13,001}$ should decrease somewhat
$w_{13,002}$ should increase a little

Treinando uma Regressão Linear

Gradient Descent algorithm

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

(for $j = 0$ and $j = 1$)

}

Linear Regression Model

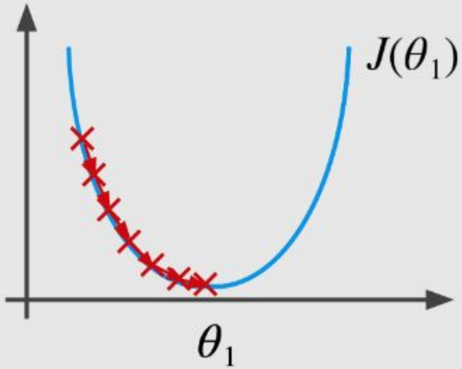
$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

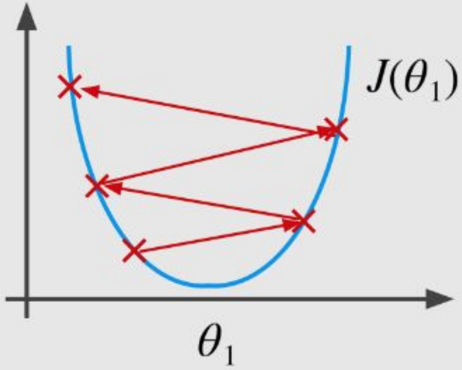
Treinando uma Regressão Linear

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.



If α is too large, gradient descent can be overshoot the minimum. It may fail to converge, or even diverge.



Treinando uma Regressão Linear

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$= \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

$$j = 0: \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

$$j = 1: \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

Treinando uma Regressão Linear

repeat until convergence {

$$\left.\begin{aligned}\theta_0 &:= \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})\\ \theta_1 &:= \theta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})\,x^{(i)}\end{aligned}\right\}$$

update θ_0 and θ_1 simultaneously

}

Batch Gradient Descent

Cada **passo**, leva em conta **todas as amostras** de treinamento:

repeat until convergence {

$$\left.\begin{aligned} \theta_0 &:= \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \\ \theta_1 &:= \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x^{(i)} \end{aligned}\right\} \text{update } \theta_0 \text{ and } \theta_1 \text{ simultaneously}$$

}

Stochastic Gradient Descent

Cada **passo**, leva em conta **uma amostra** de treinamento:

repeat until convergence {

for $i \coloneqq 1, \ldots, m$ {

$$\theta_0 := \theta_0 - \alpha(h_\theta(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$$

}

}

Mini-batch Gradient Descent

Cada **passo**, leva em conta ***b* amostras** de treinamento:

Say $b = 10$, $m = 1000$.

repeat until convergence {

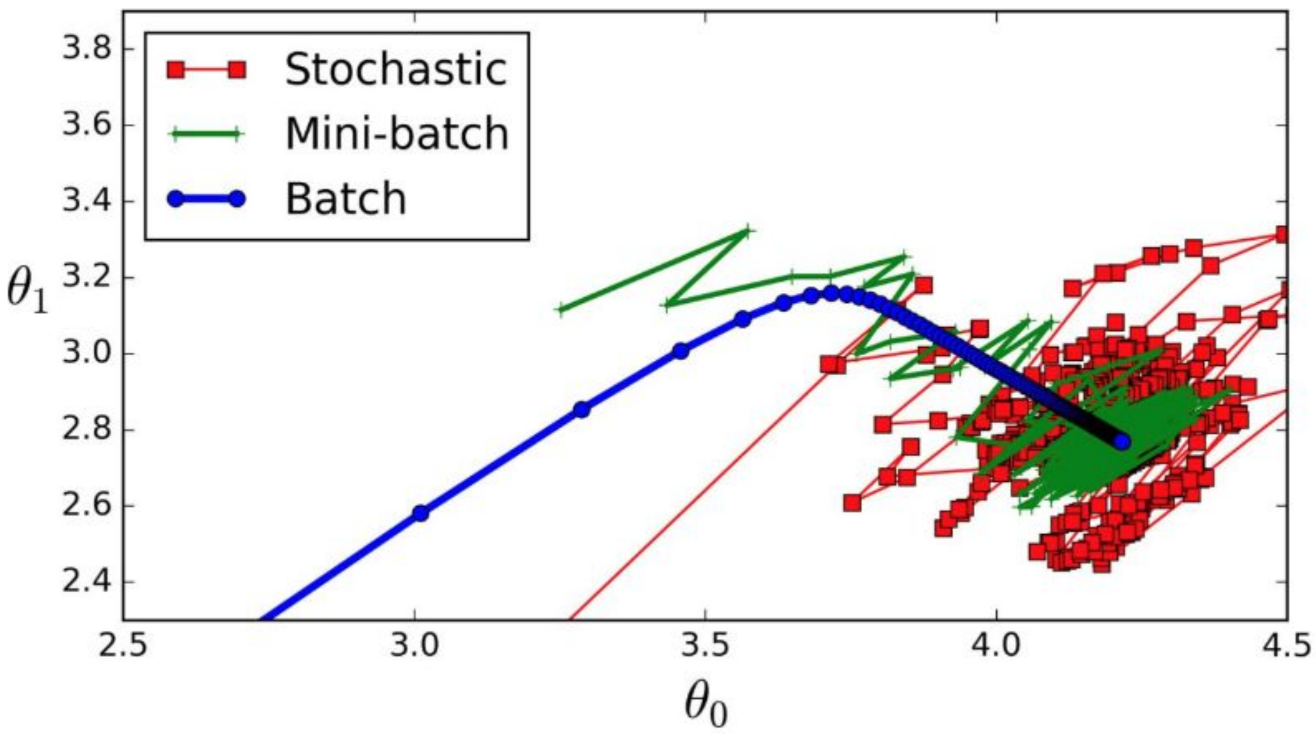
for $i := 1, 11, 21\ldots, 991$ {

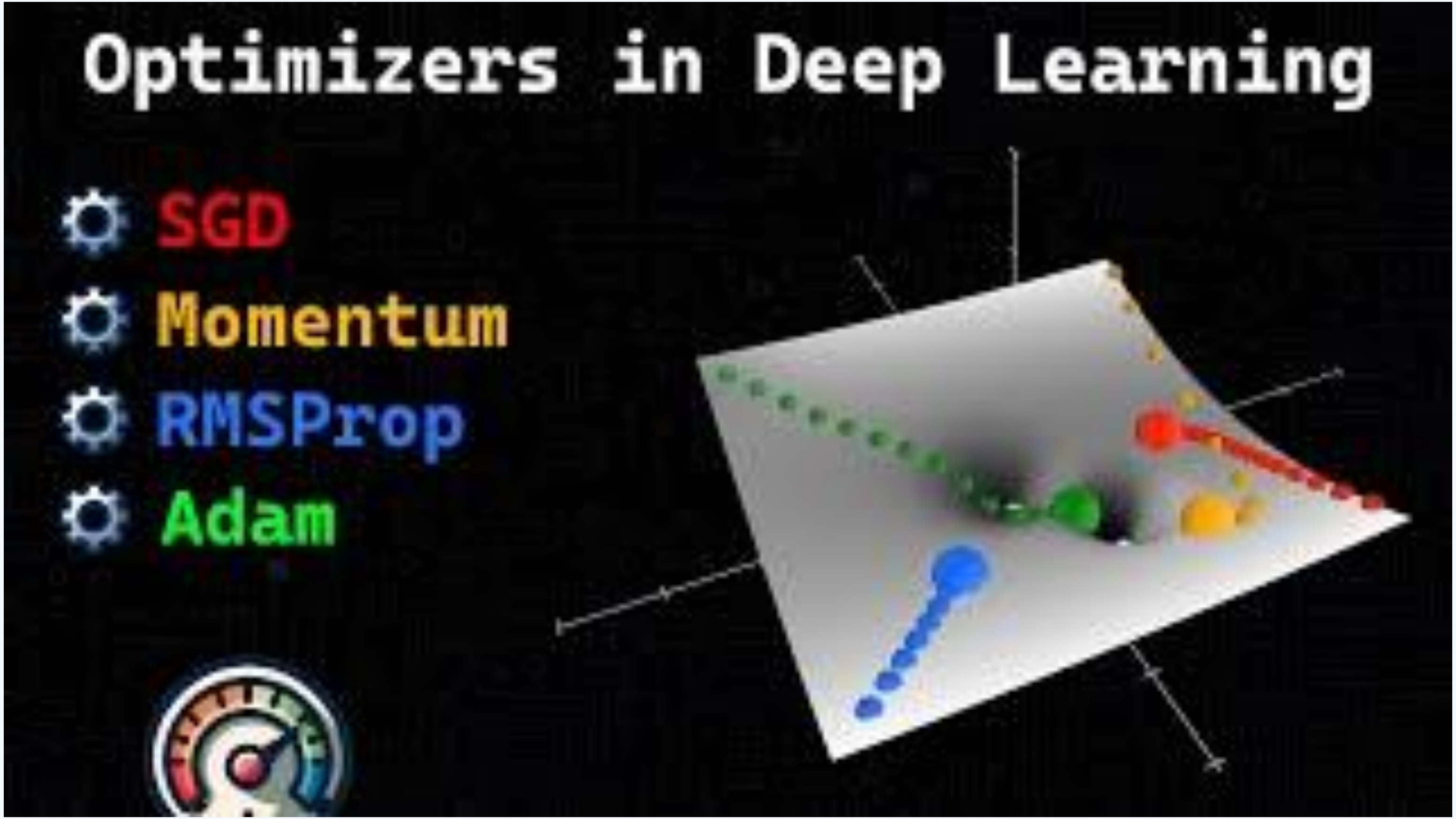
$$\theta_0 := \theta_0 - \alpha \frac{1}{10} \sum_{i=k}^{i+9} (h_\theta(x^{(k)}) - y^{(k)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{10} \sum_{i=k}^{i+9} (h_\theta(x^{(k)}) - y^{(k)})\, x^{(k)}$$

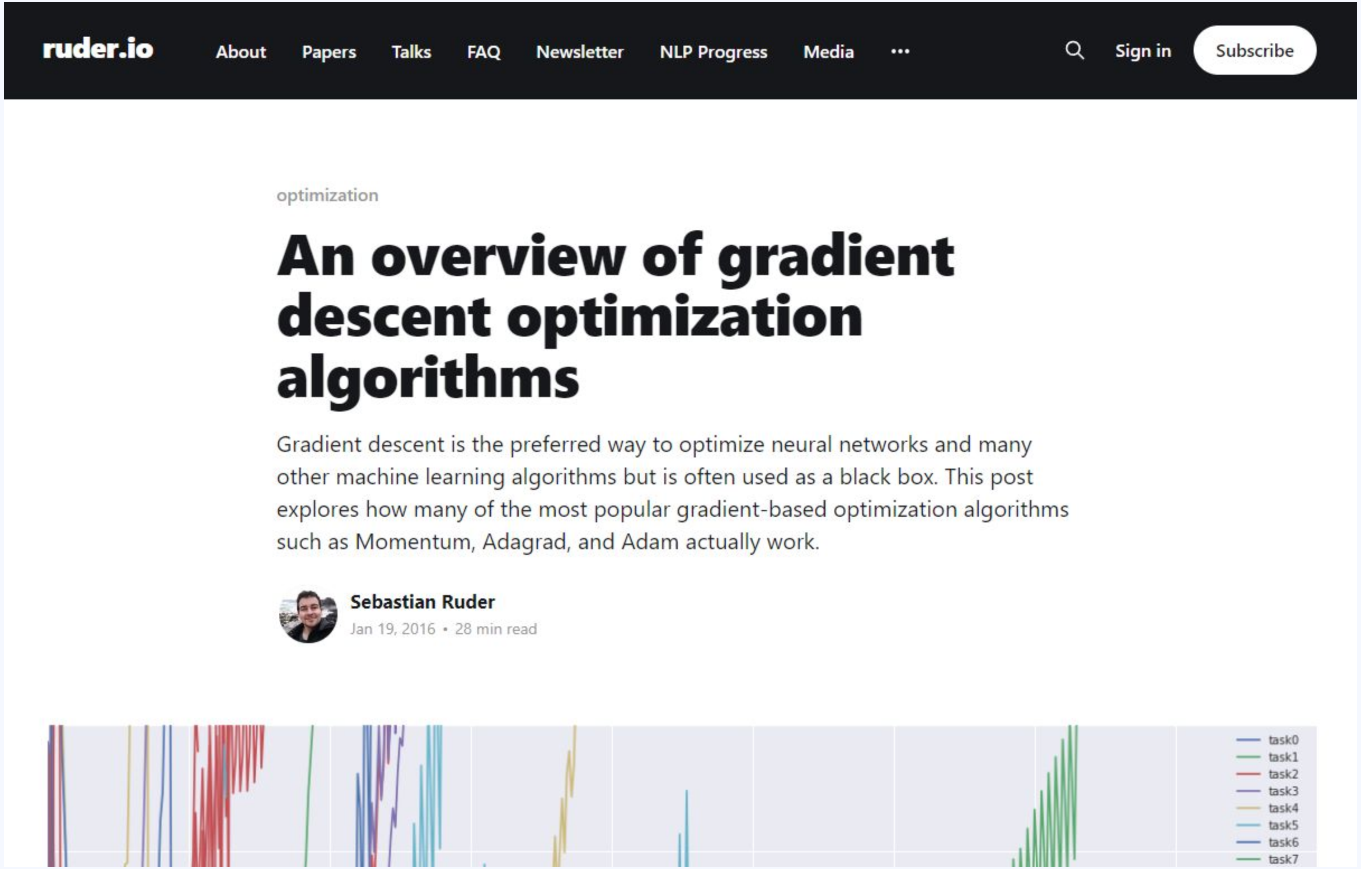
} }

Batch vs Mini-batch vs Stochastic





Instantes: *01:17-04:24; 04:38-04:54; 05:37-07:08; 12:30-12:44; 15:40-16:27; 16:28-16:50.*


ruder.io About Papers Talks FAQ Newsletter NLP Progress Media ••• Sign in Subscribe

optimization

An overview of gradient descent optimization algorithms

Gradient descent is the preferred way to optimize neural networks and many other machine learning algorithms but is often used as a black box. This post explores how many of the most popular gradient-based optimization algorithms such as Momentum, Adagrad, and Adam actually work.

Sebastian Ruder
Jan 19, 2016 • 28 min read

Link para o artigo: https://www.ruder.io/optimizing-gradient-descent/

04

Equação Normal

O que é a Equação Normal?

Basicamente se trata de:

Implementar o Método dos Mínimos Quadrados (MMQ)

Em inglês, *Ordinary Least Squares* (*OLS*)

A **Equação Normal** representa a fórmula para resolver uma Regressão Linear **de maneira analítica**:

$$\frac{d}{d\theta} J(\theta) = \ldots = 0 \quad \text{Solve for } \theta$$

Qual a Equação Normal?

Considerando **m exemplos** e **n *features***, temos:

x_0	Size (feet^2) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price (\$) in 1000's y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$$

Qual a Equação Normal?

Considerando **m exemplos** e **n *features***, temos:

x_0	Size (feet^2) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($) in 1000's y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

Qual a Equação Normal?

Considerando **m exemplos** e **n *features***, temos:

x_0	Size (feet^2) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($) in 1000's y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

Qual a Equação Normal?

Considerando **m exemplos** e **n *features***, temos:

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}_{m \times (n+1)} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}_{m}$$

Qual a Equação Normal?

Considerando **m exemplos** e **n *features***, temos:

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$X = \begin{bmatrix} \text{---} & (x^{(1)})^T & \text{---} \\ \text{---} & (x^{(2)})^T & \text{---} \\ \text{---} & \vdots & \text{---} \\ \text{---} & (x^{(m)})^T & \text{---} \end{bmatrix}$$

Qual a Equação Normal?

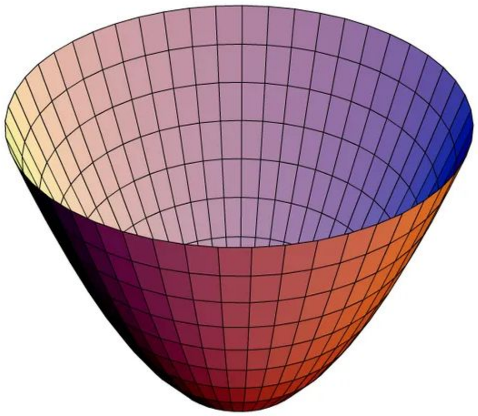
É possível chegar à seguinte expressão:

$$\theta = (X^T X)^{-1} X^T y$$

E se a matriz $X^T X$ **não for invertível?**

Provavelmente está utilizando duas *features* altamente correlacionadas, ou seja, **linearmente dependentes**.

Medium Search Write Sign up Sign in



https://qph.ec.quoracdn.net/main-qimg-99f7ea7d0929ed41dbecf67ec51b80b3?convert_to_webp=true

Rohan #3: Deriving the Normal Equation using matrix calculus

Understanding the analytical solution to linear regression.

Rohan Kapur · Follow
Published in A Year of Artificial Intelligence · 11 min read · Feb 16, 2016

Link para o artigo:
https://ayearofai.com/rohan-3-deriving-the-normal-equation-using-matrix-calculus-1a1b16f65dda

05

Implementando os algoritmos

Obrigado pessoal!

Até próxima aula :)

Iris Data Science UNICAMP

@irisdatascienceunicamp

Referências

Parte do material foi adaptado dos slides da Profª. Sandra Avila apresentados na disciplina MC886.

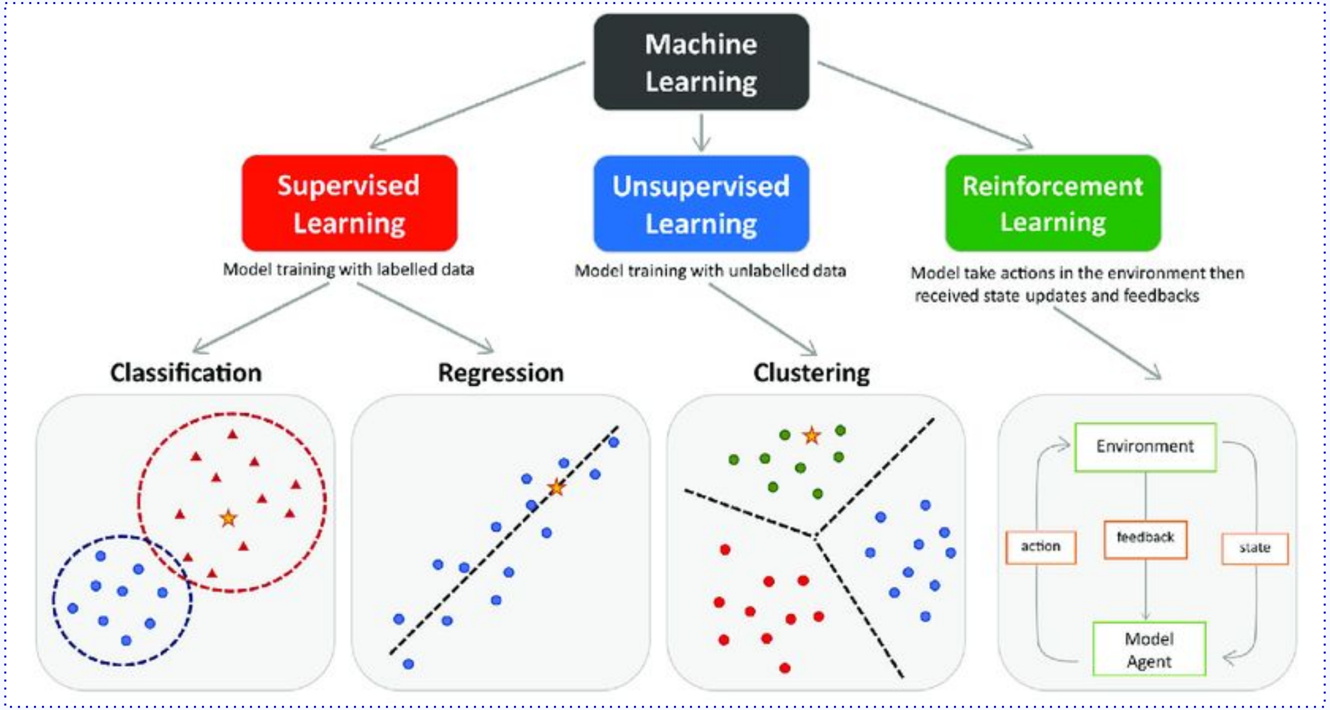
Material online:

- Gradient descent, how neural networks learn | Chapter 2, Deep learning (YouTube)

Livros utilizados:

- An Introduction to Statistical Learning (2023)
- Hands-On Machine Learning with Scikit-Learn, Keras, and Tensor Flow (2017)

Paradigmas de aprendizado



Retirado de:
https://www.researchgate.net/figure/The-main-types-of-machine-learning-Main-approaches-include-classification-and_fig1_354960266

Paradigmas de aprendizado

Aprendizado supervisionado:

O objetivo é "**predizer**" uma saída esperada/alvo.

Saída esperada /alvo é **conhecida**.

Aprendizado Não-supervisionado:

O objetivo é **encontrar padrões** na estrutura dos dados.

Não há uma saída esperada /alvo **conhecida**.

Aprendizado Auto-supervisionado:

O objetivo é extrair e **capturar os padrões** nos dados.

A saída esperada /alvo **faz parte dos dados de entrada**.

Como é o aprendizado supervisionado?

Dados sem
a resposta

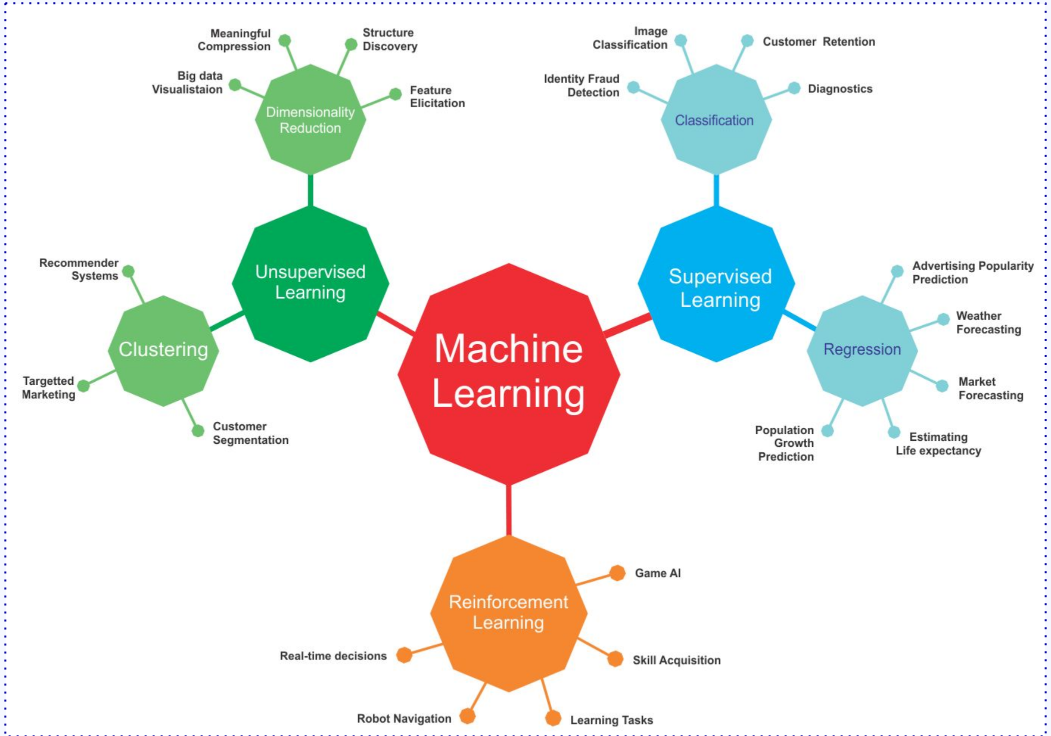
Processamento
do modelo

Chute de
resposta

O chute
está certo
ou errado?

Ajuste das
regras

Tarefas (ou “problemas”) em ML



Retirado de: https://subscription.packtpub.com/book/data/9781789345070/1/ch01lvl1sec04/ml-tasks

Tarefas supervisionadas

Regressão

O objetivo é **quantificar e inferir a relação** de uma **variável dependente** (***variável de resposta***) com **variáveis independentes** (***variáveis explicativas***).

Classificação

Identificar **a que categoria pertence uma nova observação**, **com base em um conjunto de dados** contendo observações cujas suas categorias são conhecidas.

Tipos de regressão

- Regressão Linear
- Regressão Polinomial
- Regressão Não-linear
- Regressão Logística

O aprendizado de máquina

Programação tradicional:

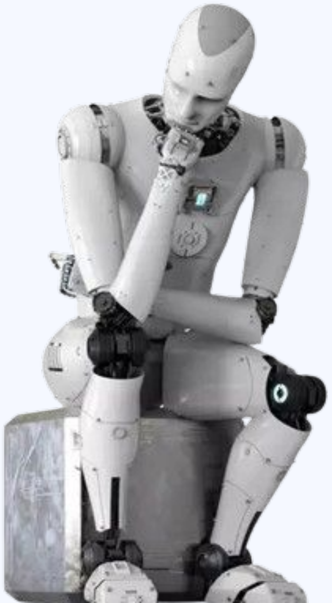
Dados
+
Regras

→ Processamento computacional → Respostas

Aprendizado de máquina:

Dados
+
Respostas
(opcional)

→ Processamento computacional → Regras



Revisão e atividade prática

Ambiente de desenvolvimento: Google Colab.

Atividade de revisão:

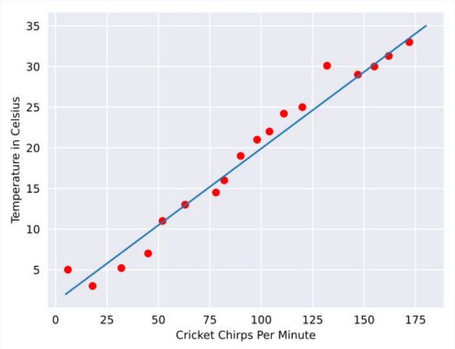
- Python;
- Manipulação de dados;
- Análise de dados;
- Operação com matrizes.

O que é Machine Learning?

"Um programa de computador **aprende a partir de uma experiência E** com respeito a uma **classe de tarefas T** e uma **métrica de performance P**, se a **sua performance em tarefas de T, quando medida por P, melhora com a experiência E**."

— Tom Mitchell (1997)

O que é Machine Learning?



Modelos que se ajustam aos dados:

- Regressão linear
- Regressão logística
- Algoritmos de Clusterização
- ...

Exemplos:

- Classificadores simples
- Sistemas preditivos
- Sistemas de recomendação
- ...

Retirado de:
https://training.galaxyproject.org/training-material/topics/statistics/tutorials/clustering_machinelearning/tutorial.html,https://www.javatpoint.com/classification-algorithm-in-machine-learning,https://developers.google.com/machine-learning/crash-course/descending-into-ml/linear-regression

O que é Deep Learning?

"A aprendizagem profunda é uma forma de aprendizagem de máquina que permite aos computadores aprender com a experiência e **compreender o mundo em termos de uma hierarquia de conceitos**. Como o computador adquire conhecimento a partir da experiência, **não é necessário que um operador de computador humano especifique formalmente todo o conhecimento que o computador precisa**. A hierarquia de conceitos permite que o computador **aprenda conceitos complicados construindo-os a partir de conceitos mais simples**."

— Goodfellow et al. (2016)

O que é Deep Learning?

Sistemas que descobrem, aprendem e combinam padrões aprendidos:

- CNNs
- RNNs e LSTMs
- Transformers
- ...

Exemplos:

- Sistemas de visão computacional
- Chatbots modernos
- IA generativa
- ...

Retirado de:
https://exame.com/inteligencia-artificial/o-que-e-chatgpt-como-usar-a-ia-em-portugues-no-seu-dia-a-dia/
https://medium.com/analytics-vidhya/yolo-explained-5b6f4564f31,
https://deepmind.google/discover/blog/alphafold-a-solution-to-a-50-year-old-grand-challenge-in-biology/

Emergência e homogeneização

Emergência

Surgimento de novos comportamentos.
“Soma das partes não é igual ao todo”.

Homogeneização

Uso de uma mesma técnica em várias áreas.

Retirado de: https://arxiv.org/abs/2108.07258

Emergência e homogeneização

Machine Learning

Emergência	Sistemas passam a ter a capacidade de descobrirem como resolver uma tarefa a partir de dados.
Homogeneização	Diversas aplicações passaram a poder se basear em um algoritmo genérico de aprendizado com base em dados.

Deep Learning

Emergência	Aprendizagem de conceitos abstratos sem a especificação explícita. A escala enorme de modelos trouxe características como *in-context learning*.
Homogeneização	As mesmas arquiteturas de modelos passaram a ser aplicáveis para diversos tipos de aplicações.

Introduction

Mercury is the closest planet to the Sun and the smallest one in the entire Solar System. **This planet's name has nothing to do with the liquid metal,** since Mercury was named after the Roman messenger god. Mercury's surface is filled with craters

Mercury takes a little more than 58 days to complete its rotation, so try to imagine how long days must be there! **Since the temperatures are so extreme, albeit not as extreme** as on Venus, Mercury has been deemed to be non-habitable for humans

“This is a quote, words full of wisdom that someone important said and can make the reader get inspired”

–Someone Famous

Concepts

Mercury

Mercury is the closest planet to the Sun and **the smallest one** in the Solar System—it's only a bit larger than the Moon

Venus

Venus has a beautiful name and is the **second planet from the Sun.** It's hot and has a poisonous atmosphere

What is this topic about?

Mercury

It's the closest planet to the Sun and the **smallest** in the Solar System

Venus

Venus has a beautiful name and is the second planet from the Sun

Mars

Despite being red, Mars is actually a **cold place.** It's full of iron oxide dust

Features of the topic

Mars

Despite **being red**, Mars is very cold

Neptune

It's the farthest planet from the Sun

Jupiter

Jupiter is the biggest planet of them all

Saturn

Saturn is a **gas giant** and has several rings

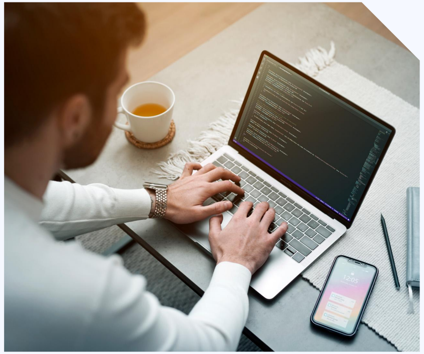
Examples

Mercury

It's the closest planet to the Sun and the **smallest** in the Solar System

Venus

Venus has a beautiful name and is the second planet from the Sun



Mars

Despite being red, Mars is actually a **cold place.** It's full of iron oxide dust

Recommendations

Mars

Despite being red, Mars is very cold

Venus

Venus is the second planet from the Sun

Neptune

Neptune is the farthest planet from the Sun

Mercury

Mercury is the closest planet to the Sun

Saturn

Saturn is a gas giant and has several rings

Jupiter

Jupiter is the biggest planet of them all

Image always reinforce the concept

You can give a brief description of the topic you want to talk about here. For example, if you want to talk about Mercury, you can say that it's the smallest planet in the entire Solar System

4,498,300,000

Big numbers catch your audience's attention

9h 55m 23s

Jupiter’s rotation period

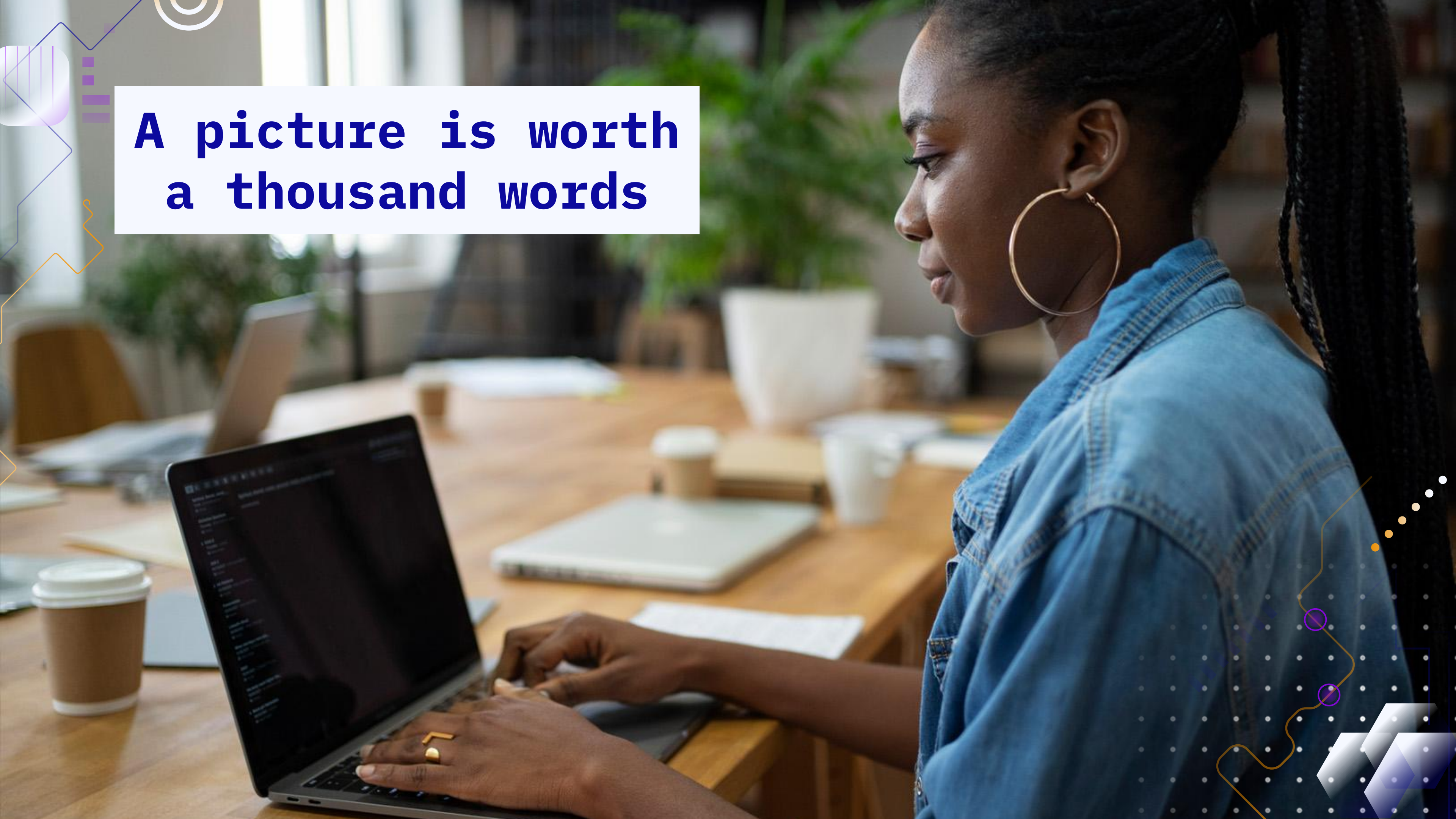
333,000

The Sun’s mass compared to Earth’s

386,000 km

Distance between the Earth and the Moon

Awesome words


A picture is worth
a thousand words

Practical exercise - calculator

Objective:

Introduce participants to basic coding concepts by building a **simple calculator**

Instructions:

1. Open a Python development environment and write the following code:

```
# Simple Calculator
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
print("Sum:", num1 + num2)
print("Difference:", num1 - num2)
print("Product:", num1 * num2)
print("Quotient:", num1 / num2)
```

2. Run the program and experiment with different numbers
3. Observe the output

Brainstorm and idea generation

- Power 1
  - Earth
    - Mercury: It's the smallest planet of them all
    - Venus: Venus is the second planet from the Sun
  - Mars
    - Neptune: It's the farthest planet from the Sun
    - Saturn: Saturn was named after a Roman god

Main topic and details

Mars

Despite being red,
Mars is **very cold**

Neptune

It’s the farthest
planet from the Sun

Jupiter

Jupiter is the biggest
planet of them all

Saturn

It’s a gas giant and
has **several rings**

Popular programming languages

01 — **Neptune** — Mercury is the closest planet to the Sun and the **smallest** of them all

02 — **Venus** — Venus has a beautiful name and is the **second planet from the Sun**

03 — **Earth** — Earth is the third planet from the Sun and the only one that harbors life in the Solar System

04 — **Saturn** — Saturn is a gas giant and has several rings. It’s composed mostly of hydrogen and helium

Sequences

Saturn is composed of **hydrogen and helium**

First

Mercury is the **closest** planet to the Sun

Next

Jupiter was named after a Roman god

Next

Next

Despite being red, Mars is **very cold**

Next

Earth is the third planet from the Sun

Last

Venus has extremely high temperatures

Classification

Mars	Venus	Mercury	Jupiter
• Small • Red • Cold • Rocky	• Small • Hot • Dry • Volcanic	• Small • Hot • Rocky • Cratered	• Large • Cold • Gassy • Striped
Mars is full of iron oxide dust	Venus has high temperatures	Mercury is quite a small planet	Jupiter is a huge gas giant

Cause and effect

Problem	**Mars** Despite being red, Mars is very cold	**Venus** Venus is the second planet from the Sun
Solution	**Mercury** Mercury is the closest planet to the Sun	**Saturn** Saturn is a gas giant and has several rings

Question and answer

Question

Is Mercury the closest planet to the Sun and the smallest one in the Solar System? **Note that it's a bit larger than the Moon**

Answer

Venus has a beautiful name and is **the second planet from the Sun.** It's hot and has a poisonous atmosphere

Step-by-step coding

01

Earth

It's the only planet known to **harbor life**

02

Mercury

Mercury is the closest planet to the Sun

03

Jupiter

Jupiter is the **biggest** planet of them all

04

Saturn

Saturn was named after a Roman god

Parts and whole

The whole objective

Mercury is the closest planet to the Sun and the smallest one in the entire Solar System

Parts of the object

- Mercury
- Jupiter
- Venus
- Mars
- Earth
- Saturn
- Mercury

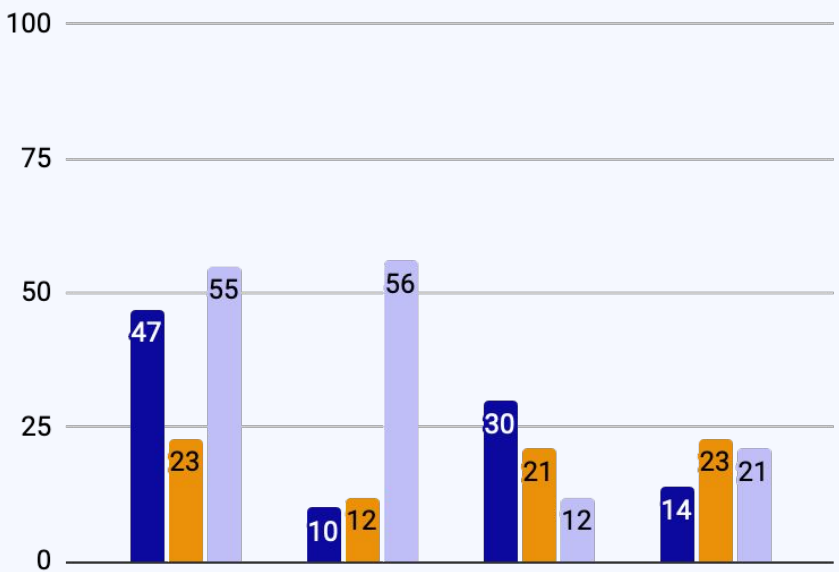
What happens if the parts are missing?

Earth is the third planet from the Sun and the **only one that harbors life in the Solar System**

What's the function of the parts?

Jupiter is a gas giant and the biggest planet in the Solar System

You can use this graph



Follow the link in the graph to modify its data and then paste the new one here. **For more info, click here**

Mercury

Mercury is the closest planet to the Sun

Jupiter

Jupiter is the biggest planet of them all

Saturn

Saturn was named after a Roman god

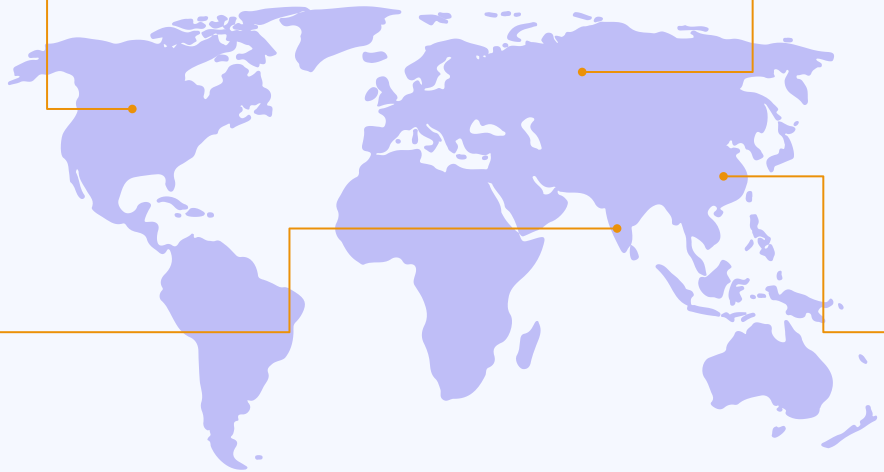
This is a map

USA

Despite being red, Mars is **very cold**

India

Jupiter is the biggest planet of them all



Russia

Neptune is the farthest planet from the Sun

China

Saturn is **a gas giant** and has several rings

Mockups

You can replace the images on the screen with your own work. Just right-click on them and select “Replace image”

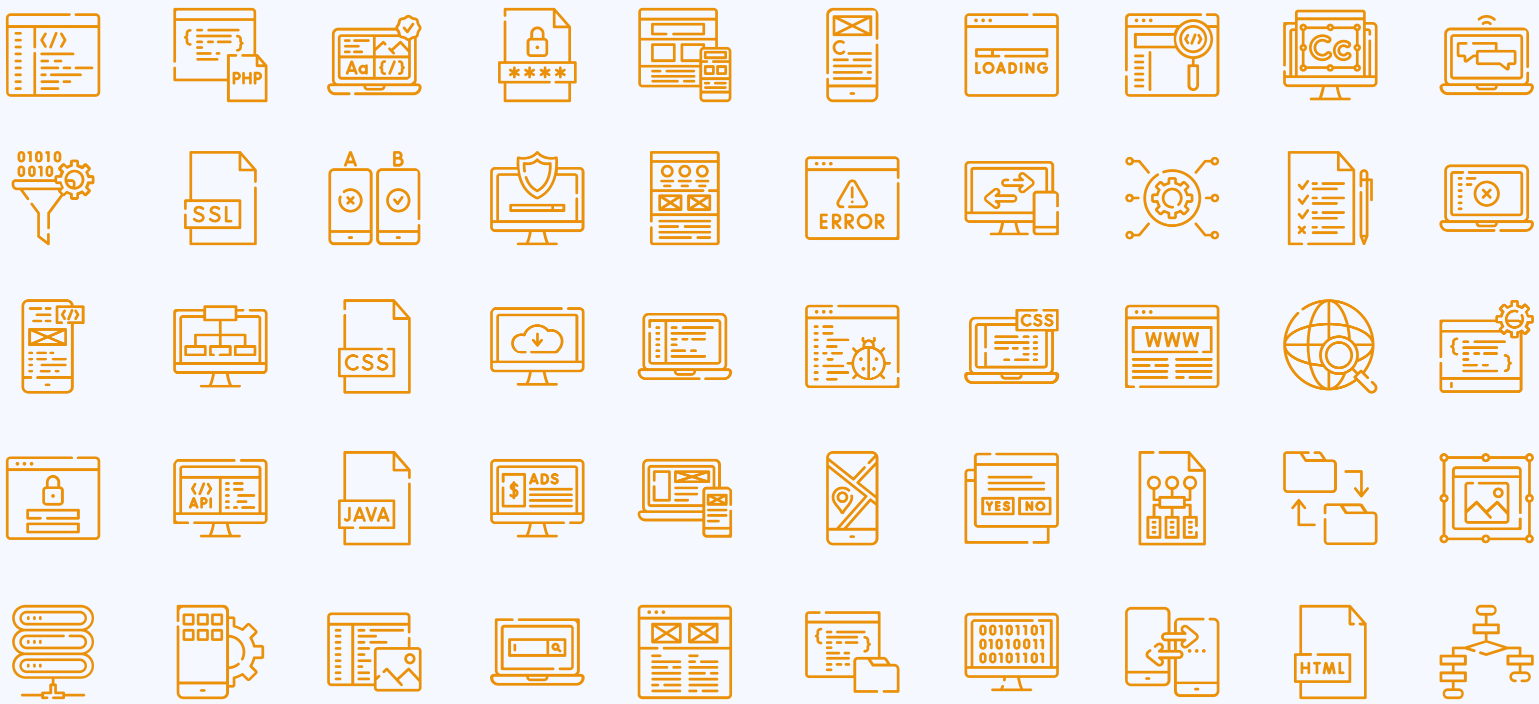
Obrigado!

Do you have any questions?

@irisdatascienceunicamp

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**

Icon pack



PHP
Aa {/}
****
LOADING
Cc
01010
0010
SSL
A
B
ERROR
CSS
CSS
WWW
API
JAVA
ADS
YES NO
00101101
01010011
00101101
HTML

Alternative resources

Here’s an assortment of alternative resources whose style fits that of this template:

Resources

Did you like the resources in this template? Get them for free at our other websites:

Photos:

- Medium shot man working on laptop
- Group of friends planning a trip in a cafe
- Medium shot man working on laptop
- Side view of men working on laptops at the office
- Lifestyle of woman in the office
- Secretary working on laptop

Vectors:

- Abstract gradient circuit board background

Icon pack:

- Icon Pack: Coding | Lineal

Agenda

01

Apresentação do curso

You can describe the topic
of the section here

02

O que é IA, ML e DL?

You can describe the topic
of the section here

03

Tips

You can describe the topic
of the section here

04

Problemáticas atuais

You can describe the topic
of the section here

Contents of this template

You can delete this slide when you're done editing the presentation

Fonts	To view this template correctly in PowerPoint, download and install the fonts we used
Used and alternative resources	An assortment of graphic resources that are suitable for use in this presentation
Thanks slide	You must keep it so that proper credits for our design are given
Colors	All the colors used in this presentation
Icons and infographic resources	These can be used in the template, and their size and color can be edited
Editable presentation theme	You can edit the master slides easily. For more info, click here

Instructions for use

If you have a free account, in order to use this template, you must credit Slidesgo by keeping the Thanks slide. Please refer to the next slide to read the instructions for premium users.

As a Free user, you are allowed to:

- Modify this template.
- Use it for both personal and commercial projects.

You are not allowed to:

- Sublicense, sell or rent any of Slidesgo Content (or a modified version of Slidesgo Content).
- Distribute Slidesgo Content unless it has been expressly authorized by Slidesgo.
- Include Slidesgo Content in an online or offline database or file.
- Offer Slidesgo templates (or modified versions of Slidesgo templates) for download.
- Acquire the copyright of Slidesgo Content.

For more information about editing slides, please read our FAQs or visit our blog:
https://slidesgo.com/faqs and https://slidesgo.com/slidesgo-school

Instructions for use (premium users)

As a Premium user, you can use this template without attributing Slidesgo or keeping the Thanks slide.

You are allowed to:

- Modify this template.
- Use it for both personal and commercial purposes.
- Hide or delete the “Thanks” slide and the mention to Slidesgo in the credits.
- Share this template in an editable format with people who are not part of your team.

You are not allowed to:

- Sublicense, sell or rent this Slidesgo Template (or a modified version of this Slidesgo Template).
- Distribute this Slidesgo Template (or a modified version of this Slidesgo Template) or include it in a database or in any other product or service that offers downloadable images, icons or presentations that may be subject to distribution or resale.
- Use any of the elements that are part of this Slidesgo Template in an isolated and separated way from this Template.
- Register any of the elements that are part of this template as a trademark or logo, or register it as a work in an intellectual property registry or similar.

For more information about editing slides, please read our FAQs or visit our blog:
https://slidesgo.com/faqs and https://slidesgo.com/slidesgo-school

Fonts & colors used

This presentation has been made using the following fonts:

IBM Plex Mono Bold
(https://fonts.google.com/specimen/IBM+Plex+Mono)

Poppins Normal & Bold
(https://fonts.google.com/specimen/Poppins)

#f5f8ff

#bfbef7

#0c0a9e

#8208d5

#1d1d1d

#eb9109

Storyset

Create your Story with our illustrated concepts. Choose the style you like the most, edit its colors, pick the background and layers you want to show and bring them to life with the animator panel! It will boost your presentation. Check out **how it works**.

Pana

Amico

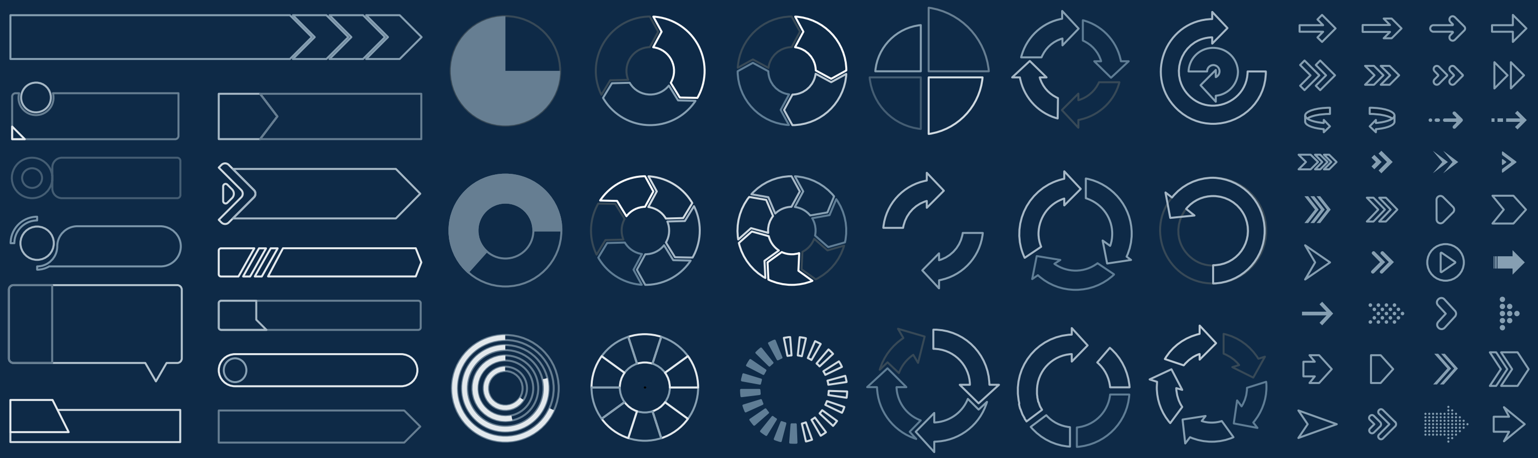
Bro

Rafiki

Cuate

Use our editable graphic resources...

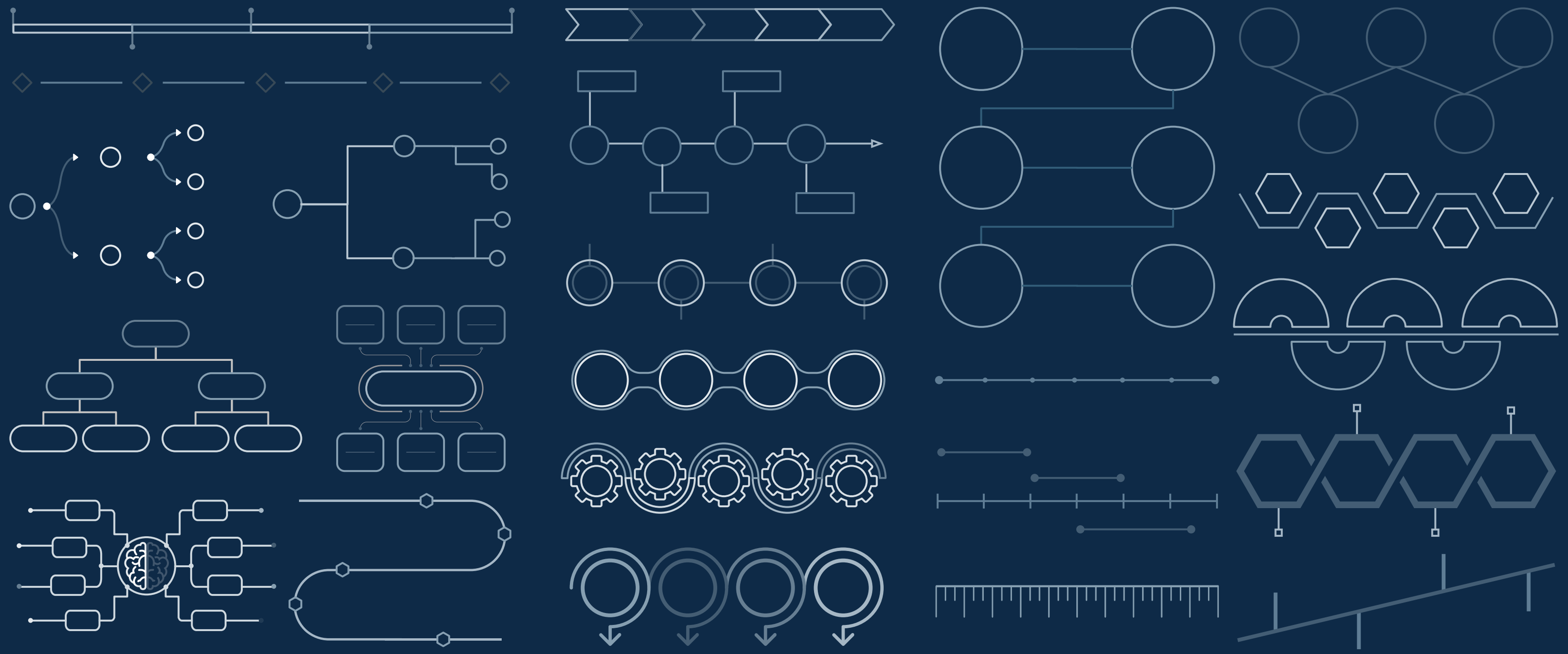
You can easily **resize** these resources without losing quality. To **change the color**, just ungroup the resource and click on the object you want to change. Then, click on the paint bucket and select the color you want. Group the resource again when you're done. You can also look for more **infographics** on Slidesgo.

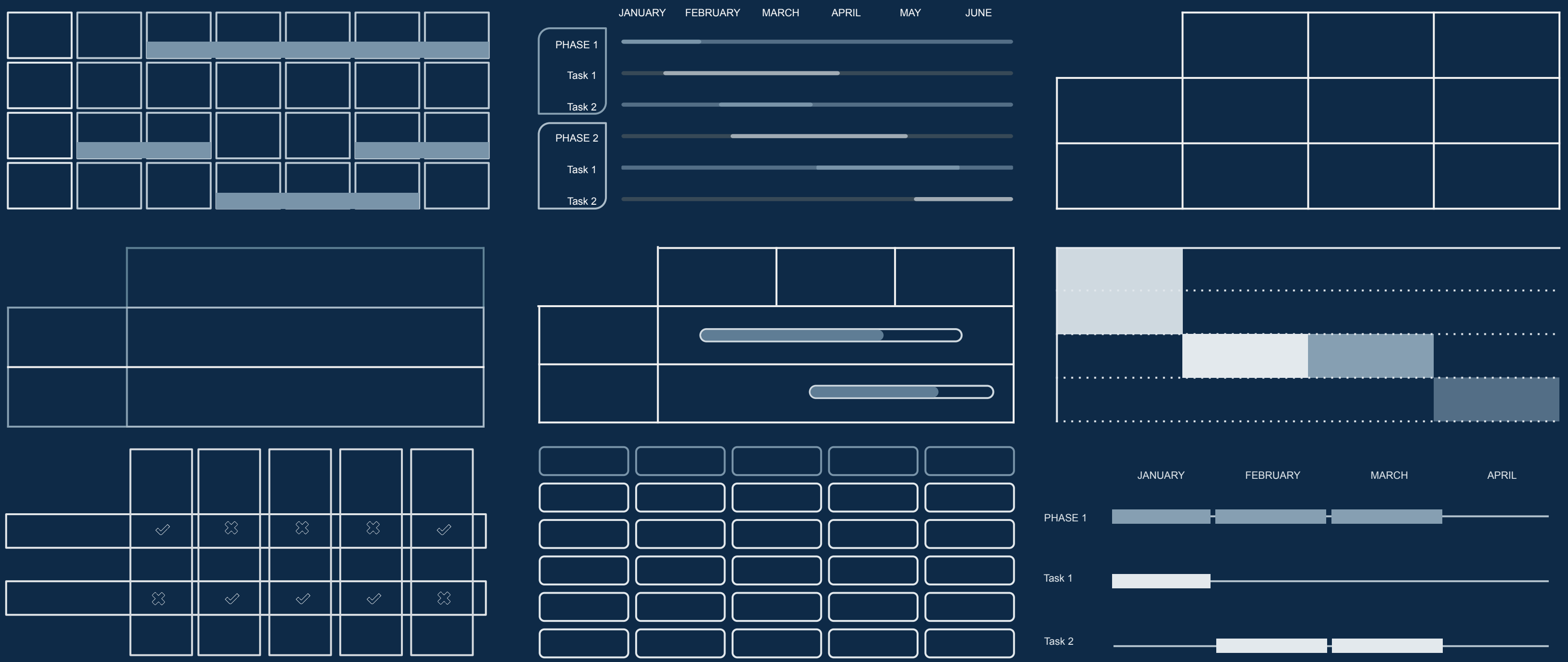


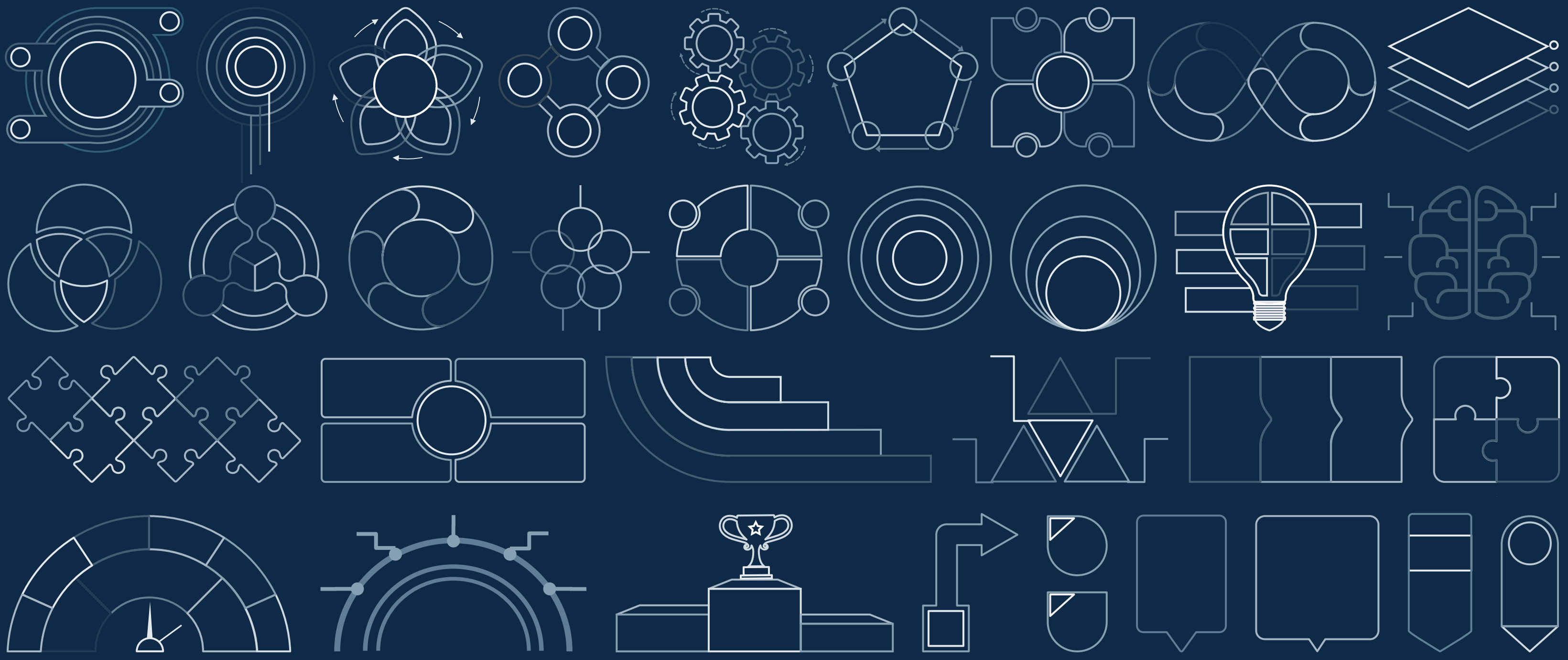


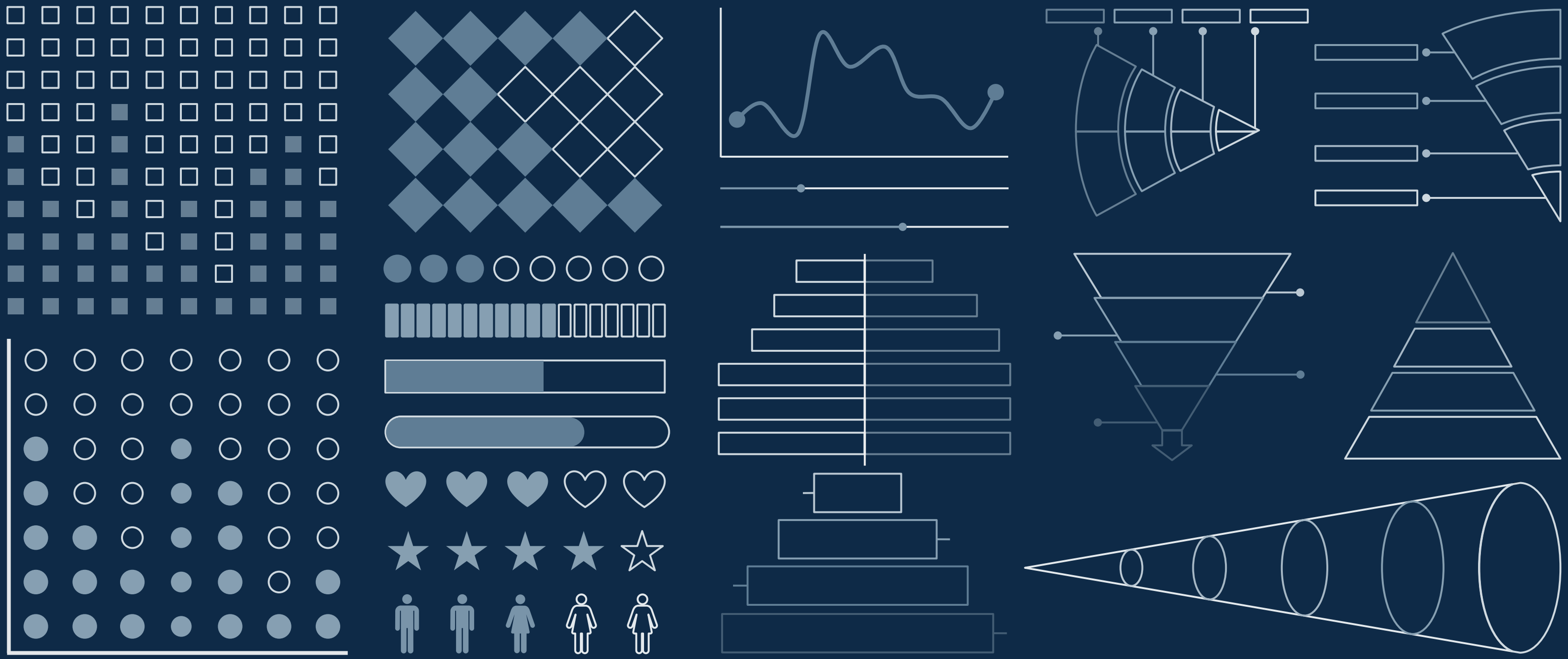







JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
PHASE 1
Task 1
Task 2
PHASE 2
Task 1
Task 2
JANUARY
FEBRUARY
MARCH
APRIL
PHASE 1
Task 1
Task 2





...and our sets of editable icons

You can **resize** these icons without losing quality.
You can **change the stroke and fill color**; just select the icon and click on the **paint bucket/pen**.
In Google Slides, you can also use **Flaticon's extension**, allowing you to customize and add even more icons.

Educational Icons

Medical Icons

Business Icons

Teamwork Icons

Help & Support Icons

Avatar Icons

Creative Process Icons

Performing Arts Icons

Nature Icons

SEO & Marketing Icons