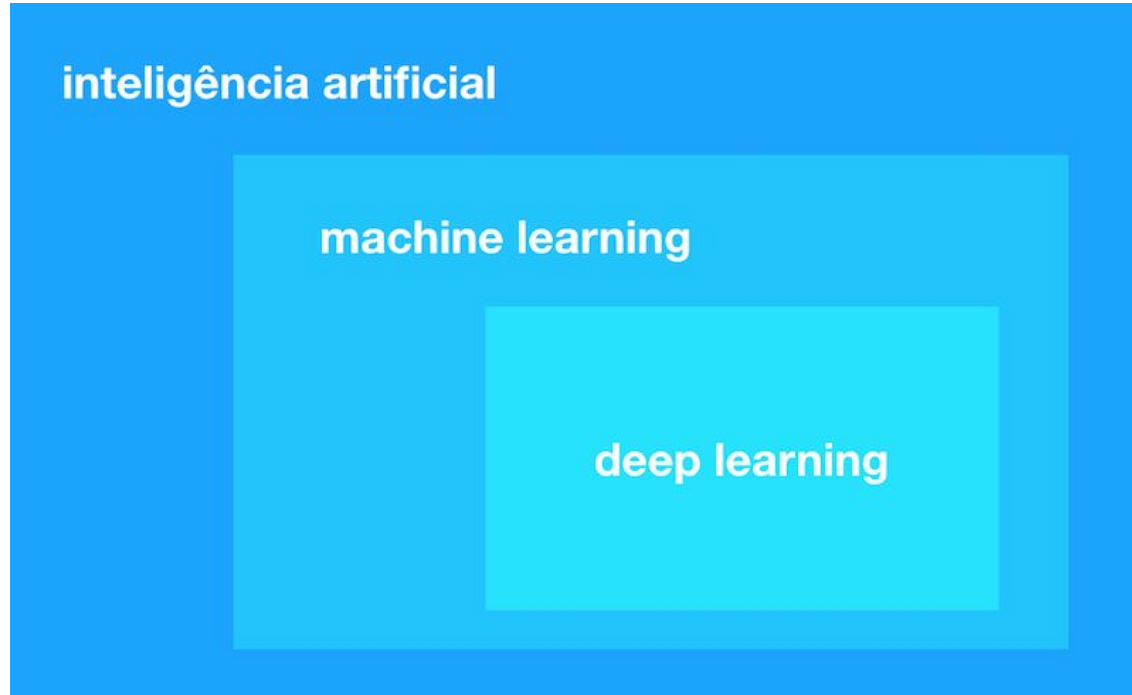


# Introdução - IA

# Inteligência Artificial? Machine Learning?



# Inteligência Artificial? Machine Learning?

- Inteligência Artificial
  - A Inteligência Artificial é uma subárea da Ciência da Computação, relacionada ao estudo de agentes racionais.

# Inteligência Artificial? Machine Learning?

- Inteligência Artificial

- A Inteligência Artificial é uma subárea da Ciência da Computação, relacionada ao estudo de agentes racionais.

- Machine Learning

- Machine Learning é uma das áreas de estudo em IA, um dos grupos de agentes em IA tem a capacidade de se tornar mais competente com o aprendizado sobre o ambiente em que ele atua.

# Inteligência Artificial? Machine Learning?

- Inteligência Artificial

- A Inteligência Artificial é uma subárea da Ciência da Computação, relacionada ao estudo de agentes racionais.

- Machine Learning

- Machine Learning é uma das áreas de estudo em IA, um dos grupos de agentes em IA tem a capacidade de se tornar mais competente com o aprendizado sobre o ambiente em que ele atua.

- Deep Learning

- Qualquer rede neural com mais de uma camada escondida em sua arquitetura.

# Dado - Passado e presente

- Passado
  - Dado em menor volume;
  - Registro manual;
  - Baixa qualidade.

# Dado - Passado e presente

- Passado

- Dado em menor volume;
- Registro manual;
- Baixa qualidade.

- Presente

- Big Data;
- Registro automático;
- Processos para garantir a qualidade.

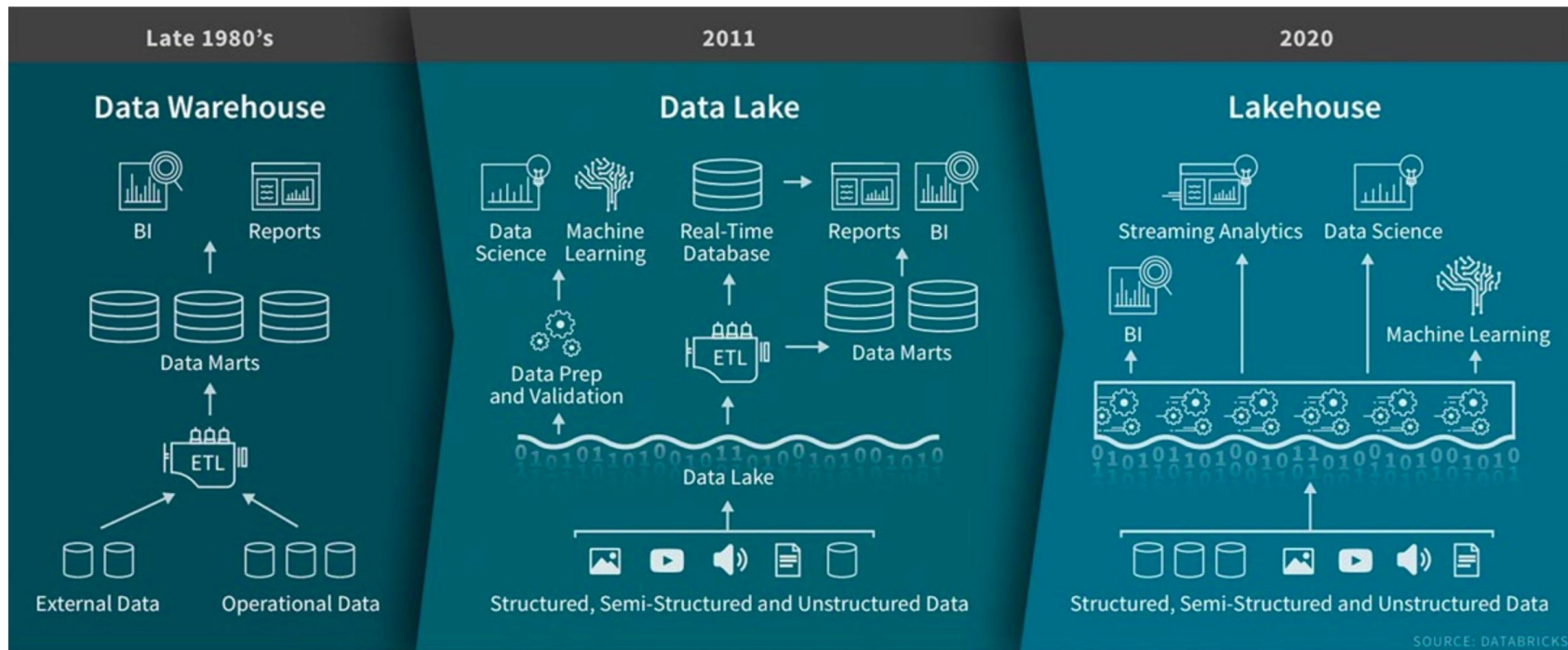
# Big Data - Qual é o interesse?

“Desde o princípio do tempo até o ano 2003, a humanidade criou cinco exabytes de informações digitais. Um exabyte é um bilhão de gigabytes - ou um 1 seguido de dezoito zeros. Enquanto escrevo este livro, no ano de 2010, a espécie humana está gerando cinco exabytes de informações a cada dois dias. No ano 2013, o número será de cinco exabytes produzidos a cada dez minutos.”

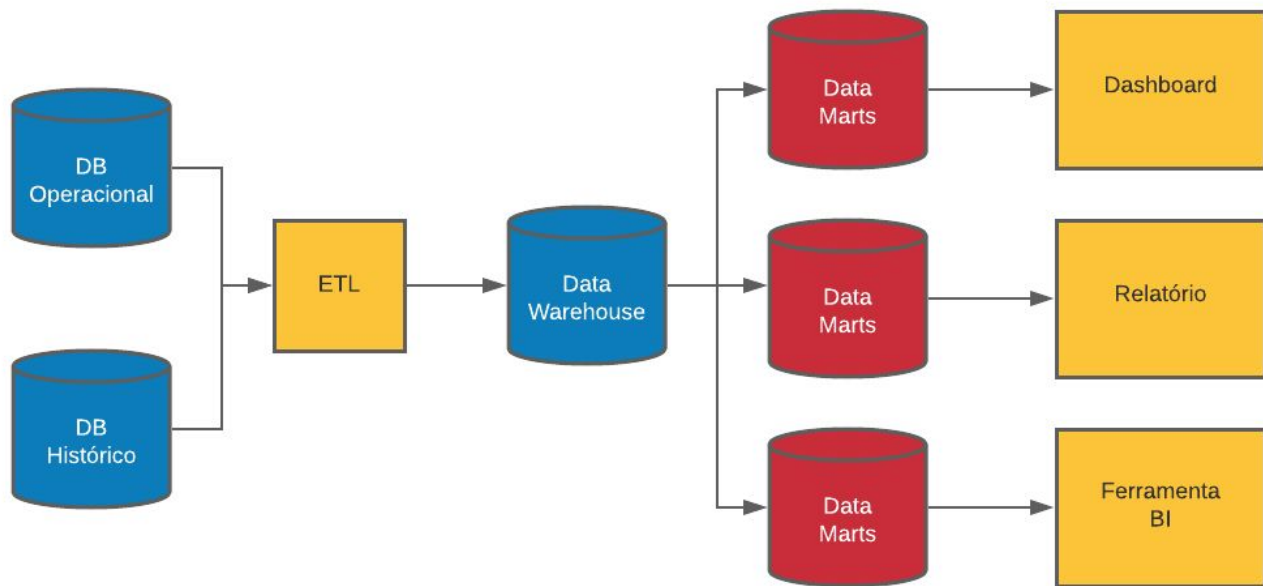
Abundância: O futuro é melhor do que você imagina; Peter H. Diamandis, Steven Kotler.



# Dados - Arquiteturas



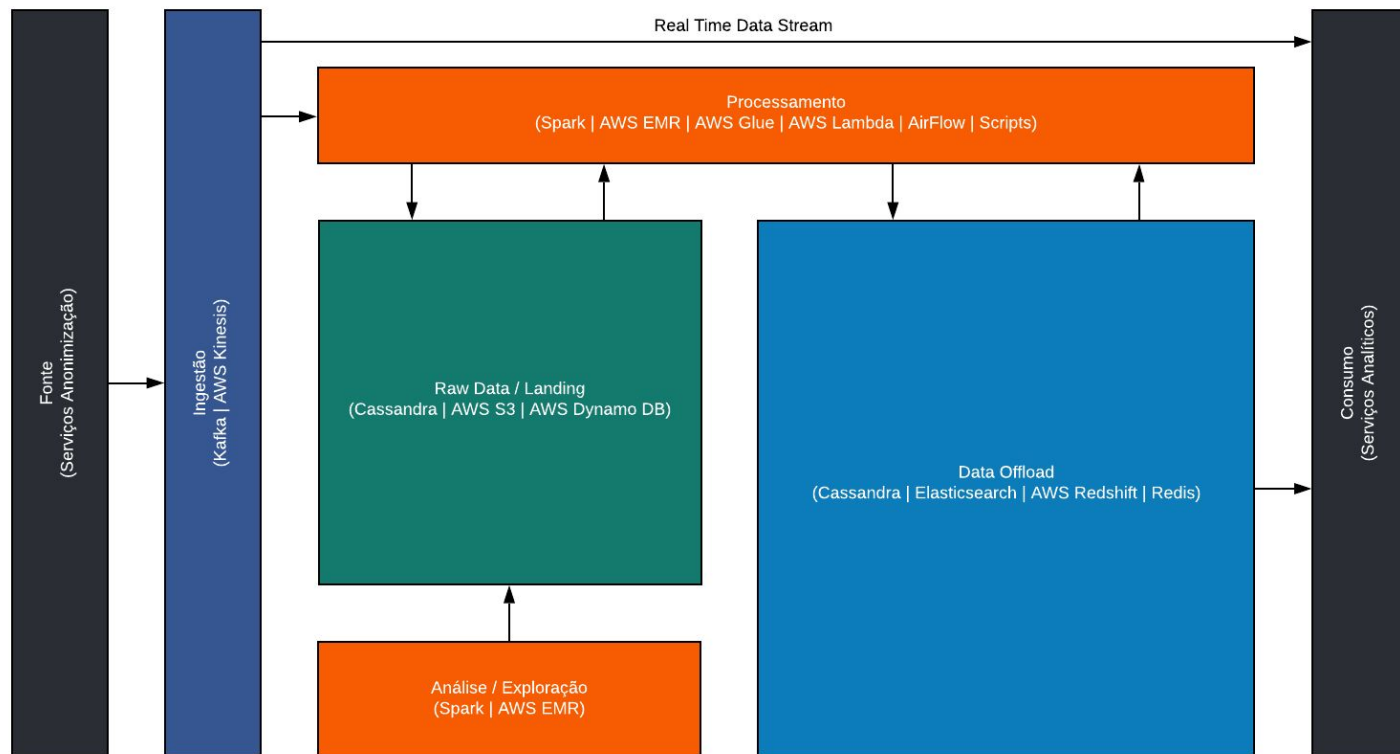
# Dados - Data Warehouse



# Dados - Data Lake



# Dados - Data Lake



# Data Flow

Tem origem nos conceitos iniciais das pesquisas que originaram as estruturas conhecidas como **Redes Neurais** realizados por **Von Newman**.



# Data Flow

Tem origem nos conceitos iniciais das pesquisas que originaram as estruturas conhecidas como **Redes Neurais** realizados por **Von Newman**.

Representado por um **diagrama de fluxo de dados (DFD)**.





# Data Flow

Tem origem nos conceitos iniciais das pesquisas que originaram as estruturas conhecidas como **Redes Neurais** realizados por **Von Newman**.

Representado por um **diagrama de fluxo de dados (DFD)**.

Um **DFD** é também utilizado para visualizar o processamento de dados dentro de um sistema.



# Data Flow

Imagine vários processos sendo utilizados para calcular o valor da hipotenusa de um triângulo retângulo.

$$a = \sqrt{b^2 + c^2}$$



# Data Flow

Qual é o desafio?

# Data Flow

Qual é o desafio?

Para que o computador tenha o valor de ***a***, antes é necessário calcular a soma dos valores de ***b*** e ***c*** elevados ao quadrado, computando por final a raiz do resultado desta soma.

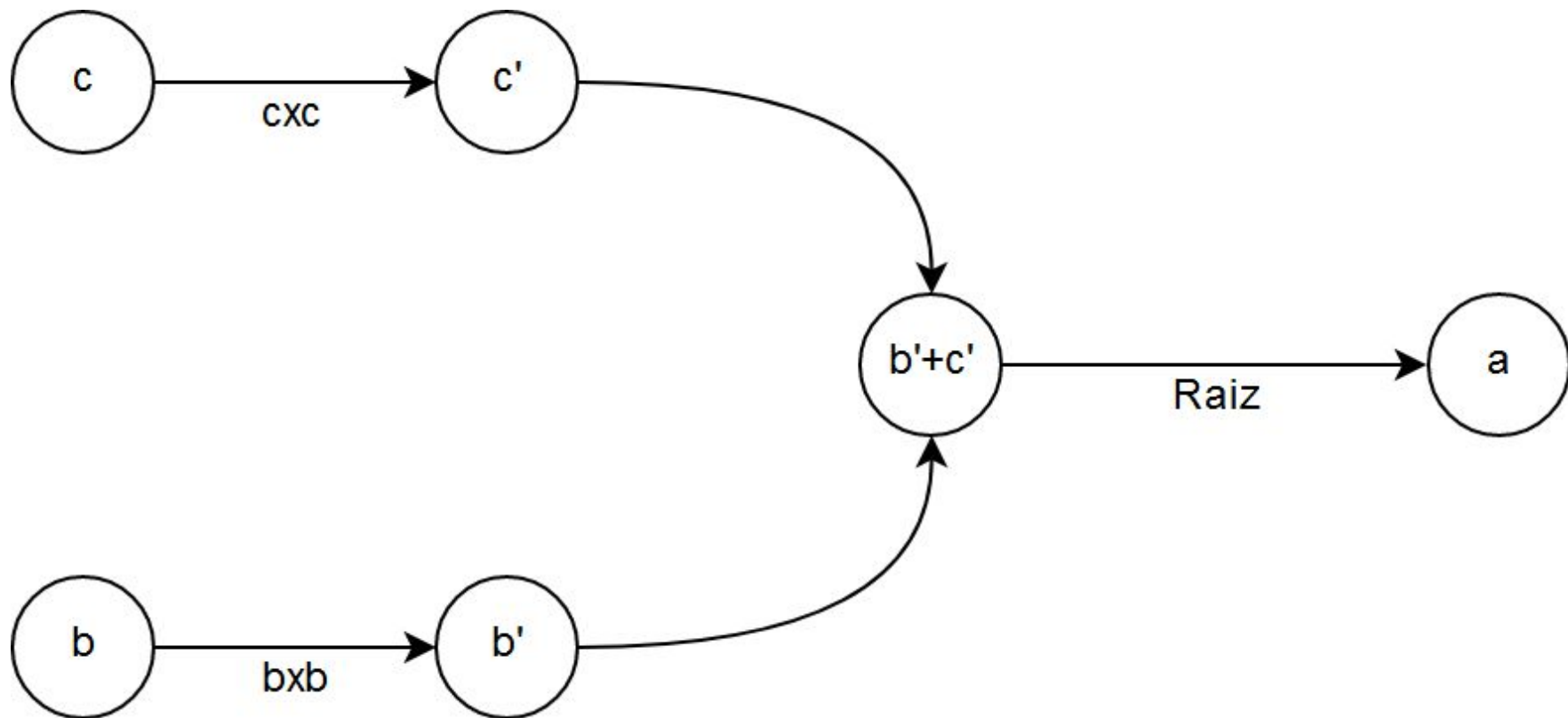
# Data Flow

Qual é o desafio?

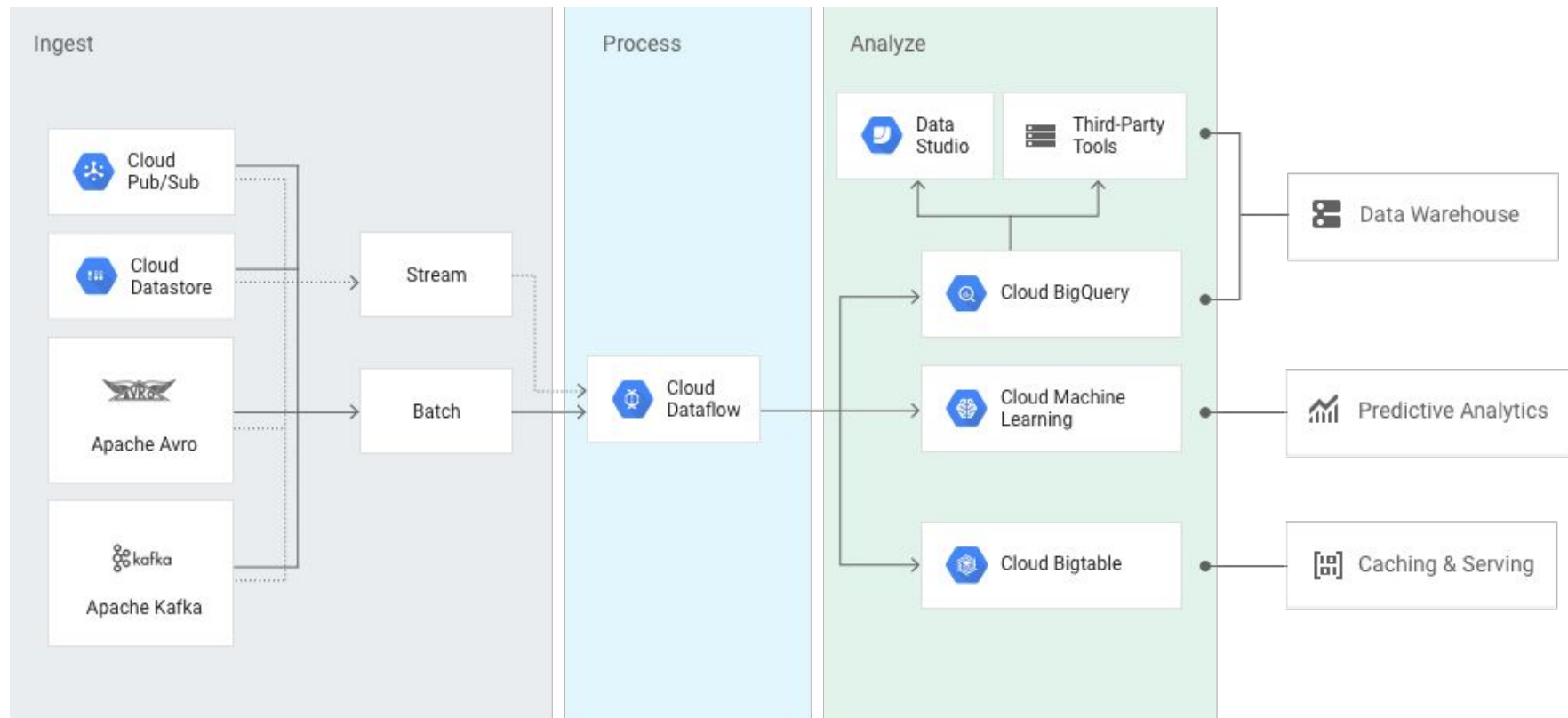
Para que o computador tenha o valor de  $a$ , antes é necessário calcular a soma dos valores de  $b$  e  $c$  elevados ao quadrado, computando por final a raiz do resultado desta soma.

Precisamos ter controle suficiente para garantir que uma etapa não seja iniciada sem que suas etapas dependentes sejam concluídas.

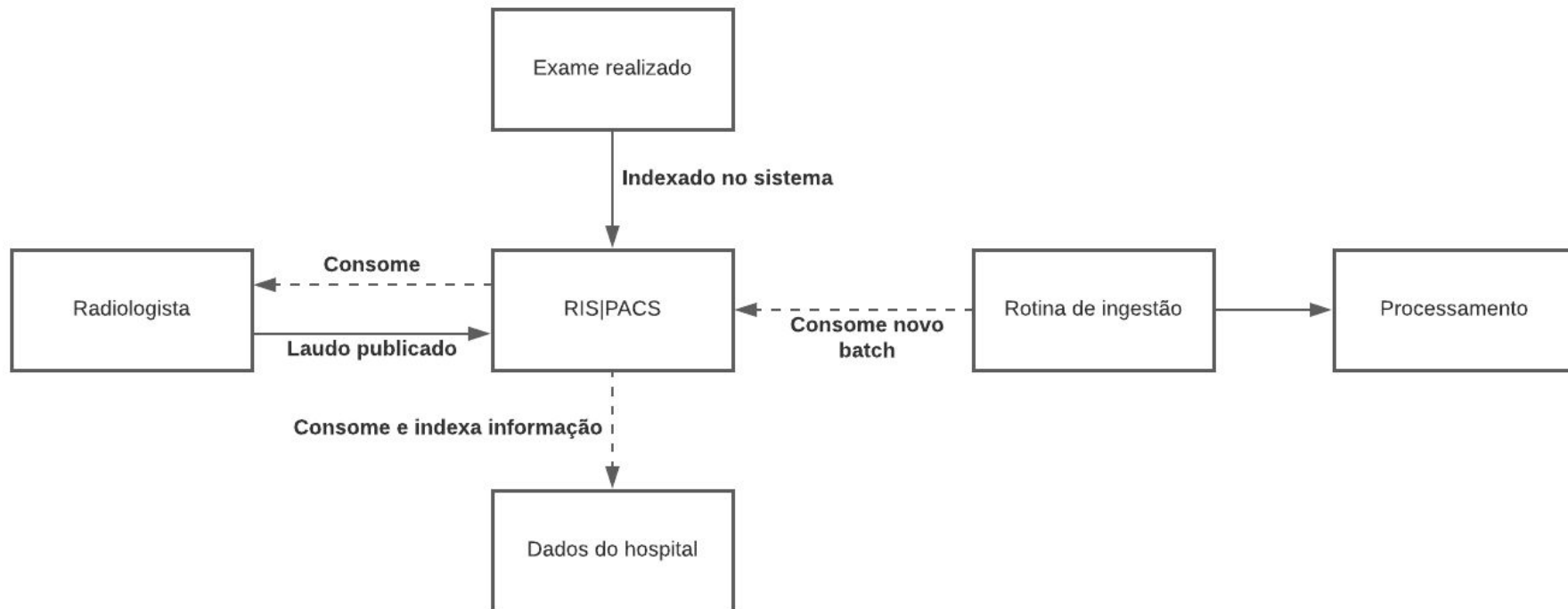
## Data Flow



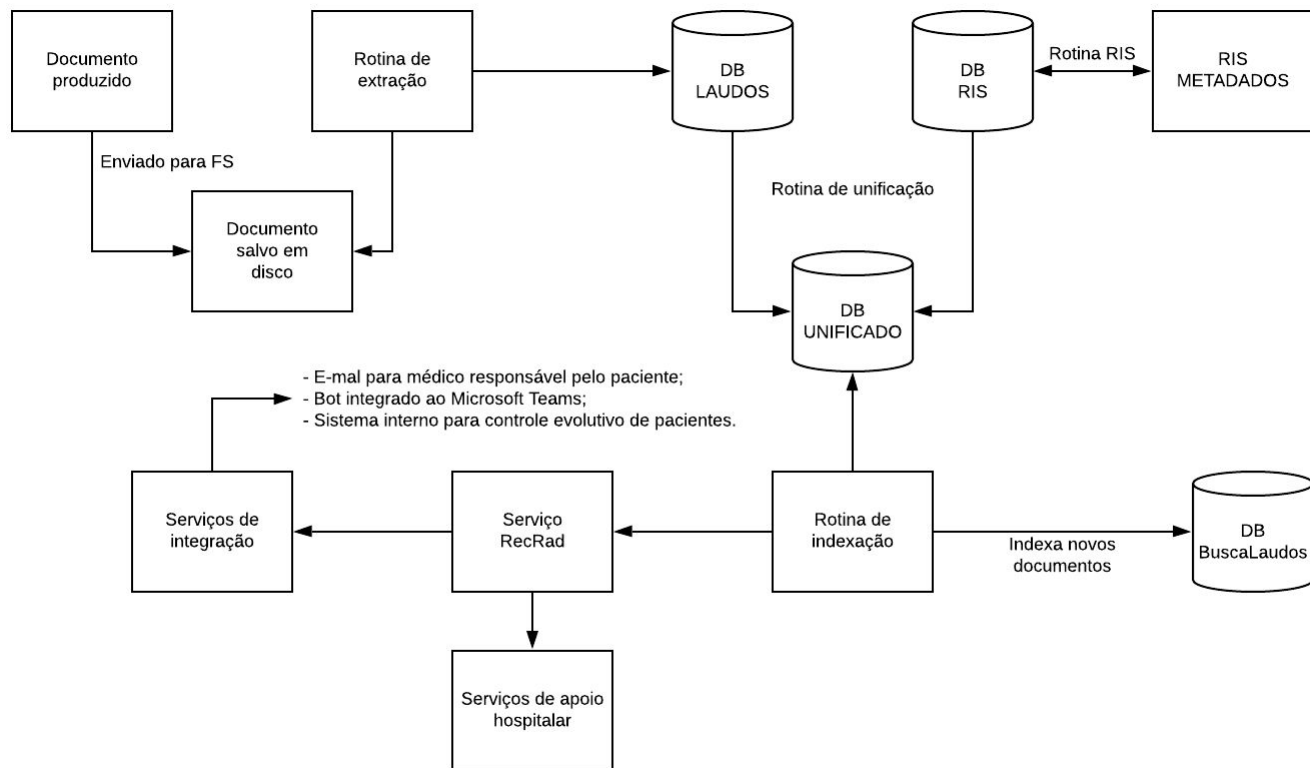
# Data Flow - Sistemas de ingestão de dados



# Data Flow - Estudo de caso - NLP



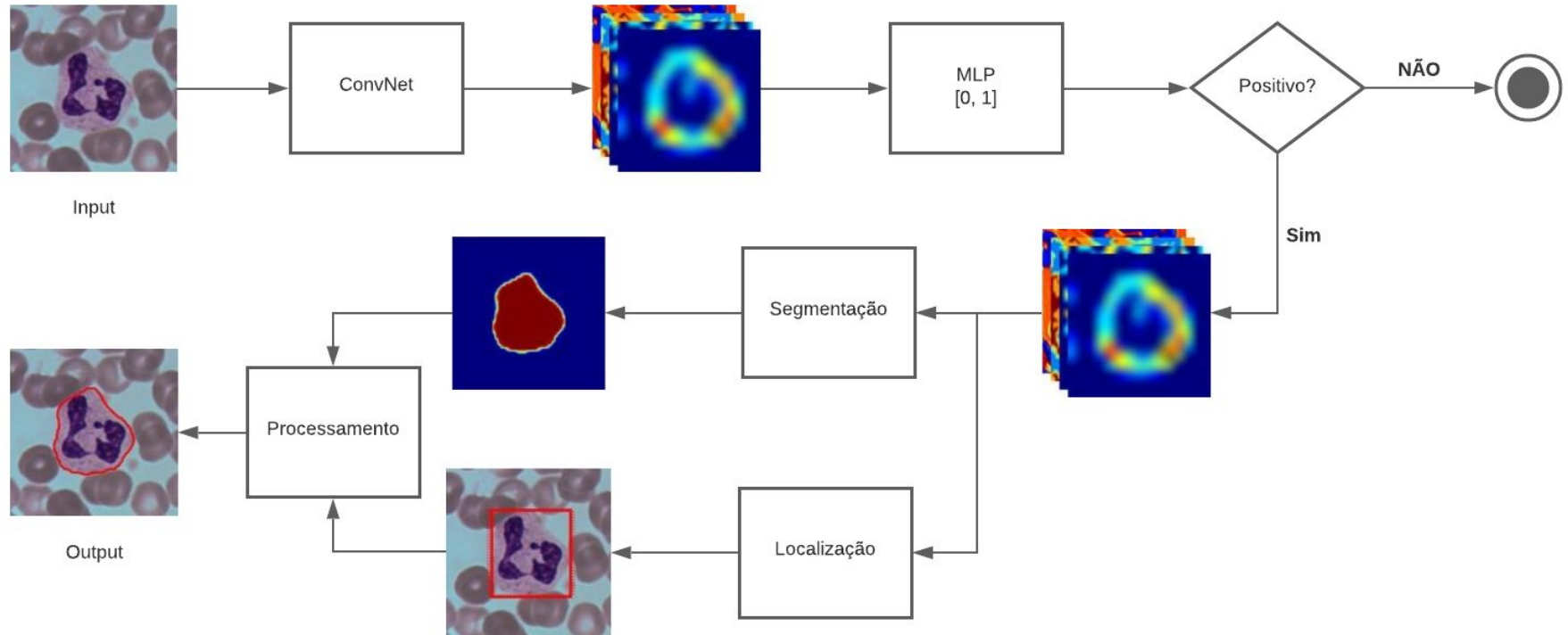
# Data Flow - Estudo de caso - NLP







# Data Flow - Estudo de caso - Loc + Mask Leucócitos



# Data Flow - Sistemas de Inteligência Artificial

Uma inteligência artificial é uma sequência de processamento de dados, que toma decisões através do resultado de modelos de aprendizado de máquina.

# Computação Cognitiva

## Conceito

- A computação cognitiva estende a computação básica, permitindo a máquina inferir e raciocinar, dado um objetivo;

# Computação Cognitiva

## Conceito

- A computação cognitiva estende a computação básica, permitindo a máquina inferir e raciocinar, dado um objetivo;
- Capaz de lidar com problemas humanos de forma natural (ex. agentes de suporte por voz);

# Computação Cognitiva

## Conceito

- A computação cognitiva estende a computação básica, permitindo a máquina inferir e raciocinar, dado um objetivo;
- Capaz de lidar com problemas humanos de forma natural (ex. agentes de suporte por voz);
- Através da análise do contexto, obter a melhor resposta ao invés da resposta correta (ex. sistemas especialistas de busca).

# Computação Cognitiva - Sistemas devem ser

## Adaptativos

- Um sistema deve ter a capacidade de se **adaptar constantemente** junto aos dados em que é exposto, através da análise em tempo real dos dados.

# Computação Cognitiva - Sistemas devem ser

## Adaptativos

- Um sistema deve ter a capacidade de se **adaptar constantemente** junto aos dados em que é exposto, através da análise em tempo real dos dados.

## Interativos

- Um sistema deve se comunicar naturalmente com os usuários, facilitando a interação entre a **máquina** e o **humano**.

# Computação Cognitiva - Sistemas devem ser

## Iterativos

- Um sistema deve ter a capacidade de resolver problemas como ambiguidade ou falta de informação, realizando um conjunto de perguntas para o usuário, assim aumentando o domínio sobre o problema em que se encontra.



# Computação Cognitiva - Sistemas devem ser

## Iterativos

- Um sistema deve ter a capacidade de resolver problemas como ambiguidade ou falta de informação, realizando um conjunto de perguntas para o usuário, assim aumentando o domínio sobre o problema em que se encontra.

## Contextual

- Um sistema deve ter a capacidade de utilizar-se do contexto atual do diálogo para ajudar na identificação da melhor resposta para o problema proposto.

# Computação Cognitiva - Sistemas devem ser

Como aprendemos?

# Computação Cognitiva - Sistemas devem ser

## Como aprendemos?

- Aprendemos de forma muito similar a probabilidade bayesiana.
  - Partimos de uma hipótese;
  - Estabelecemos nossos intervalos de confiança junto ao número de evidências que sustentam nossa hipótese.

# Computação Cognitiva - Sistemas devem ser

## Como aprendemos?

- Aprendemos de forma muito similar a probabilidade bayesiana.
  - Partimos de uma hipótese;
  - Estabelecemos nossos intervalos de confiança junto ao número de evidências que sustentam nossa hipótese.
- Uma criança, enquanto brinca e explora as coisas que, para ela são novidade, cria hipóteses de como o mundo a sua volta se comporta;

# Computação Cognitiva - Sistemas devem ser

## Como aprendemos?

- Aprendemos de forma muito similar a probabilidade bayesiana.
  - Partimos de uma hipótese;
  - Estabelecemos nossos intervalos de confiança junto ao número de evidências que sustentam nossa hipótese.
- Uma criança, enquanto brinca e explora as coisas que, para ela são novidade, cria hipóteses de como o mundo a sua volta se comporta;
- Realiza testes e verifica se suas hipóteses são corretas;

# Computação Cognitiva - Sistemas devem ser

## Como aprendemos?

- Aprendemos de forma muito similar a probabilidade bayesiana.
  - Partimos de uma hipótese;
  - Estabelecemos nossos intervalos de confiança junto ao número de evidências que sustentam nossa hipótese.
- Uma criança, enquanto brinca e explora as coisas que, para ela são novidade, cria hipóteses de como o mundo a sua volta se comporta;
- Realiza testes e verifica se suas hipóteses são corretas;
- Ideias são formuladas e verificadas a cada novo questionamento, brincadeira ou descoberta;

# Computação Cognitiva - Sistemas devem ser

## Como aprendemos?

- Aprendemos de forma muito similar a probabilidade bayesiana.
  - Partimos de uma hipótese;
  - Estabelecemos nossos intervalos de confiança junto ao número de evidências que sustentam nossa hipótese.
- Uma criança, enquanto brinca e explora as coisas que, para ela são novidade, cria hipóteses de como o mundo a sua volta se comporta;
- Realiza testes e verifica se suas hipóteses são corretas;
- Ideias são formuladas e verificadas a cada novo questionamento, brincadeira ou descoberta;
- Desta forma, constrói seus **modelos mentais** de como o mundo funciona;

# Computação Cognitiva - Sistemas devem ser

## Como aprendemos?

- Aprendemos de forma muito similar a probabilidade bayesiana.
  - Partimos de uma hipótese;
  - Estabelecemos nossos intervalos de confiança junto ao número de evidências que sustentam nossa hipótese.
- Uma criança, enquanto brinca e explora as coisas que, para ela são novidade, cria hipóteses de como o mundo a sua volta se comporta;
- Realiza testes e verifica se suas hipóteses são corretas;
- Ideias são formuladas e verificadas a cada novo questionamento, brincadeira ou descoberta;
- Desta forma, constrói seus **modelos mentais** de como o mundo funciona;
- Da mesma maneira, fazemos máquinas aprenderem.



# Tipos de aprendizado

## Supervisionado

- Treinamos o computador com exemplos de dados de entrada e sua saída esperada.
  - **Ex.** Treinamos um modelo para prever o custo de determinado paciente dado seu perfil. Para isso, treinamos o modelo com nossos dados históricos.

# Tipos de aprendizado

## Não supervisionado

- Treinamos o computador sem fornecer nenhum tipo de informação de saída. Esperamos que o computador encontre padrões nos dados fornecidos
  - **Ex.** Esperamos que um modelo encontre dois clusters de pixels dominantes em uma imagem de ressonância magnética.

# Tipos de aprendizado

## Por reforço

- Modelo é exposto ao ambiente, seja ele real ou simulado. Espera-se que o modelo execute alguma função e fornecemos um resultado ao modelo no final de cada ação.
  - **Ex.** Esperamos que o modelo aprenda a melhor forma de distribuir demandas de pacientes no pronto socorro aos médicos seguindo uma regra de negócio.

# Tipos de aprendizado - Supervisionado

input	1	2	3	4	5	6	7
output	1	4	9	16	25	36	49



# Tipos de aprendizado - Não supervisionado



# Tipos de aprendizado - Por reforço



# Machine Learning - Além do csv...

## MNIST Database

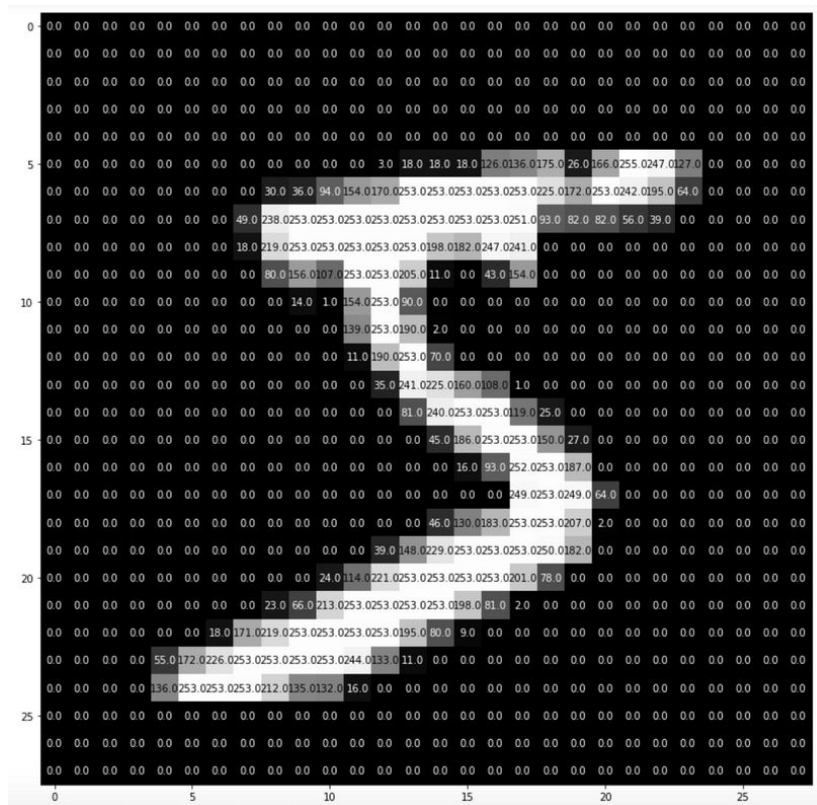
Um grande dataset composto por imagens (28x28)px de números manuscritos de 0 até 9. Disponibiliza 60 mil amostras e é muito utilizado para teste e introdução em desenvolvimento de modelos para classificação e processamento de imagem.



# Machine Learning - Além do csv...

Uma imagem monocromática, como no exemplo ao lado, tem a estrutura de informação de uma matriz ( $i$  vs  $j$ );

Cada posição dessa matriz guarda um valor num range de 256 possibilidades (0 ~ 255) onde 0 é a cor mais escura da imagem e 255 a cor mais clara.

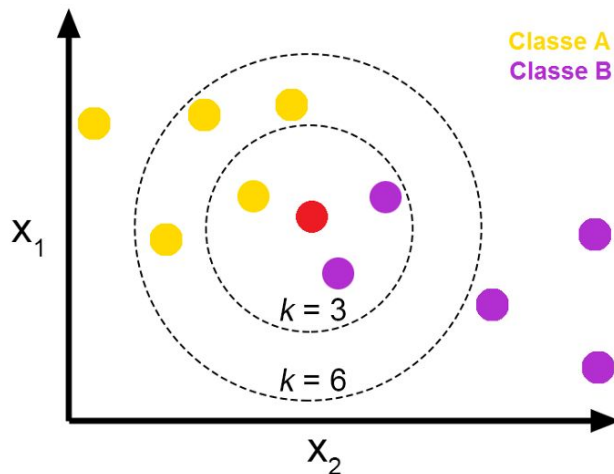




# Machine Learning - KNN

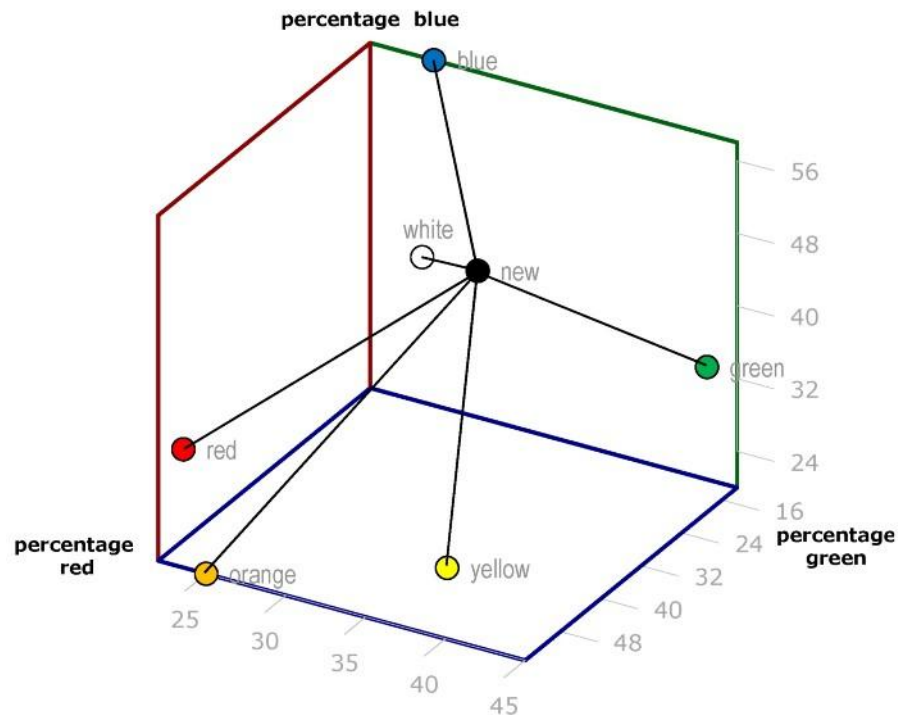
## KNN (K-Nearest Neighbors)

Um dos algoritmos clássicos em machine learning (aprendizagem supervisionada), faz uso de um conjunto de pontos conhecidos para classificar um novo ponto desconhecido.



# Machine Learning - KNN

1. Recebe um novo ponto desconhecido;
2. Mede a distância deste novo ponto em relação a todos os pontos conhecidos;
3. Ordena as menores K distâncias encontradas;
4. Destes K menores pontos conhecidos, contabiliza as classes obtidas;
5. Classifica para este ponto desconhecido, a classe com maior presença nessas K classes de menor distância.



# Atividade

Clique [aqui](#) para abrir a revisão

Clique [aqui](#) para abrir a atividade

# Otimização

# Otimização

Em um problema de otimização, busca-se encontrar a melhor solução entre todas as soluções viáveis.

# Otimização

Em um problema de otimização, busca-se encontrar a melhor solução entre todas as soluções viáveis.

Essa busca pela melhor solução pode levar a um problema dito como NP (tempo polinomial não determinístico).

# Otimização

Em um problema de otimização, busca-se encontrar a melhor solução entre todas as soluções viáveis.

Essa busca pela melhor solução pode levar a um problema dito como NP (tempo polinomial não determinístico).

Imagine calcular qual é a melhor forma de organizar torcedores num estádio de futebol, levando em consideração algumas variáveis como casais, torcidas, amigos. Quantas possibilidades de combinações não temos para este problema?

# Otimização

Em um problema de otimização, busca-se encontrar a melhor solução entre todas as soluções viáveis.

Essa busca pela melhor solução pode levar a um problema dito como NP (tempo polinomial não determinístico).

Imagine calcular qual é a melhor forma de organizar torcedores num estádio de futebol, levando em consideração algumas variáveis como casais, torcidas, amigos. Quantas possibilidades de combinações não temos para este problema?

Para uma máquina calcular todas essas possibilidades através de força bruta, o resultado poderia demorar anos para ser obtido.



# Otimização

Para estes problemas, que geralmente envolvem análise combinatória, uma estratégia é utilizar métodos heurísticos.

# Otimização

Para estes problemas, que geralmente envolvem análise combinatória, uma estratégia é utilizar métodos heurísticos.

Meta-heurísticas são aplicadas para a resolução de problemas onde não se conhece um algoritmo eficiente.

# Otimização

Para estes problemas, que geralmente envolvem análise combinatória, uma estratégia é utilizar métodos heurísticos.

Meta-heurísticas são aplicadas para a resolução de problemas onde não se conhece um algoritmo eficiente.

Utilizando de aleatoriedade e de conhecimento histórico para encontrar uma próxima solução melhor ou igual a obtida anteriormente.

# Otimização

Para estes problemas, que geralmente envolvem análise combinatória, uma estratégia é utilizar métodos heurísticos.

Meta-heurísticas são aplicadas para a resolução de problemas onde não se conhece um algoritmo eficiente.

Utilizando de aleatoriedade e de conhecimento histórico para encontrar uma próxima solução melhor ou igual a obtida anteriormente.

Nesta aula, vamos estudar o emprego de **algoritmos genéticos** na solução de problemas de otimização.

# Algoritmo Genético

Um **algoritmo genético (AG)** é uma técnica de busca utilizada na ciência da computação para achar soluções aproximadas em problemas de otimização e busca. São uma classe particular de algoritmos evolutivos que usam técnicas inspiradas pela biologia evolutiva como hereditariedade, mutação, seleção natural e recombinação (ou *crossing over*).

[https://pt.wikipedia.org/wiki/Algoritmo\\_gen%C3%A9tico](https://pt.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico)

# Algoritmo Genético

Existe uma diversidade de seres devido aos contingentes da natureza (comida, clima, ...) e é pela lei da Seleção Natural que os seres mais adaptados ao seus ambientes sobrevivem.

# Algoritmo Genético

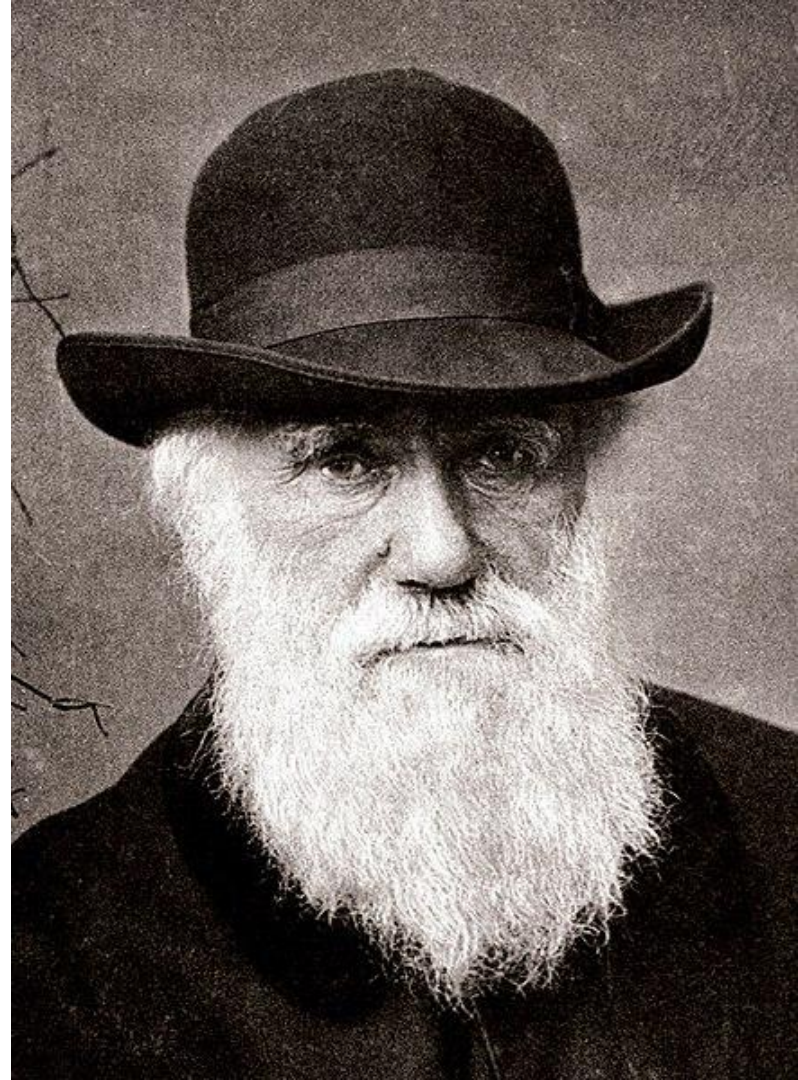
Existe uma diversidade de seres devido aos contingentes da natureza (comida, clima, ...) e é pela lei da Seleção Natural que os seres mais adaptados ao seus ambientes sobrevivem.

Os caracteres adquiridos são herdados pelas gerações seguintes.

# Algoritmo Genético

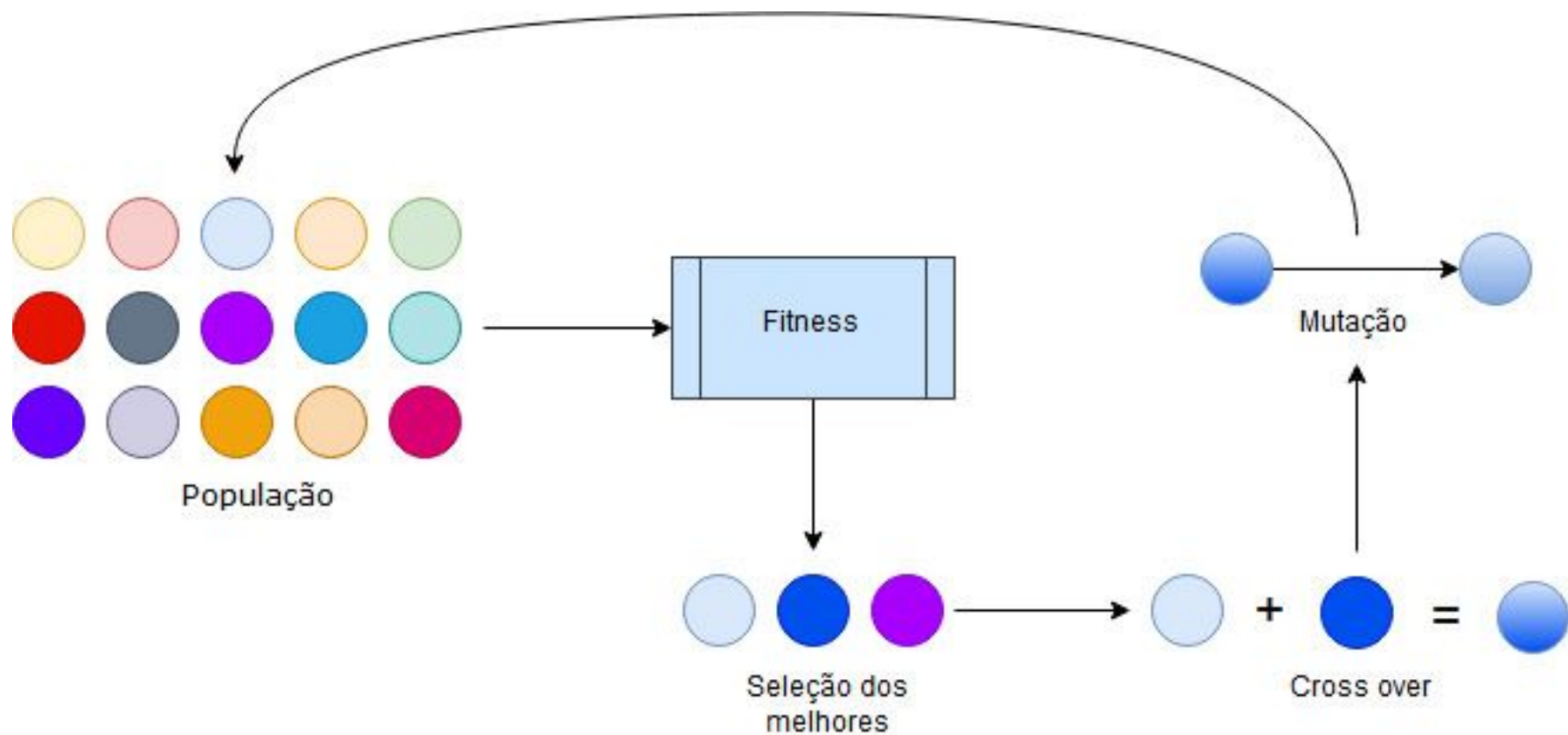
“Quanto melhor um indivíduo se adaptar ao seu meio ambiente, maior será sua chance de sobreviver e gerar descendentes.”

**Darwin, 1859**





# Algoritmo Genético



# Algoritmo Genético - Conceitos Básicos

Algoritmos Genéticos manipulam uma população de indivíduos.

# Algoritmo Genético - Conceitos Básicos

Algoritmos Genéticos manipulam uma população de indivíduos.

Indivíduos são possíveis soluções do problema.

# Algoritmo Genético - Conceitos Básicos

Algoritmos Genéticos manipulam uma população de indivíduos.

Indivíduos são possíveis soluções do problema.

Os indivíduos são combinados (crossover) uns com os outros, produzindo filhos que podem ou não sofrer mutações.

# Algoritmo Genético - Conceitos Básicos

Algoritmos Genéticos manipulam uma população de indivíduos.

Indivíduos são possíveis soluções do problema.

Os indivíduos são combinados (crossover) uns com os outros, produzindo filhos que podem ou não sofrer mutações.

As populações evoluem através de sucessivas gerações até encontrar uma solução desejável ou a solução ótima.

# Algoritmo Genético - População Inicial

Dado um problema, precisamos definir como será a representação dos indivíduos (soluções).

# Algoritmo Genético - População Inicial

Dado um problema, precisamos definir como será a representação dos indivíduos (soluções).

Um indivíduo é um **cromossomo**!

- Vetores de números reais [3; 5.22; 9; ...]
- Cadeias de bits [1, 0, 0, 1, 1, 0, ...]
- Qualquer outra estrutura de dados...

# Algoritmo Genético - População Inicial

A população inicial deve ser gerada aleatoriamente.



# Algoritmo Genético - População Inicial

A população inicial deve ser gerada aleatoriamente.

Precisamos garantir a variedade da população inicial:

- Quanto menor a variedade, menor a velocidade de convergência e vice-versa.
- Quanto maior a variedade, maior a chance de encontrar a solução ótima, porém, menor a velocidade de convergência.

# Algoritmo Genético - Fitness

A função fitness (q-function ou função de aptidão) é uma função que avalia a capacidade do indivíduo em realizar a tarefa.

# Algoritmo Genético - Fitness

A função fitness (q-function ou função de aptidão) é uma função que avalia a capacidade do indivíduo em realizar a tarefa.

É feita através de uma função que melhor representa o problema e tem por objetivo fornecer uma medida de aptidão de cada indivíduo na população corrente que irá dirigir o processo de busca por uma solução.

# Algoritmo Genético - Fitness

A função fitness (q-function ou função de aptidão) é uma função que avalia a capacidade do indivíduo em realizar a tarefa.

É feita através de uma função que melhor representa o problema e tem por objetivo fornecer uma medida de aptidão de cada indivíduo na população corrente que irá dirigir o processo de busca por uma solução.

Similar a função objetivo em problemas de otimização na matemática.

# Algoritmo Genético - Criando e Evoluindo Gerações

Quais são os métodos para criar novos indivíduos?

# Algoritmo Genético - Criando e Evoluindo Gerações

Quais são os métodos para criar novos indivíduos?

- Reprodução sexuada (crossover);

# Algoritmo Genético - Criando e Evoluindo Gerações

Quais são os métodos para criar novos indivíduos?

- Reprodução sexuada (crossover);
- Reprodução assexuada;

# Algoritmo Genético - Criando e Evoluindo Gerações

Quais são os métodos para criar novos indivíduos?

- Reprodução sexuada (crossover);
- Reprodução assexuada;
- Mutação.



# Algoritmo Genético - Criando e Evoluindo Gerações

Como garantir encontrar a solução?

# Algoritmo Genético - Criando e Evoluindo Gerações

Como garantir encontrar a solução?

- Os indivíduos devem ser selecionados (os mais aptos sobrevivem) para garantir a convergência;

# Algoritmo Genético - Criando e Evoluindo Gerações

Como garantir encontrar a solução?

- Os indivíduos devem ser selecionados (os mais aptos sobrevivem) para garantir a convergência;
- Deve-se evitar a convergência prematura;

# Algoritmo Genético - Criando e Evoluindo Gerações

Como garantir encontrar a solução?

- Os indivíduos devem ser selecionados (os mais aptos sobrevivem) para garantir a convergência;
- Deve-se evitar a convergência prematura;
- Deve-se estabelecer um critério de parada.

# Algoritmo Genético - Criando e Evoluindo Gerações

Convergência?

# Algoritmo Genético - Criando e Evoluindo Gerações

Convergência?

- Nas n últimas gerações quando não houver melhora no **score** do fitness obtido do melhor indivíduo.

# Algoritmo Genético - Reprodução

Assexuada (o mesmo que duplicação)

# Algoritmo Genético - Reprodução

Assexuada (o mesmo que duplicação)

- Indivíduos produzem clones, cópias idênticas de si próprios, ocorrendo, em geral, em organismos de baixa complexidade estrutural. Neste caso, estes organismos praticamente não apresentam variações, o que só ocorrerá quando houver mutações.



# Algoritmo Genético - Reprodução

Sexuada (crossover)

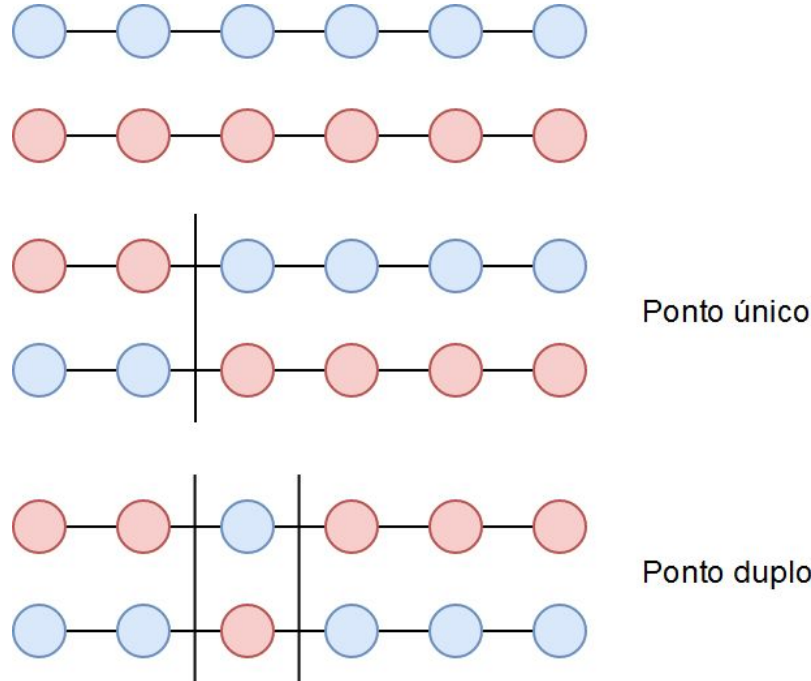
# Algoritmo Genético - Reprodução

## Sexuada (crossover)

- Indivíduos trocam material genético, podendo ou não haver mutações nos cromossomos;
- Existe uma taxa de crossover que controla a quantidade de reprodução que será feita.

# Algoritmo Genético - Reprodução

Sexuada (crossover)



# Algoritmo Genético - Reprodução

Mutação

# Algoritmo Genético - Reprodução

## Mutação

- Objetivo: Gerar diversidade entre os indivíduos para minimizar o problema dos ótimos locais;

# Algoritmo Genético - Reprodução

## Mutação

- Objetivo: Gerar diversidade entre os indivíduos para minimizar o problema dos ótimos locais;
- Existe uma taxa de mutação que diminui com o tempo para garantir a convergência (ex. porcentagem da população selecionada).

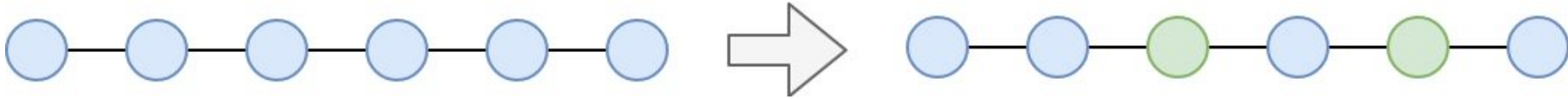
# Algoritmo Genético - Reprodução

Mutação (Tipos)

# Algoritmo Genético - Reprodução

Mutação (Tipos)

- Generativa





# Algoritmo Genético - Reprodução

## Mutação (Tipos)

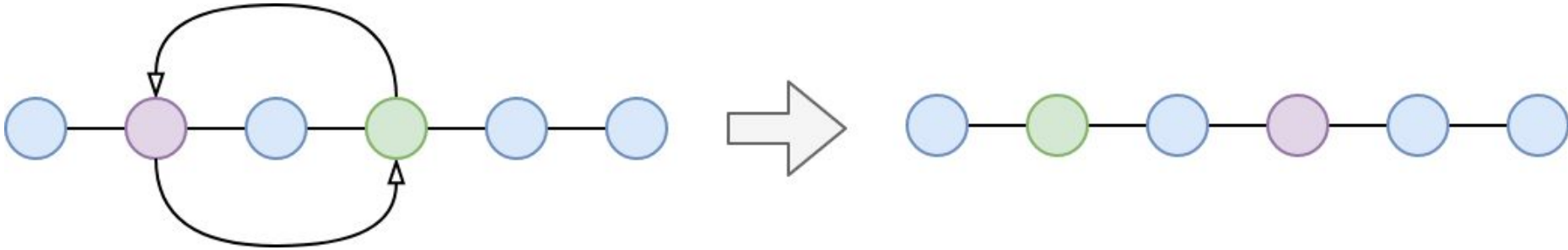
- Destrutiva



# Algoritmo Genético - Reprodução

## Mutação (Tipos)

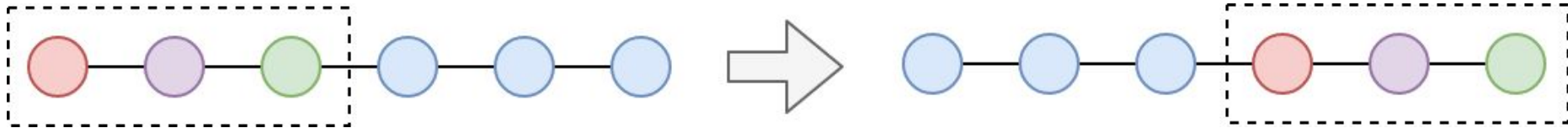
- Swap



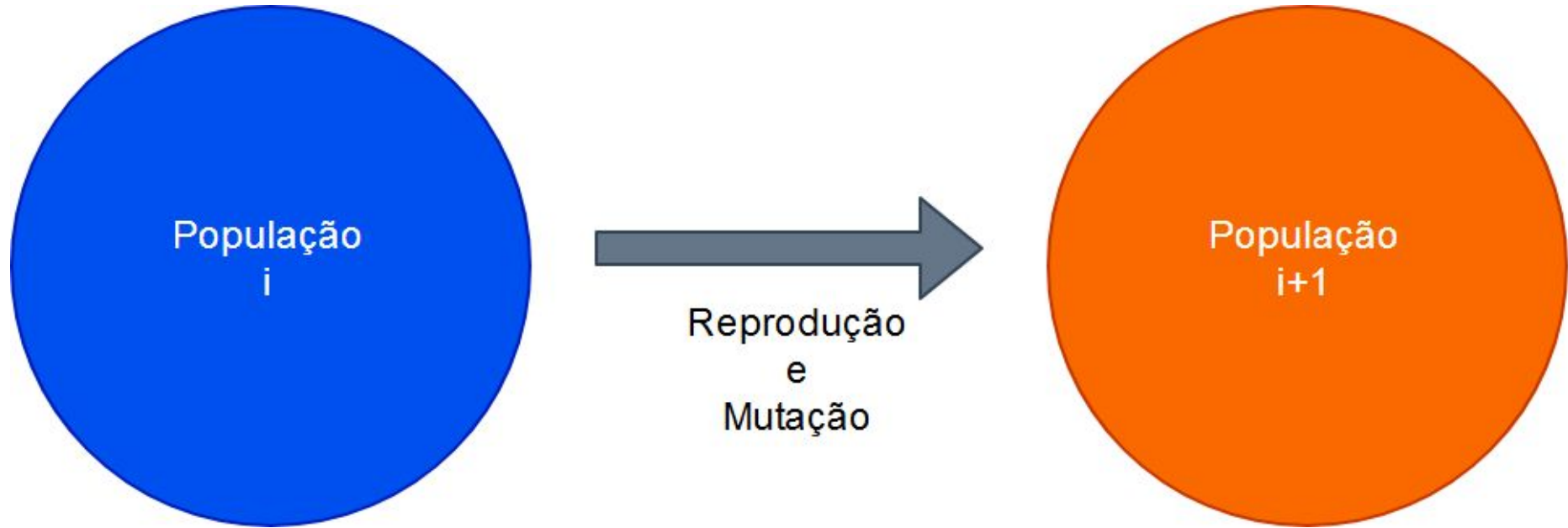
# Algoritmo Genético - Reprodução

## Mutação (Tipos)

- Swap de sequência



# Algoritmo Genético - Reprodução



# Algoritmo Genético - Reprodução

Composições possíveis para população  $i+1$

# Algoritmo Genético - Reprodução

Composições possíveis para população  $i+1$

- Troca de toda a população
  - Isso significa que a população  $i+1$  substitui a população  $i$ .

# Algoritmo Genético - Reprodução

Composições possíveis para população  $i+1$

- Troca de toda a população
  - Isso significa que a população  $i+1$  substitui a população  $i$ .
- Elitismo
  - A população 2 substitui a população 1 ( $k=n-1$ ) sendo  $k$  os indivíduos de 2 e  $n$  os de 1;
  - Acrescenta-se o mais apto da população 1 na população 2.

# Algoritmo Genético - Reprodução

Composições possíveis para população  $i+1$

- Troca de toda a população
  - Isso significa que a população  $i+1$  substitui a população  $i$ .
- Elitismo
  - A população 2 substitui a população 1 ( $k=n-1$ ) sendo  $k$  os indivíduos de 2 e  $n$  os de 1;
  - Acrescenta-se o mais apto da população 1 na população 2.
- Steady State
  - Gera-se  $k < n$  indivíduos e esses  $k$  substituem os  $n$  piores do conjunto da população 1.



# Algoritmo Genético - Critérios de parada

Até onde devo iterar?

# Algoritmo Genético - Critérios de parada

Até onde devo iterar?

- Definir um **número máximo** de gerações;

# Algoritmo Genético - Critérios de parada

Até onde devo iterar?

- Definir um **número máximo** de gerações;
- Parar quando o sistema encontrar a solução para o problema (se conheço a solução desejada);

# Algoritmo Genético - Critérios de parada

Até onde devo iterar?

- Definir um **número máximo** de gerações;
- Parar quando o sistema encontrar a solução para o problema (se conheço a solução desejada);
- Parar quando ocorrer a perda de diversidade, ou seja, o score do melhor indivíduo é o mesmo por n gerações.

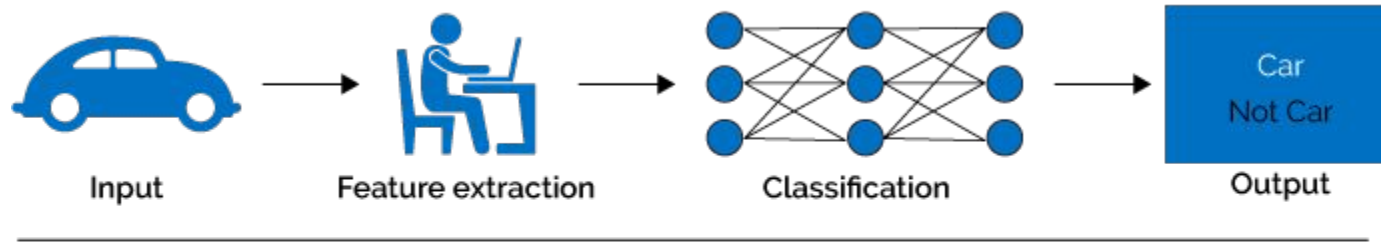
# Algoritmo Genético - Atividade

Clique [aqui](#) para abrir a atividade

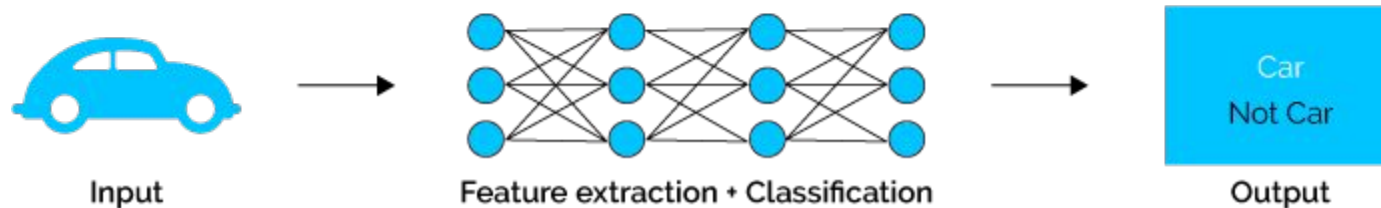
# Redes Neurais e Deep Learning

# Machine Learning e Deep Learning

## Machine Learning



## Deep Learning



# Redes Neurais e Deep Learning

Quando falamos em Deep Learning:



# Redes Neurais e Deep Learning

Quando falamos em Deep Learning:

- Estamos falando de redes neurais;

# Redes Neurais e Deep Learning

Quando falamos em Deep Learning:

- Estamos falando de redes neurais;
- Possuindo múltiplas camadas escondidas;

# Redes Neurais e Deep Learning

Quando falamos em Deep Learning:

- Estamos falando de redes neurais;
- Possuindo múltiplas camadas escondidas;
- Que formam uma hierarquia representativa para cada *feature* de entrada da rede.

# Redes Neurais - Tensores

O que é um tensor?

# Redes Neurais - Tensores

O que é um tensor?

- Tensores são entidades geométricas introduzidas na matemática e na física para generalizar a noção de escalares, vetores e matrizes.

<https://pt.wikipedia.org/wiki/Tensor>

# Redes Neurais - Tensores

O que é um tensor?

# tensor

't'
'e'
'n'
's'
'o'
'r'

tensor of dimensions [6]  
(vector of dimension 6)

3	1	4	1
5	9	2	6
5	3	5	8
9	7	9	3
2	3	8	4
6	2	6	4

tensor of dimensions [6,4]  
(matrix 6 by 4)

2	1	8	2	8	8
2	8	4	5	0	5
2	3	5	3	6	0
7	4	7	1	3	5
2	8	4	5	0	5
2	3	5	3	6	0
7	4	7	1	3	5

tensor of dimensions [4,4,2]

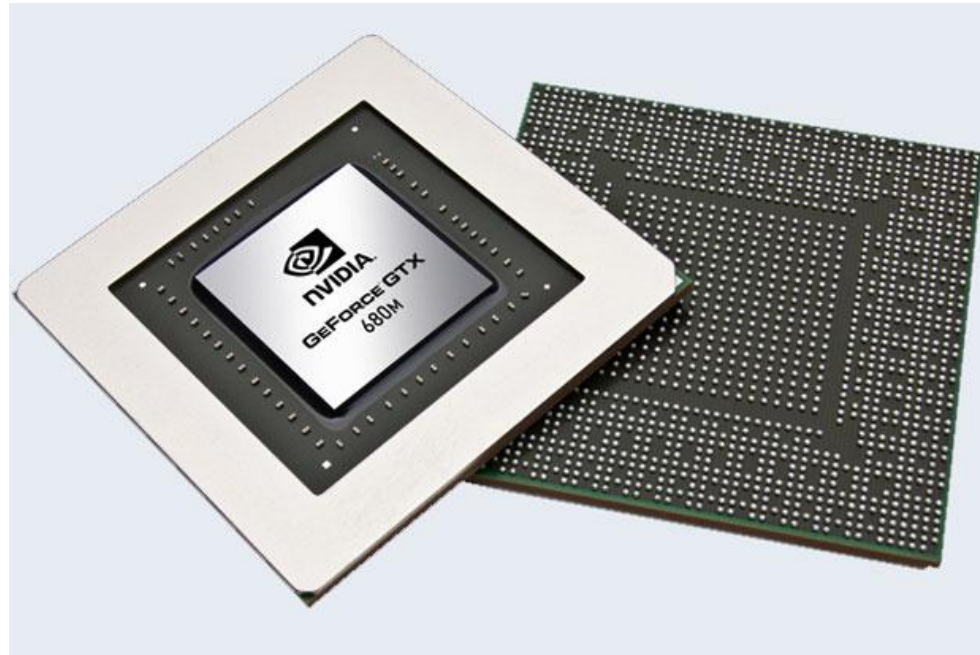
# Redes Neurais - Computação

CPU (Central Process Unit ou Unidade Central de Processamento)



# Redes Neurais - Computação

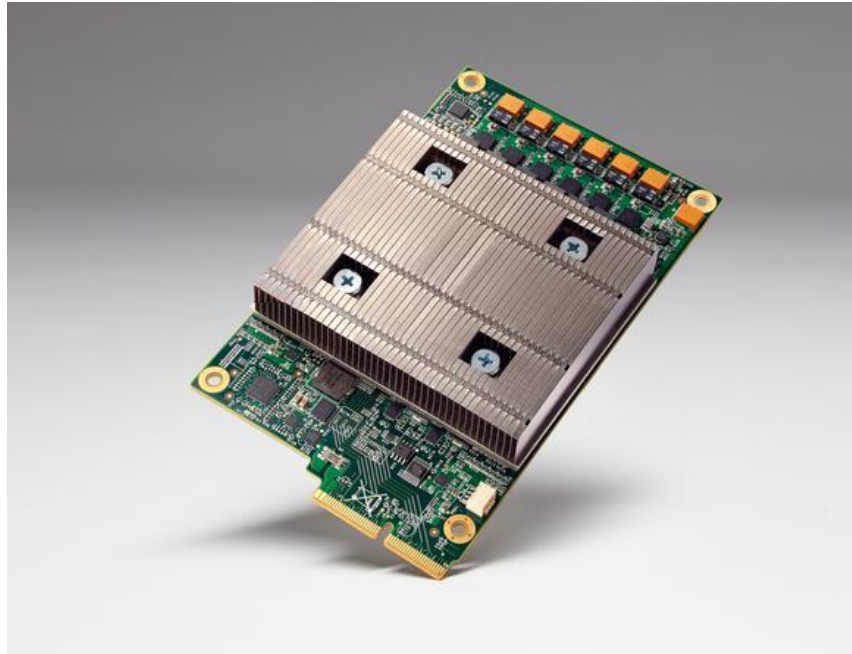
GPU (Graphics Processing Unit ou Unidade de Processamento Gráfico)





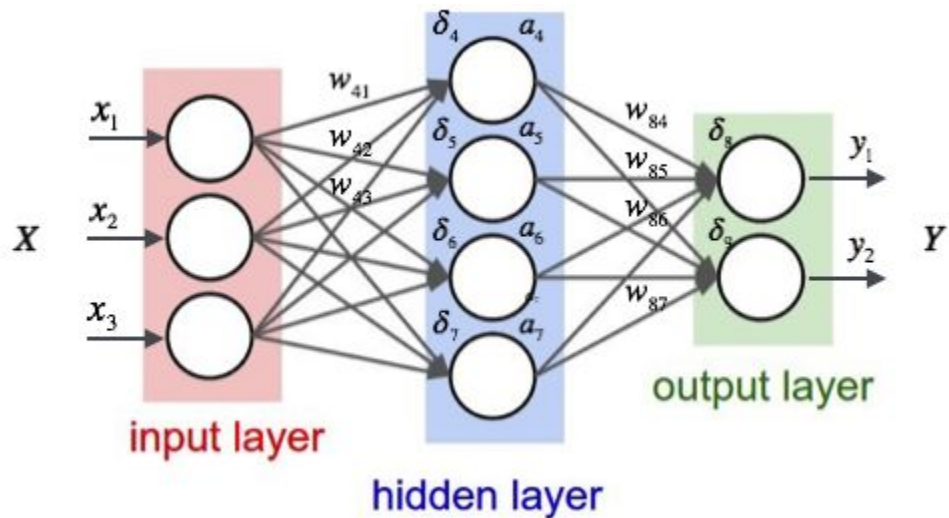
# Redes Neurais - Computação

TPU (Tensor Processing Unit ou Unidade Processamento de Tensor)



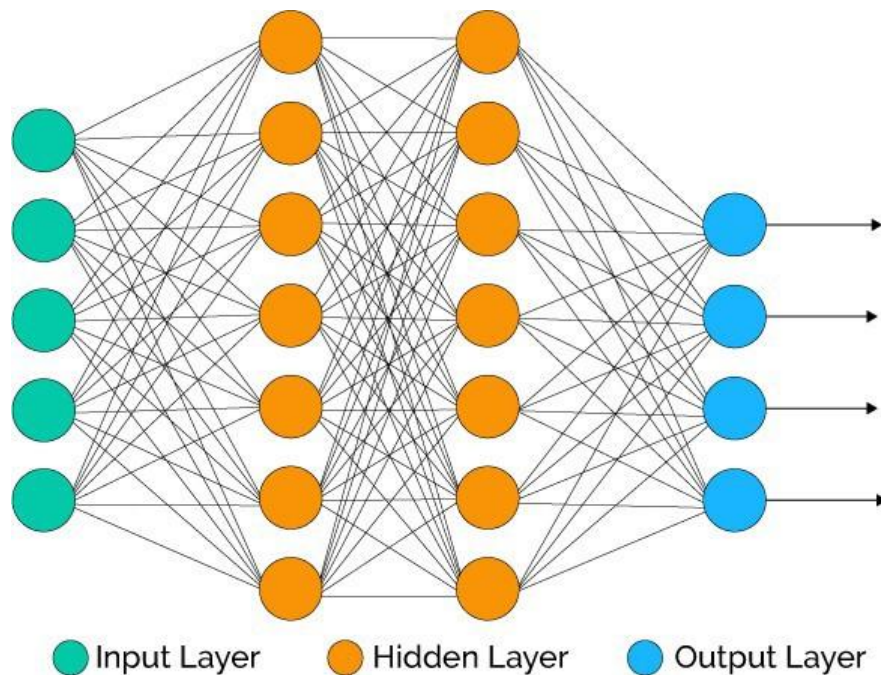
# Redes Neurais - Arquitetura

MLP - MultiLayer Perceptron



# Redes Neurais - Arquitetura

MLP - MultiLayer Perceptron (Deep Learning)



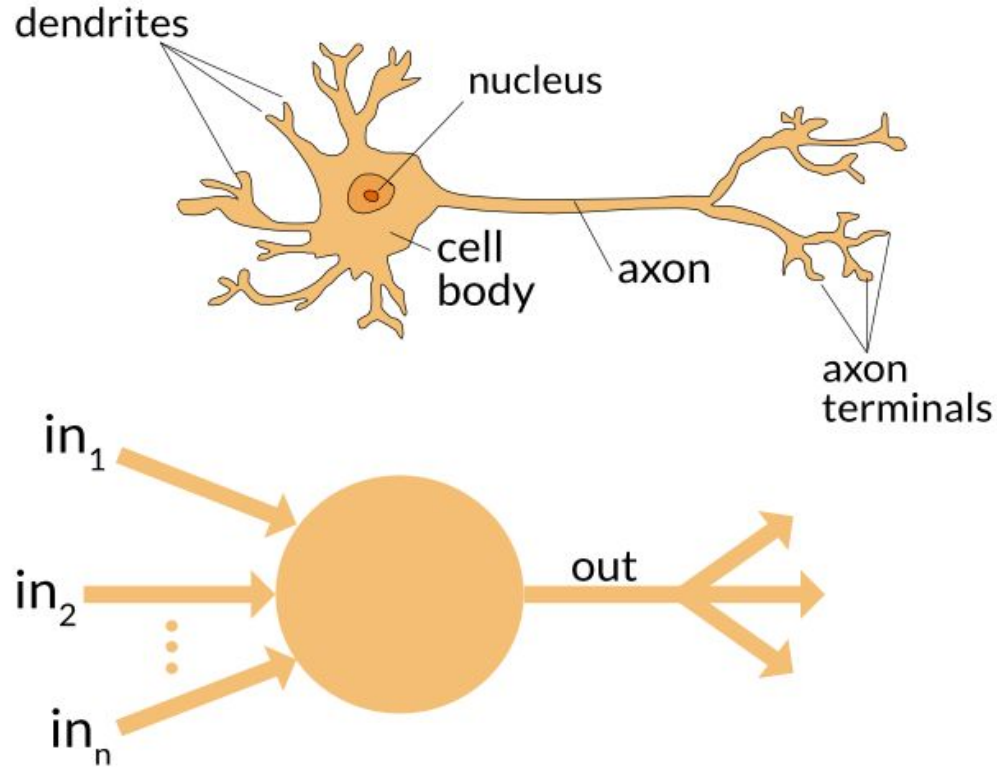
# Redes Neurais - Arquitetura

- Cada nó desta rede executa a soma de cada peso recebido por seus nós anteriores e tem sua multiplicação propagada por uma função de ativação;

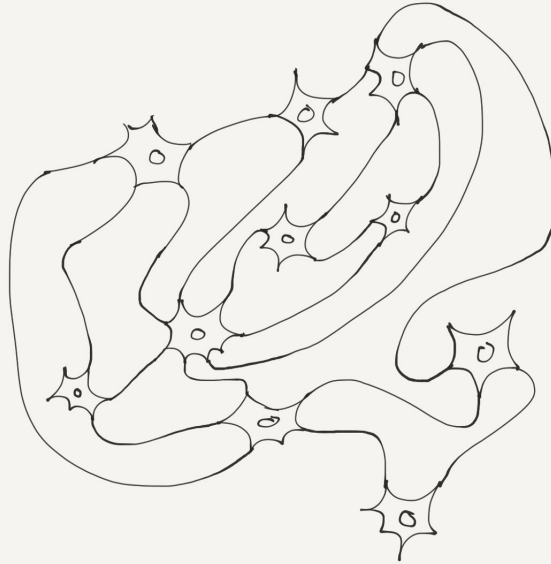
# Redes Neurais - Arquitetura

- Cada nó desta rede executa a soma de cada peso recebido por seus nós anteriores e tem sua multiplicação propagada por uma função de ativação;
- A estrutura tenta representar o que ocorre em um neurônio quando estimulado.

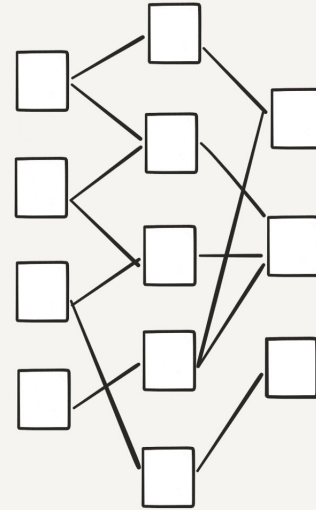
# Redes Neurais - Arquitetura



# Redes Neurais - Arquitetura



BIOLOGICAL



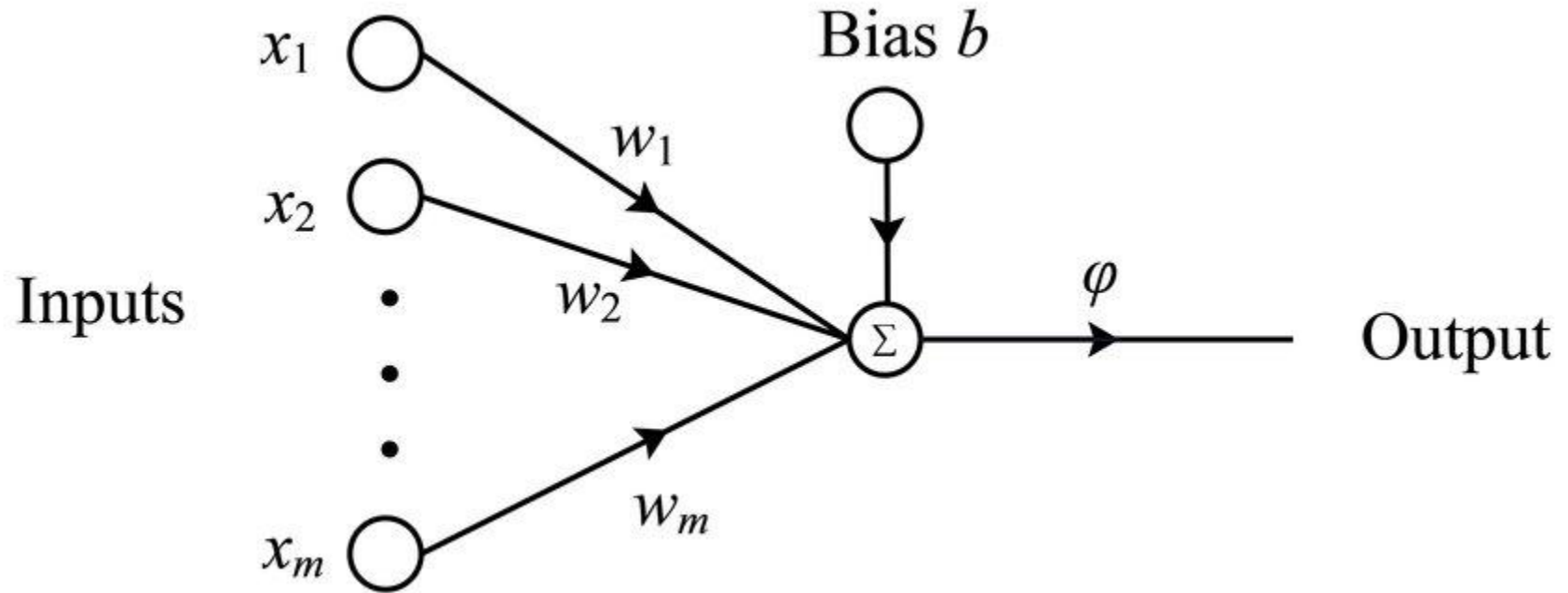
ARTIFICIAL

# Redes Neurais - Arquitetura










- Diferente dos modelos apresentados anteriormente, uma rede neural trabalha melhor com um número maior de features e pode precisar de um número muito maior de entradas de treino para convergir de maneira satisfatória.



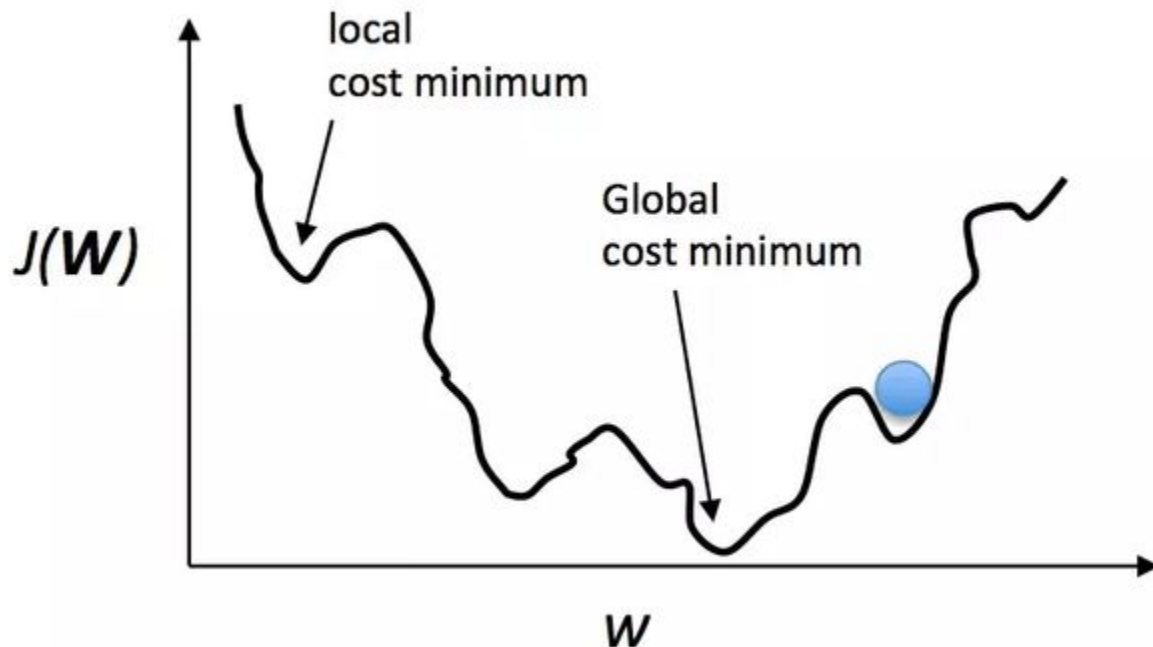
# Redes Neurais - Perceptron



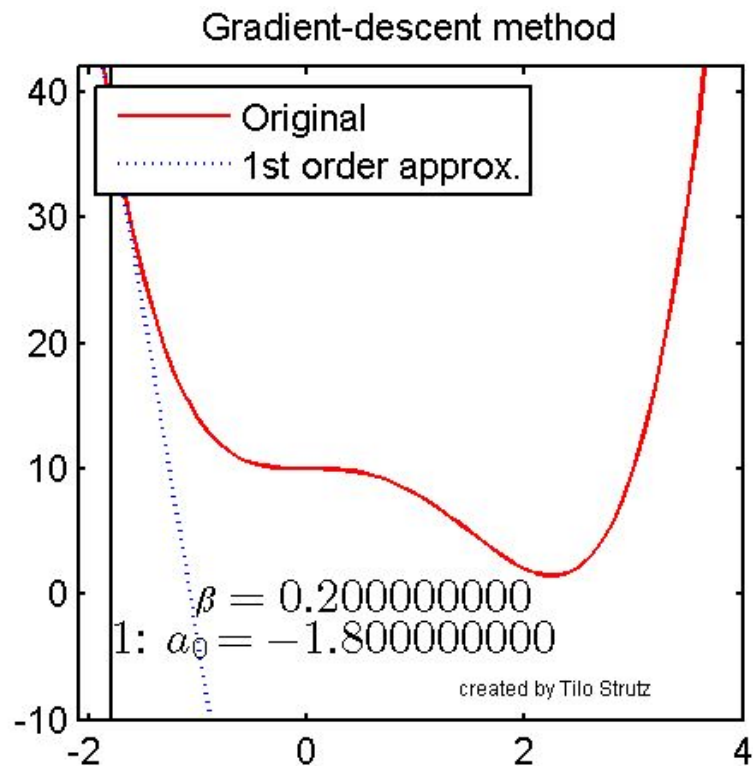
# Redes Neurais - Funções de ativação

Name	Plot	Equation	Derivative (with respect to x)	Range	Order of continuity	Monotonic	Monotonic derivative	Approximates identity near the origin
Identity		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$	$C^\infty$	Yes	Yes	Yes
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ \text{undefined} & \text{for } x = 0 \end{cases}$	$\{0, 1\}$	$C^{-1}$	Yes	No	No
Logistic (a.k.a. Sigmoid or Soft step)		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$ <sup>[1]</sup>	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$	$C^\infty$	Yes	No	No
Tanh		$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$	$C^\infty$	Yes	No	Yes
Rectified linear unit (ReLU) <sup>[11]</sup>		$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases} = \max\{0, x\} = x \mathbf{1}_{x>0}$	$f'(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$	$[0, \infty)$	$C^0$	Yes	Yes	No
Gaussian Error Linear Unit (GELU) <sup>[6]</sup>		$f(x) = x\Phi(x) = \frac{1}{2}x \left( 1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right) \right)$	$f'(x) = \Phi(x) + x\phi(x)$	$(\approx -0.17, \infty)$	$C^\infty$	No	No	No
SoftPlus <sup>[12]</sup>		$f(x) = \ln(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	$(0, \infty)$	$C^\infty$	Yes	Yes	No
Exponential linear unit (ELU) <sup>[13]</sup>		$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} f(\alpha, x) + \alpha & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$	$(-\alpha, \infty)$	$\begin{cases} C^1 & \text{when } \alpha = 1 \\ C^0 & \text{otherwise} \end{cases}$	Yes iff $\alpha \geq 0$	Yes iff $0 \leq \alpha \leq 1$	Yes iff $\alpha = 1$
Scaled exponential linear unit (SELU) <sup>[14]</sup>		$f(\alpha, x) = \lambda \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ with $\lambda = 1.0507$ and $\alpha = 1.67326$	$f'(\alpha, x) = \lambda \begin{cases} \alpha(e^x) & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\lambda\alpha, \infty)$	$C^0$	Yes	No	No
Leaky rectified linear unit (Leaky ReLU) <sup>[15]</sup>		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\infty, \infty)$	$C^0$	Yes	Yes	No

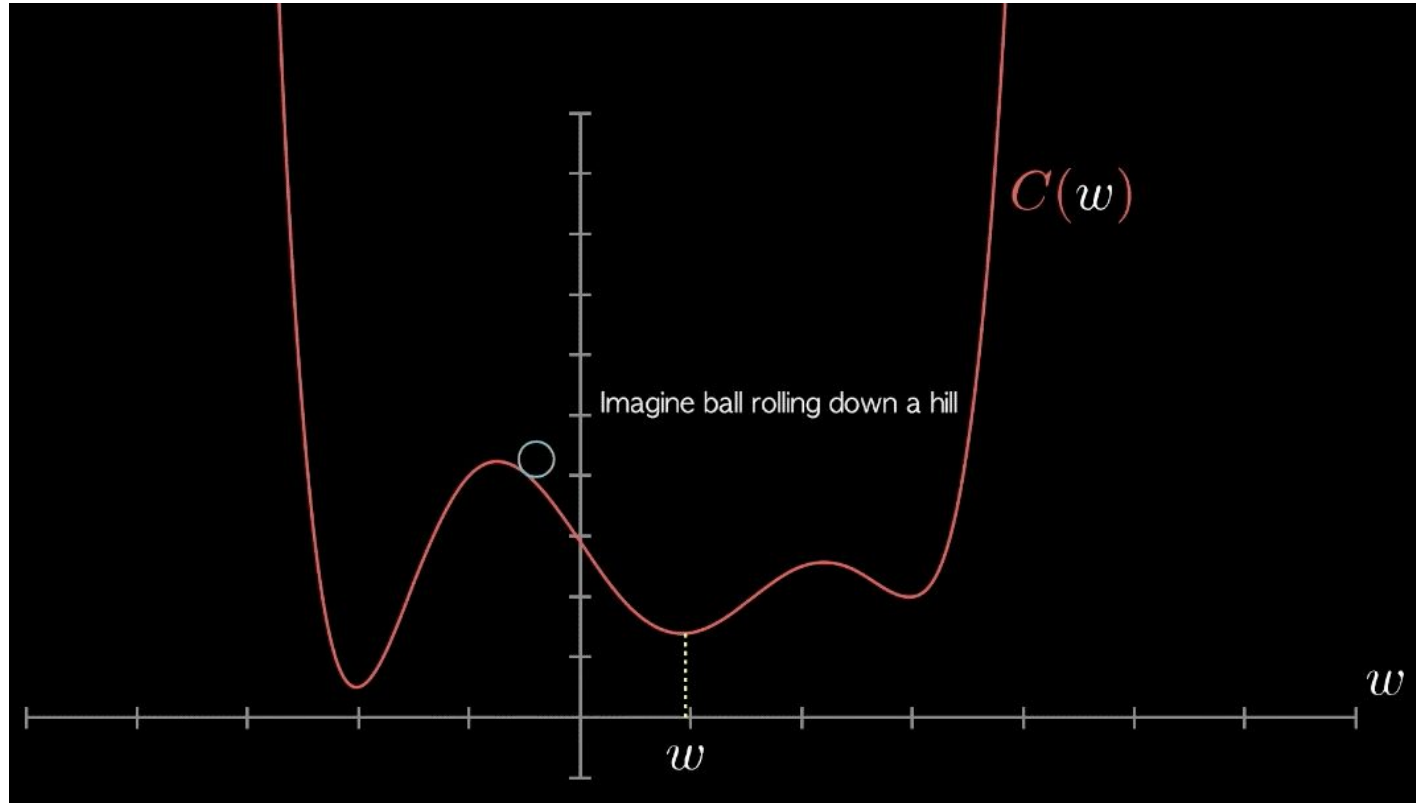
# Redes Neurais - Função de otimização + Gradiente Descendente



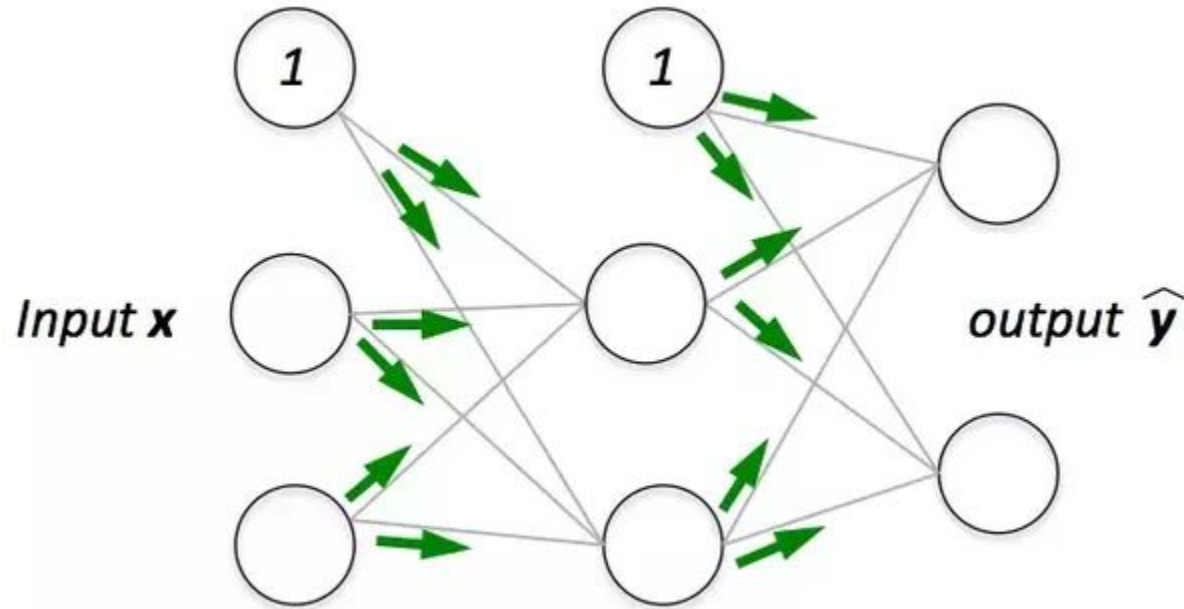
# Redes Neurais - Função de otimização + Gradiente Descendente



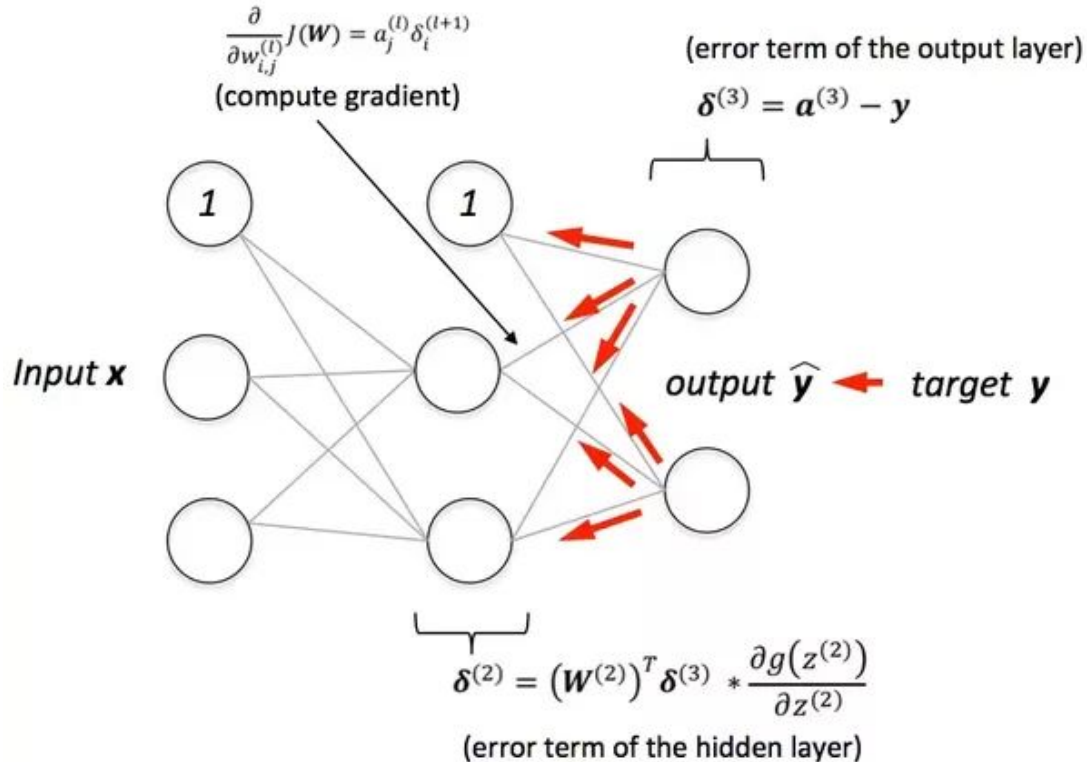
# Redes Neurais - Função de otimização + Gradiente Descendente



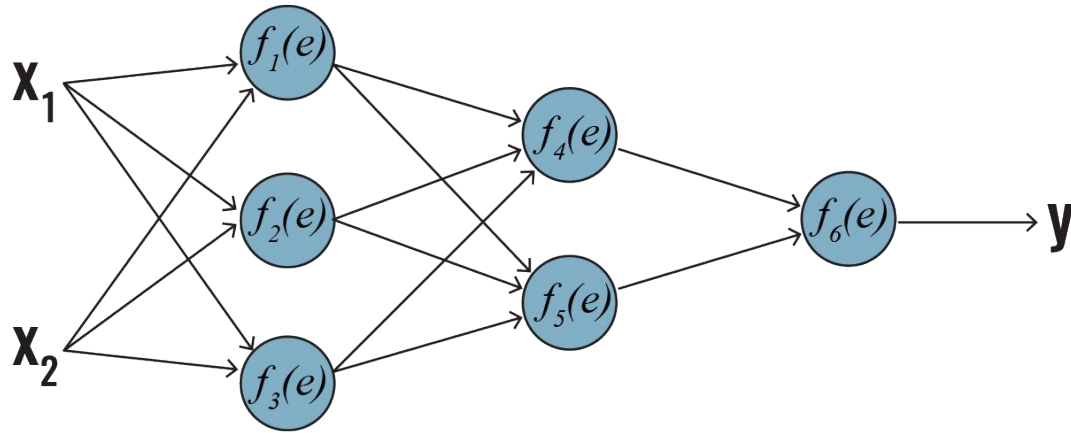
# Redes Neurais - Back Propagation



# Redes Neurais - Back Propagation

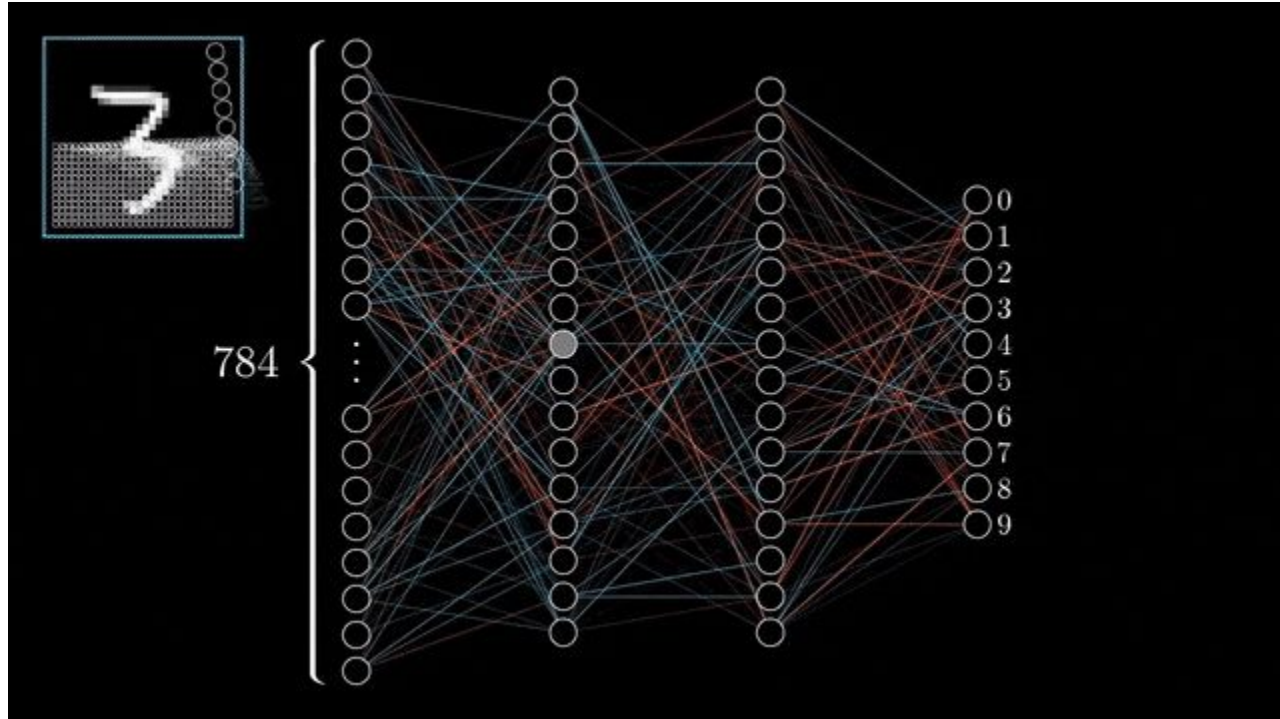


# Redes Neurais - Back Propagation

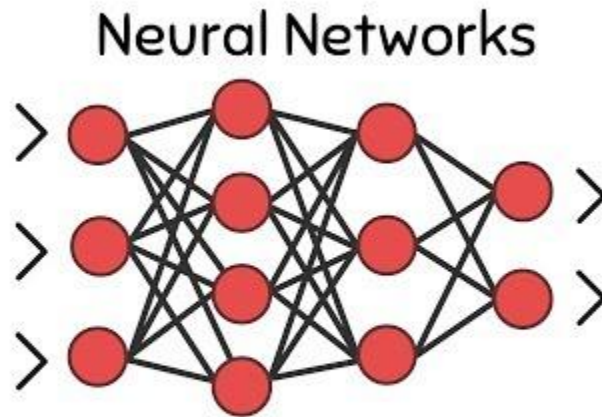




# Redes Neurais - Back Propagation



# Redes Neurais - Back Propagation



**EXPLAINED IN A MINUTE**

# Redes Neurais - Overfitting e Underfitting

Overfitting e Underfitting são o caso de resultados ruins em modelos de inteligência artificial.

# Redes Neurais - Overfitting e Underfitting

Overfitting e Underfitting são o caso de resultados ruins em modelos de inteligência artificial.

Podemos classificar como:

# Redes Neurais - Overfitting e Underfitting

## Underfitting:

- O modelo tem resultados ruins com a parcela de treino e validação, ou seja, o modelo não é capaz de entregar o resultado esperado por causa de sua arquitetura ou da qualidade da informação que lhe é apresentada.

# Redes Neurais - Overfitting e Underfitting

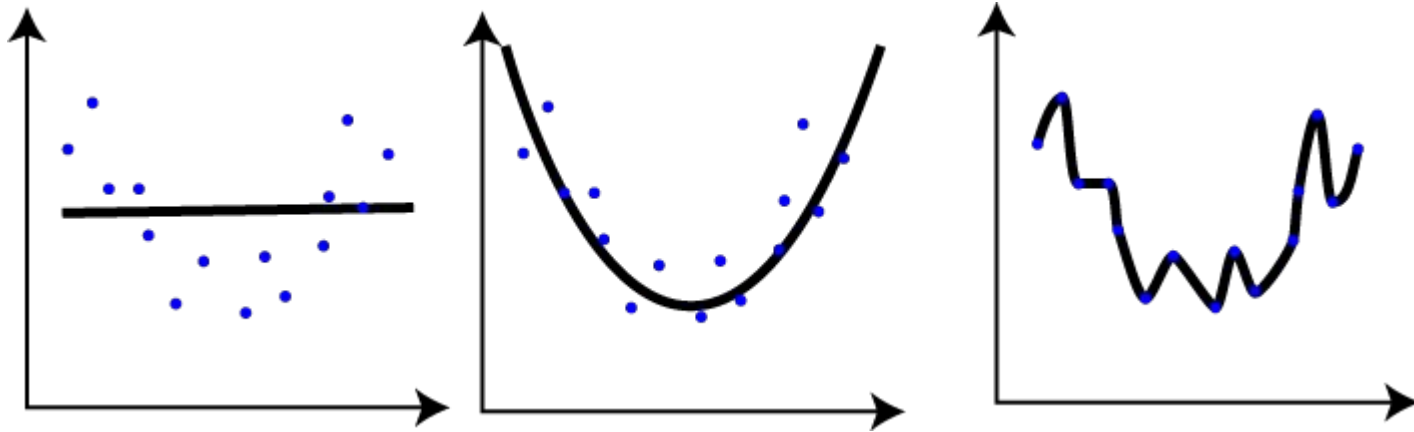
## Underfitting:

- O modelo tem resultados ruins com a parcela de treino e validação, ou seja, o modelo não é capaz de entregar o resultado esperado por causa de sua arquitetura ou da qualidade da informação que lhe é apresentada.

## Overfitting:

- O modelo tem resultados excelentes com sua parcela de treinamento, mas é incapaz de reproduzir os mesmos resultados com a parcela de validação.

# Redes Neurais - Overfitting e Underfitting



# Redes Neurais - Overfitting e Underfitting

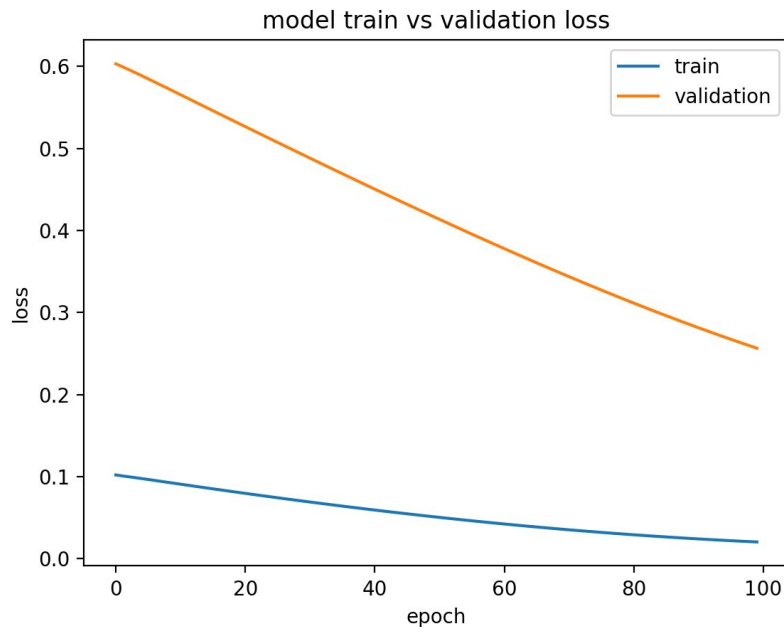
Underfitting:

- Quando verificamos que o erro da parcela de treino é menor que a de validação e existe uma tendência de queda na parcela de validação, ou seja, esse valor de erro pode melhorar com um número maior de épocas.



# Redes Neurais - Overfitting e Underfitting

Underfitting:



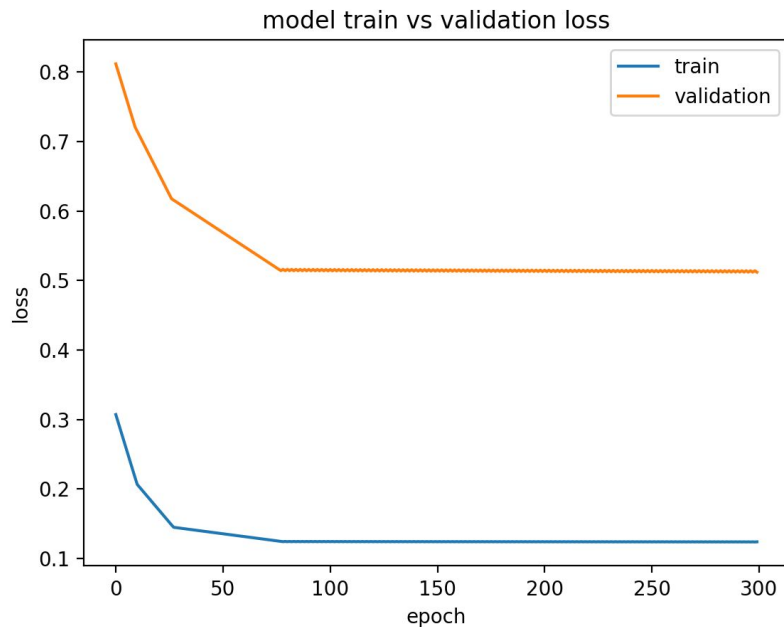
# Redes Neurais - Overfitting e Underfitting

## Underfitting:

- Outro exemplo é quando temos um comportamento parecido entre a curva de treino e validação, porém, temos uma variação no erro entre estas curvas. Esse comportamento pode ser causado por um **underfit** da rede e, neste caso, pode ser corrigido modificando a estrutura da rede.

# Redes Neurais - Overfitting e Underfitting

Underfitting:



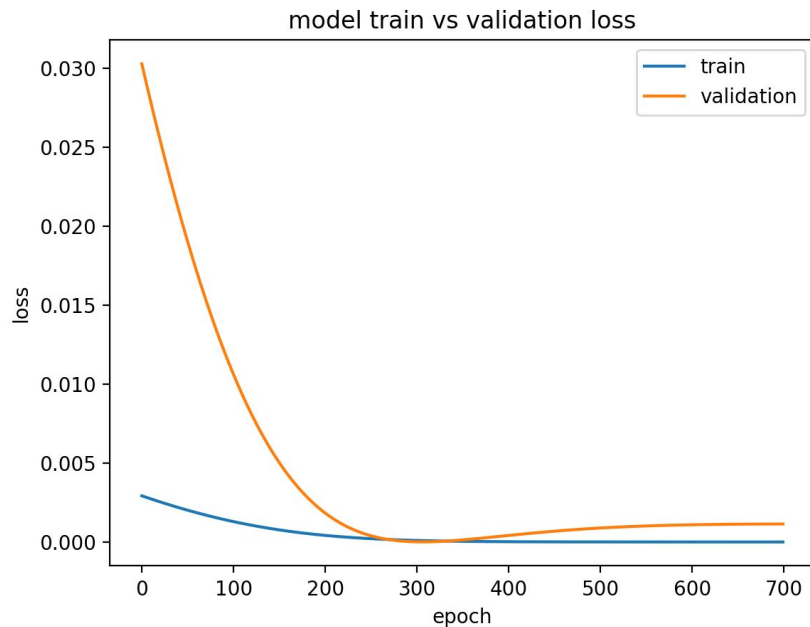
# Redes Neurais - Overfitting e Underfitting

## Overfitting:

- Quando verificamos que o erro da parcela de treino melhora quanto maior é o número de épocas, porém, a parcela de validação tem um comportamento completamente oposto ou vemos este erro diminuir até um determinado ponto e depois degradar.

# Redes Neurais - Overfitting e Underfitting

Overfitting:



# Redes Neurais - Overfitting e Underfitting

Podemos combater o **overfit da rede** adicionando uma regularização dos peso do modelo ou uma política de dropout.

# Redes Neurais - Regularização de peso

## L1 regularization

- O custo adicionado é proporcional a soma dos valores absolutos dos pesos.

# Redes Neurais - Regularização de peso

## L1 regularization

- O custo adicionado é proporcional a soma dos valores absolutos dos pesos.

## L2 regularization

- O custo adicionado é proporcional a soma dos valores quadráticos dos pesos.



# Redes Neurais - Regularização de peso

## L1 regularization

- O custo adicionado é proporcional a soma dos valores absolutos dos pesos.

## L2 regularization

- O custo adicionado é proporcional a soma dos valores quadráticos dos pesos.

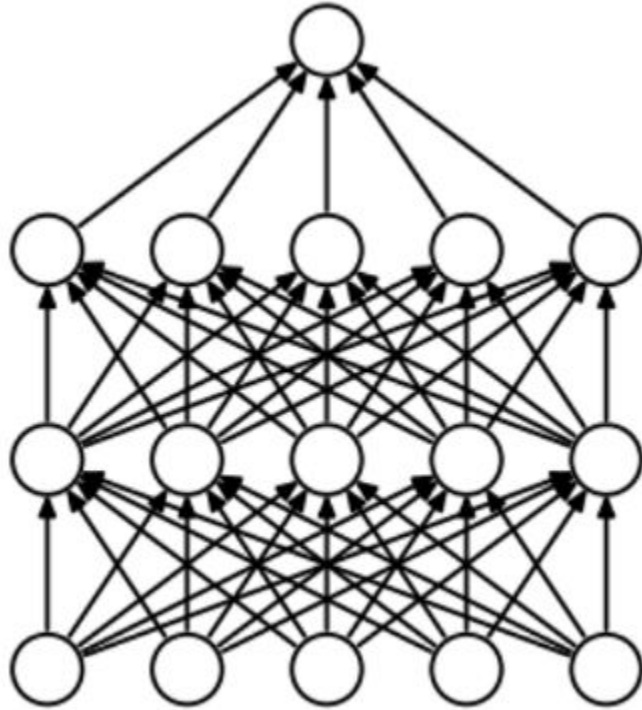
Em poucas palavras, L1 tem o mesmo efeito de reduzir o número de inputs da rede, fazendo com que inputs que tem peso pequeno se aproximem de zero e reduzindo o ruído causado. L2 resulta valores de peso geral menores e estabiliza os pesos quando há alta correlação entre os recursos de entrada.

# Redes Neurais - Regularização de peso

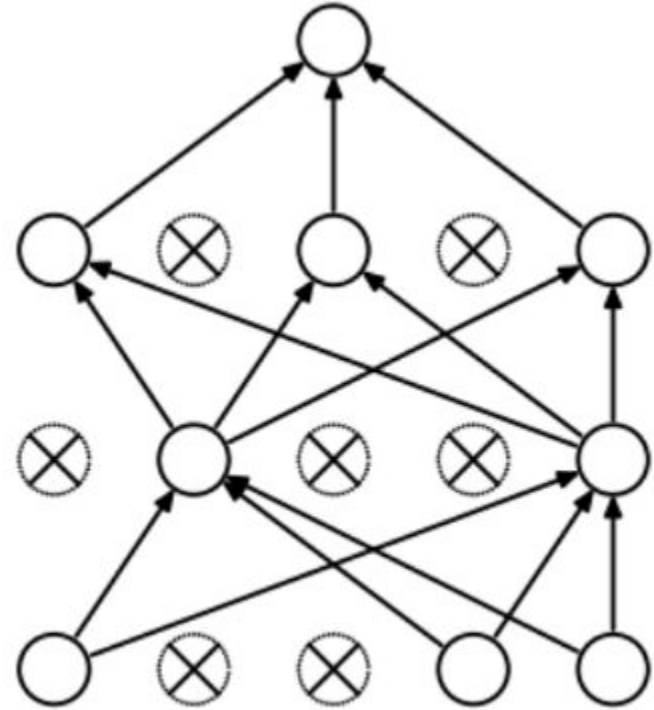
## Dropout

- Dropout se resume em, aleatoriamente, desligar conexões entre os nós de duas camadas. Diminuindo assim a complexidade da arquitetura da rede.

# Redes Neurais - Regularização de peso

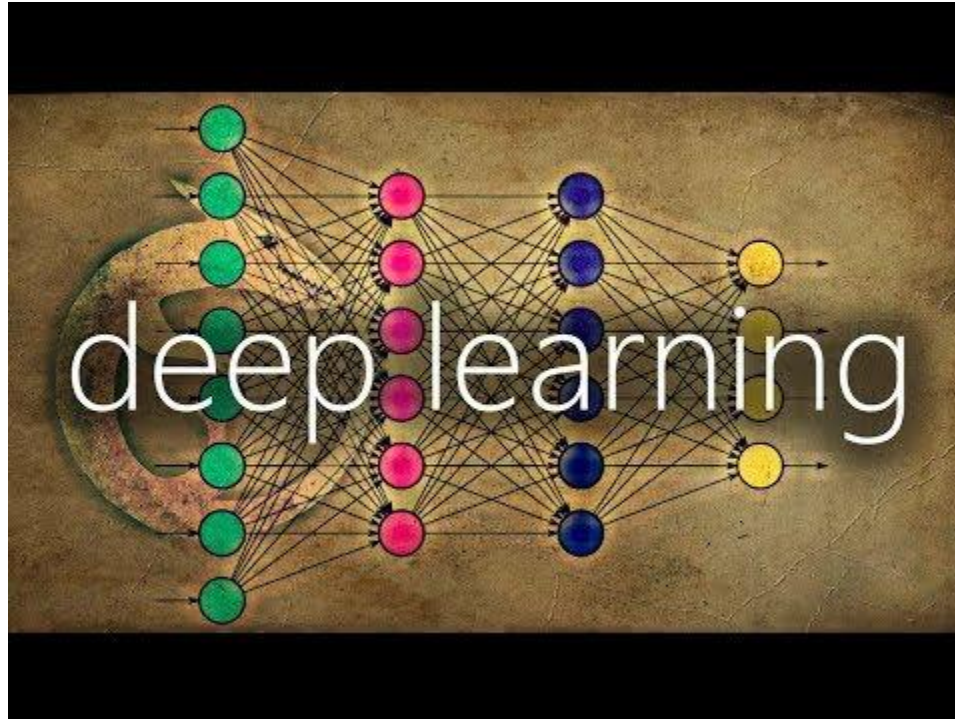


(a) Standard Neural Net

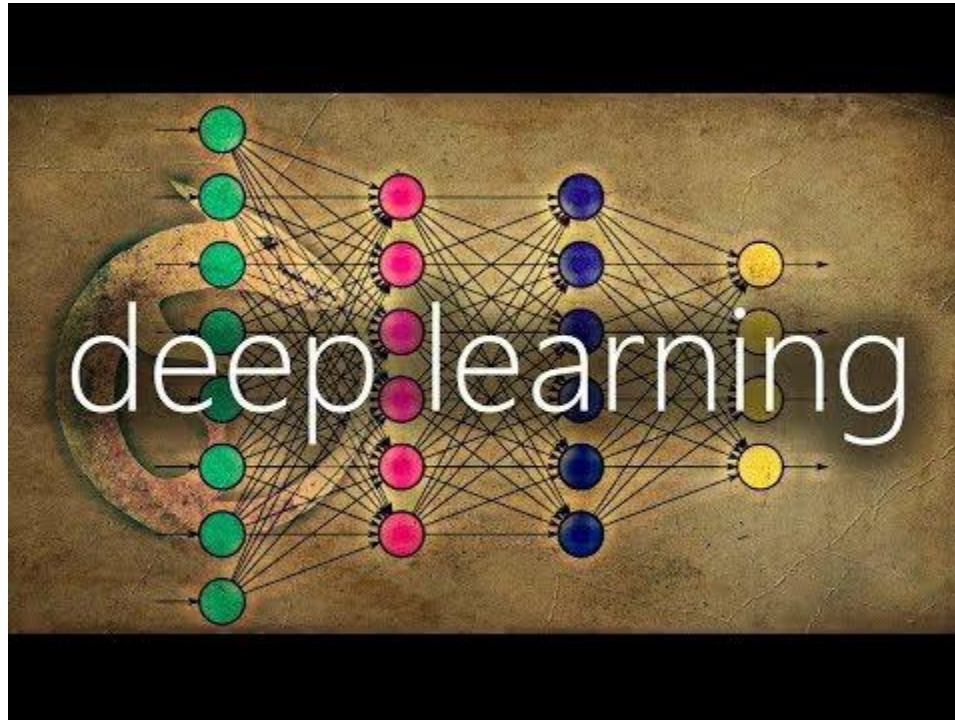


(b) After applying dropout.

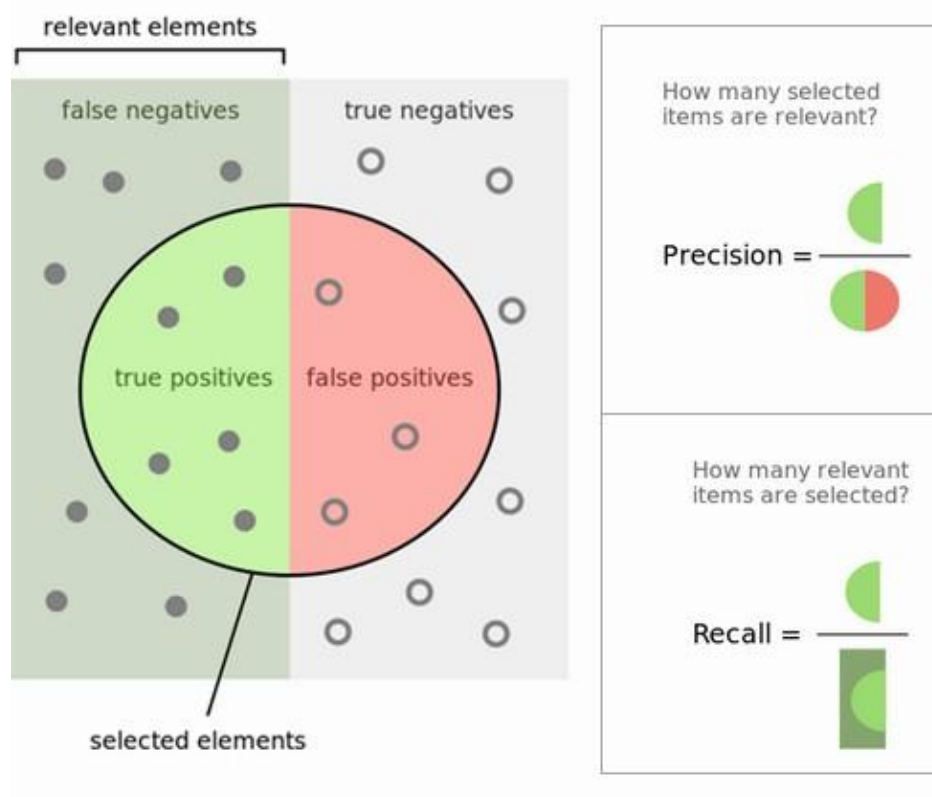
# Redes Neurais - Regularização de peso



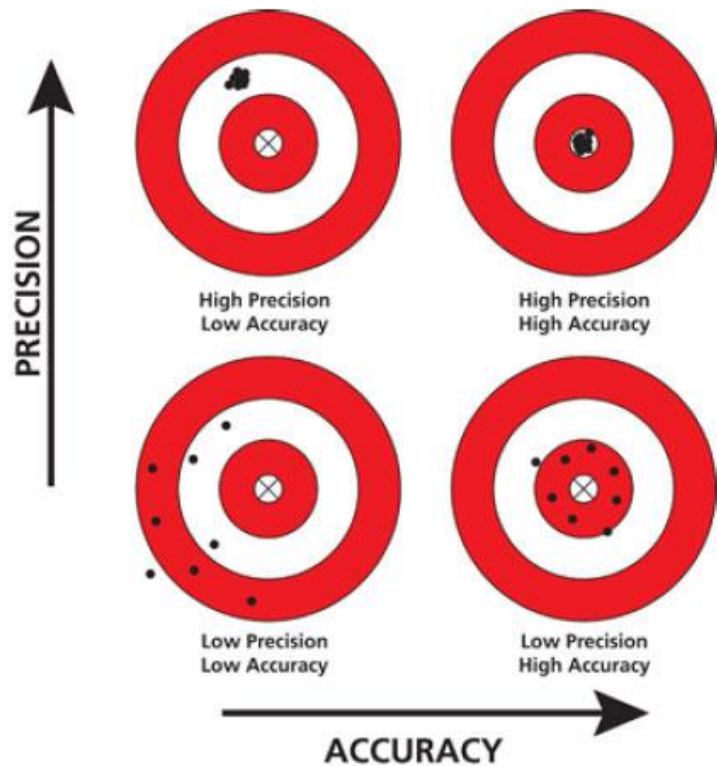
# Redes Neurais - Regularização de peso



# Redes Neurais - Métricas de resultado



# Redes Neurais - Métricas de resultado



# Redes Neurais - Métricas de resultado

## Curva ROC

- A Curva de operação do receptor (\*Receiver Operating Characteristic\*) é uma representação gráfica que ilustra o desempenho (ou performance) de um sistema classificador binário e como o seu limiar de discriminação é variado.



# Redes Neurais - Métricas de resultado

## Curva ROC

- Sensitivity
  - Também chamada de \*true positive rate\*, \*recall\* ou \*probabilidade de detecção\*. Mede a proporção de positivos realmente detectados como positivos.

$$TPR = \frac{TP}{TP + FN}$$

# Redes Neurais - Métricas de resultado

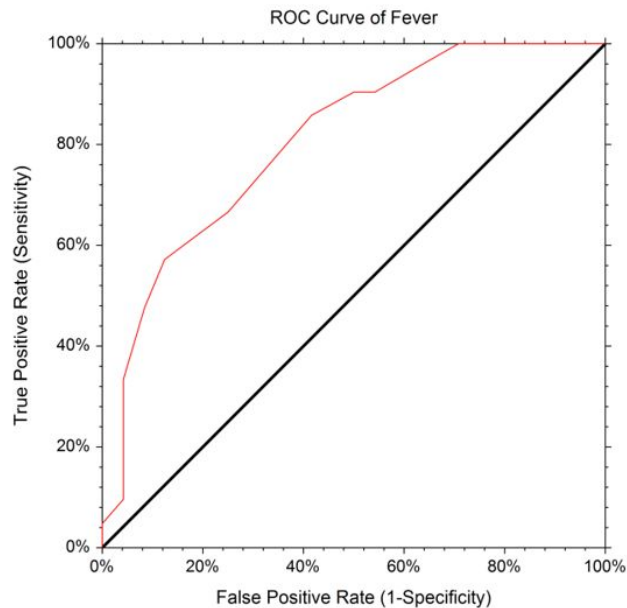
## Curva ROC

- Specificity
  - Também chamada de \*true negative rate\*, mede a proporção de negativos realmente detectados como negativos.

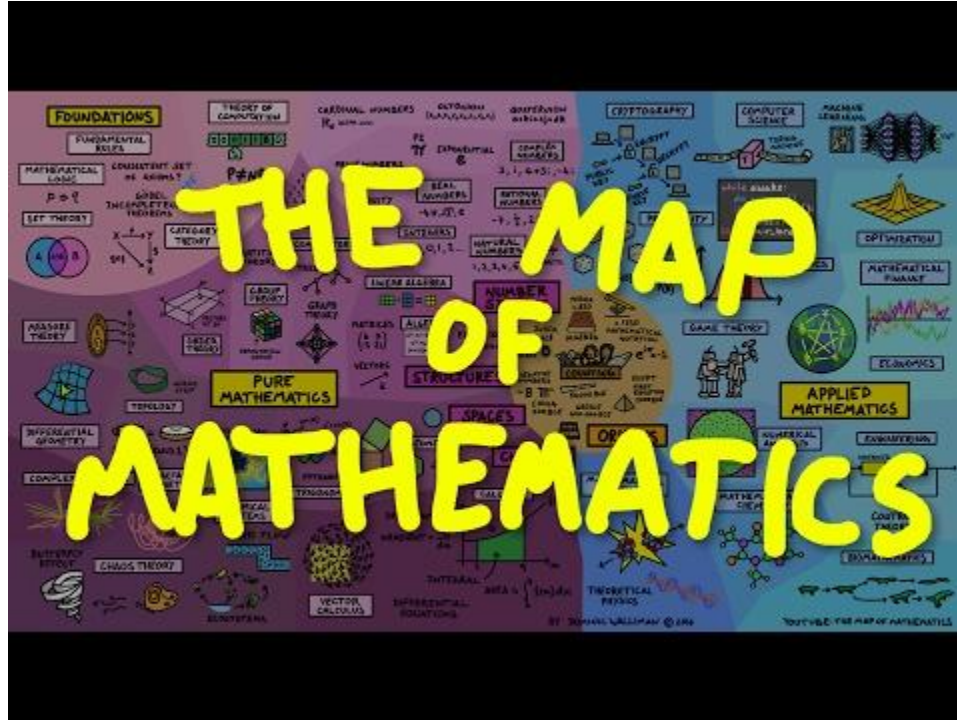
$$TNR = \frac{TN}{TN + FP}$$

# Redes Neurais - Métricas de resultado

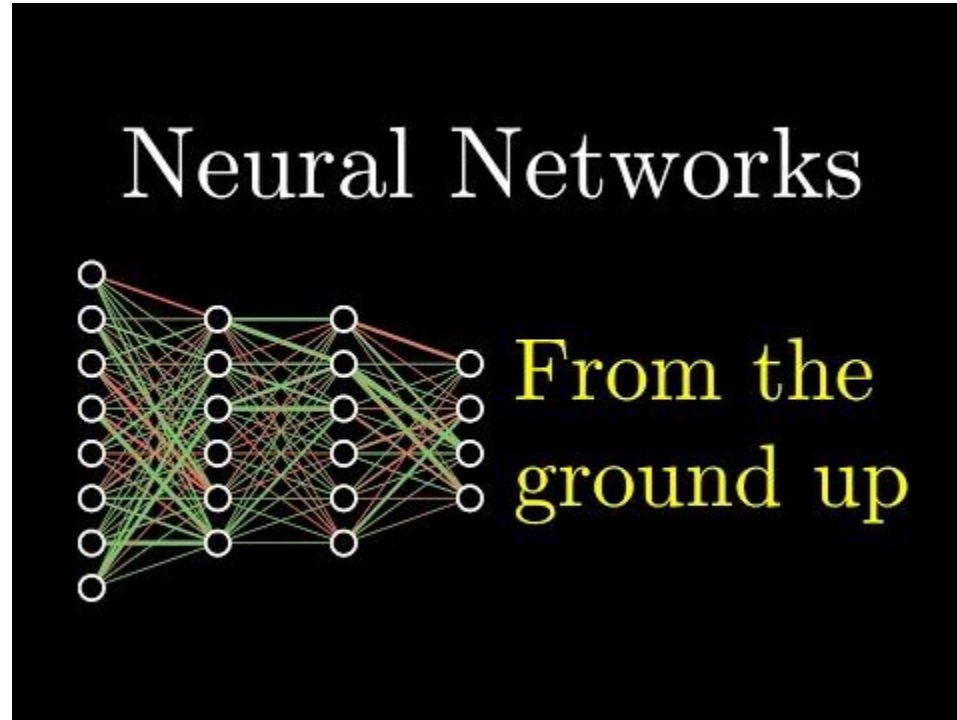
## Curva ROC



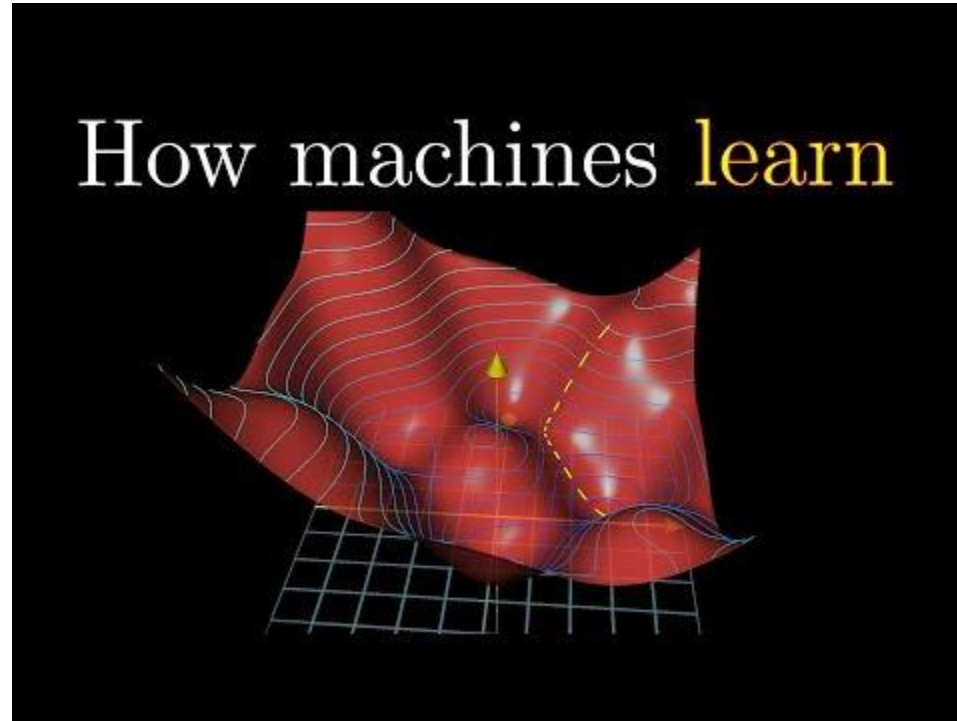
# Redes Neurais - Para revisar



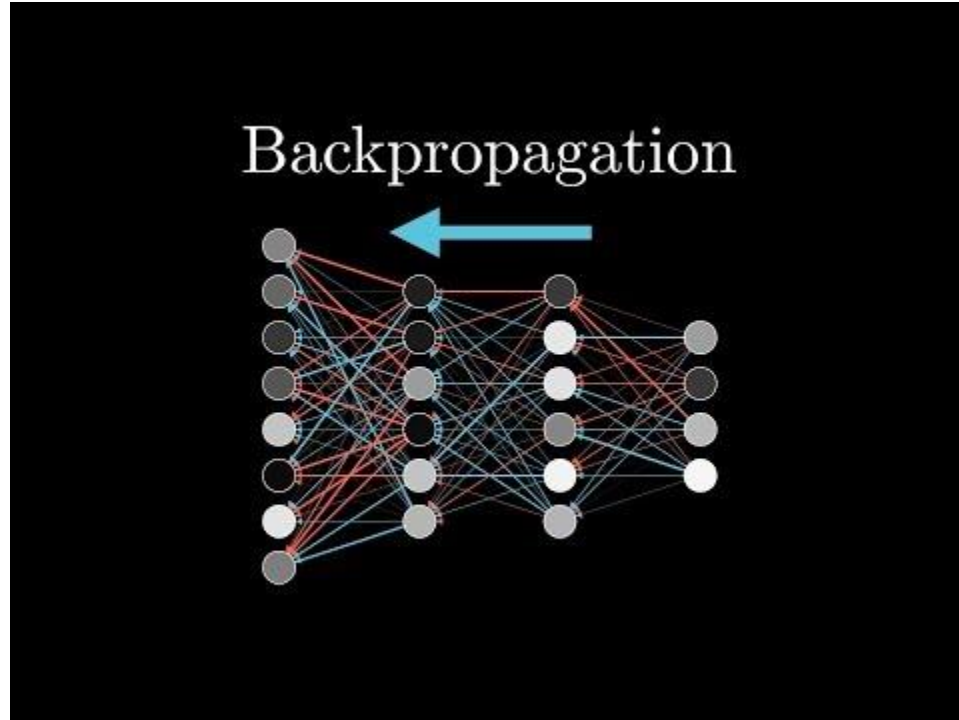
# Redes Neurais - Para revisar



# Redes Neurais - Para revisar

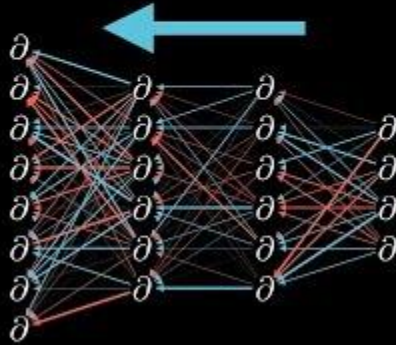


# Redes Neurais - Para revisar



# Redes Neurais - Para revisar

Backpropagation  
calculus





# Redes Neurais - Atividade

Clique [aqui](#) para abrir a atividade