

# T319 - Introdução ao Aprendizado de Máquina: *Introdução*



***Inatel***

Felipe Augusto Pereira de Figueiredo  
felipe.figueiredo@inatel.br

# A disciplina

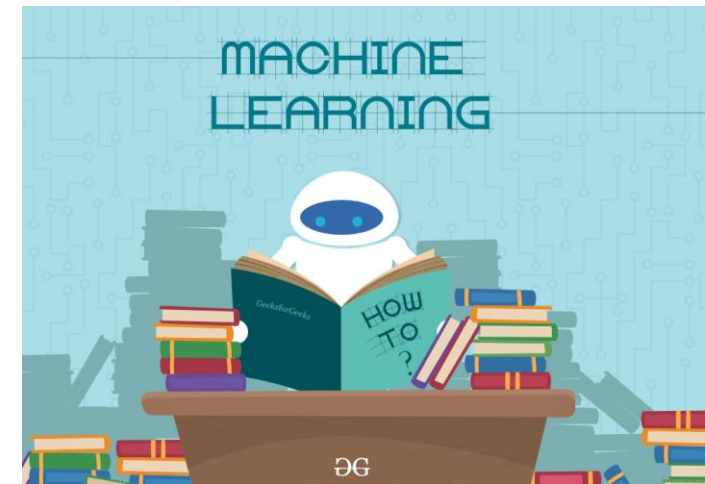
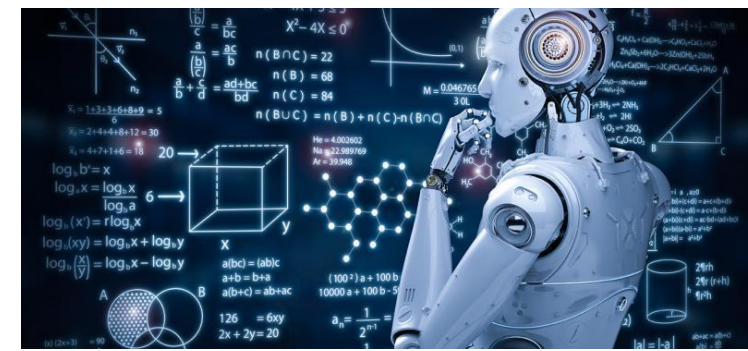
- Introdução ao aprendizado de máquina.
- Como o próprio nome diz, é um curso introdutório onde veremos os conceitos básicos de funcionamento de vários ***algoritmos de aprendizado de máquina*** ou do Inglês, ***machine learning*** (ML).
- O curso será o mais prático possível, com vários exercícios envolvendo o uso dos algoritmos discutidos.
- O curso será dividido em duas partes: T319 e T320.
- Não nós aprofundaremos nos conceitos matemáticos envolvidos.
- Porém, precisamos conhecer Python e alguns conceitos de álgebra linear e estatística.

# Cronograma

Aula	Data	Dia	Horário	Atividade
1	13/8/2021	Sexta-feira	21:30 às 23:10	Introdução ao Aprendizado de Máquina
2	27/8/2021			Introdução ao Aprendizado de Máquina
3	10/9/2021			Introdução ao Aprendizado de Máquina
4	24/9/2021			Introdução ao Aprendizado de Máquina
5	8/10/2021			Introdução ao Aprendizado de Máquina
6	22/10/2021			Introdução ao Aprendizado de Máquina
7	5/11/2021			Introdução ao Aprendizado de Máquina
8	19/11/2021			Introdução ao Aprendizado de Máquina
9	3/12/2021			Projeto Final
10	17/12/2021			NP3

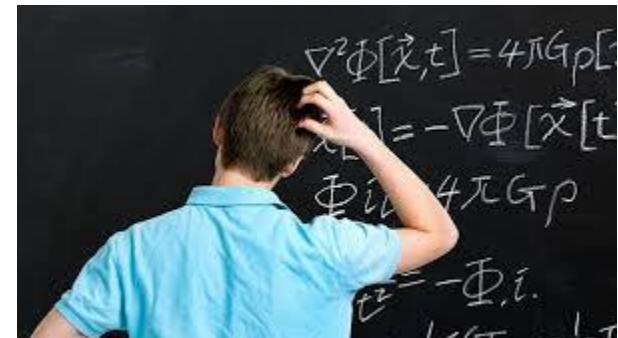
# Objetivo do curso

- O objetivo principal do curso é apresentar à vocês
  - os conceitos fundamentais da teoria do aprendizado de máquina.
  - um conjunto de ferramentas (ou seja, algoritmos) de aprendizado de máquina.
- Ao final do curso vocês devem ser capazes de
  - Entender e discutir sobre os principais algoritmos de ML.
  - Compreender a terminologia utilizada na área.
  - Aplicar algoritmos de ML para a resolução de problemas.
  - Analisar e entender novos algoritmos de ML.
  - Criar seus próprios projetos.



# Avaliação do curso

- Avaliação final
  - Uma (1) atividade final valendo 85% da nota.
  - Envolvendo questões teóricas e/ou práticas.
- Atividades
  - Exercícios e quizzes valendo 15% da nota.
  - Ao longo das aulas e para casa.
  - [Entregues no MS Teams.](#)



# Motivação

- **Emprego:** grandes companhias (e.g., Google, Facebook, Amazon, etc.) usam ML para resolver os mais diversos tipos de problemas e assim aumentarem sua eficiência e consequentemente os lucros.
- **Pesquisa:** já se prevê que ML terá um papel importante no desenvolvimento da próxima geração de redes móveis e sem-fio (e.g., 6G).

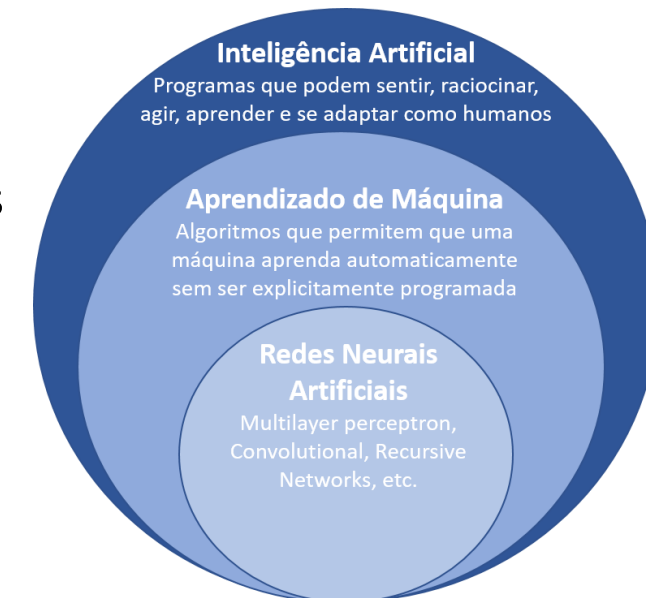
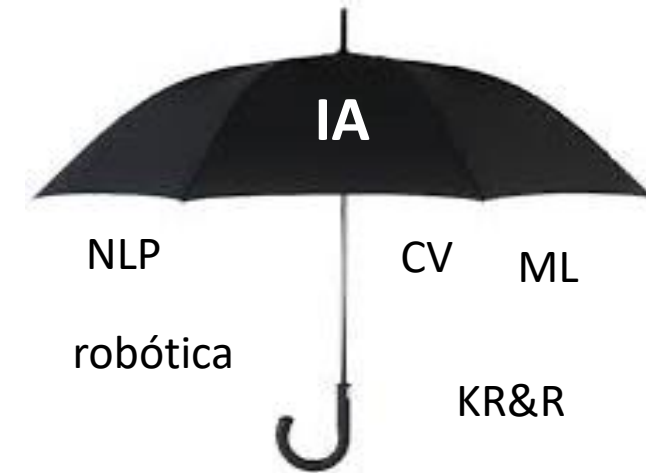


# Definições e objetivo da IA

- **Definição:** *“Capacidade de um sistema de interpretar corretamente dados externos (vindos do ambiente), aprender com esses dados e usá-los para atingir tarefas e objetivos específicos por meio de adaptação flexível.” (Andreas Kaplan).*
- **Objetivo:** Criar máquinas que **imitem** nossas *habilidades mentais*, ou seja, criar máquinas que são **modelos aproximados** de nossas habilidades de aprender, raciocinar, enxergar, falar, ouvir, etc.
- IA utiliza a **experiência** para adquirir **conhecimento** e também como aplicar esse conhecimento a problemas desconhecidos.

# Inteligência Artificial

- IA é uma área muito ampla que **engloba** várias aplicações (ou sub-áreas ou objetivos) tais como
  - processamento de linguagem natural,*
  - representação do conhecimento,*
  - raciocínio automatizado,*
  - planejamento,*
  - visão computacional,*
  - robótica,*
  - aprendizado de máquina, que por sua vez engloba redes neurais artificiais, deep learning, etc. e*
  - inteligência artificial geral.*





# Algumas aplicações de IA

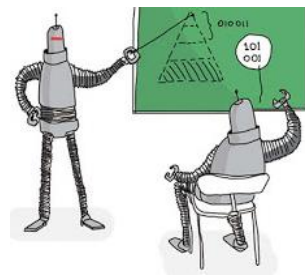


Algumas áreas onde IA é aplicada são:

- **Transporte:** veículos terrestres e aéreos autônomos, previsão do tráfego, etc.
- **Negócios:** recomendação de anúncios, produtos e conteúdos (e.g., netflix), chatbots para atendimento ao cliente, etc.
- **Educação:** pontuação automatizada de fala em testes de Inglês, correção de provas, chatbots para realização de matrículas, dúvidas, etc.
- **Clima:** previsão do tempo (temperatura, chuva, furacões, etc.).
- **Medicina:** detecção e/ou previsão de doenças (câncer, Alzheimer, pneumonia, COVID-19, etc.), chatbots que auxiliam no agendamento de consultas e respondem perguntas referentes a uma doença, descoberta de novas drogas, etc.
- **Finanças:** detecção de fraudes com cartão de crédito, previsão do comportamento do mercado de ações, etc.
- **Tecnologia:** filtros AntiSpam, “motores” de busca como o do Google, reconhecimento de fala, conversão de texto/fala e fala/texto, assistentes pessoais on-line (e.g., *Siri, Alexa*, etc.), tradução de textos.



# Foco do curso



Natural Language  
Processing



Knowledge  
Representation



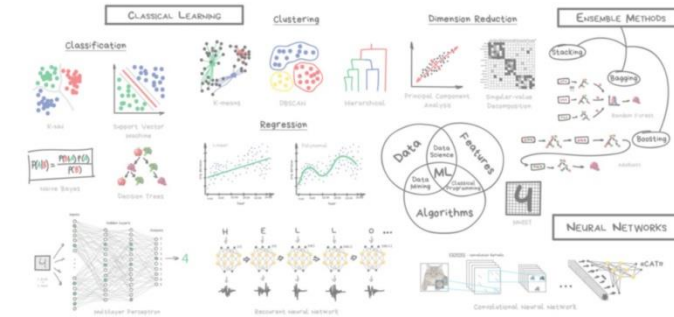
Automated  
Reasoning



Machine  
Learning

- Como vimos, IA é um termo muito amplo, abrangendo várias sub-áreas, usado para designar máquinas capazes executar ***tarefas*** de forma inteligente.
- **Foco do curso:** estudo dos principais algoritmos de ***Aprendizado de Máquina***.
- **Por quê?**
  - ***Caixa de ferramentas:*** ML oferece ferramentas importantes para a solução e análise eficiente de vários problemas em várias áreas.
  - ***Redução de complexidade e custo:*** vários algoritmos em várias áreas que apresentam desempenho ótimo não são utilizados na prática pois possuem complexidade computacional e/ou custo proibitivos.
  - ***Oportunidades:*** existem muitos empregos na área de análise de dados e pesquisas inovadoras para a solução de problemas com ML.

# Mas então, o que é ML?

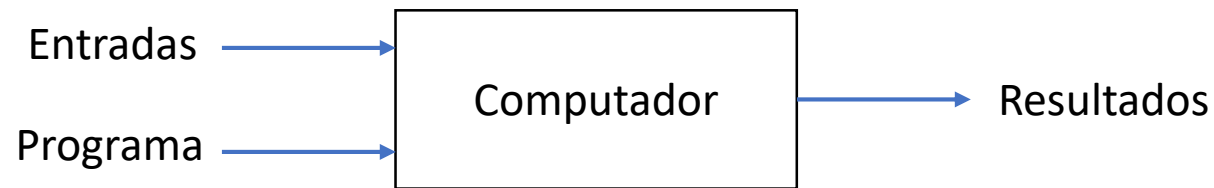


- É uma sub-área ou objetivo da inteligência artificial.
- O termo foi cunhado em 1959, por Arthur Samuel, que o definiu como o “*campo de estudo que dá aos computadores a habilidade de **aprender sem serem explicitamente programados***”.
- Uma outra definição interessante feita por Jojo John Moolayil é “*Aprendizado de máquina é o processo de **induzir** inteligência em uma máquina sem que ela seja explicitamente programada*”.
  - **Indução**: aprender um modelo ou padrão geral a partir de exemplos.
- Algoritmos de ML são **orientados a dados**, ou seja, eles aprendem automaticamente um padrão geral a partir de grandes volumes de dados.
- **Exemplo**: filtro de spam do Gmail.

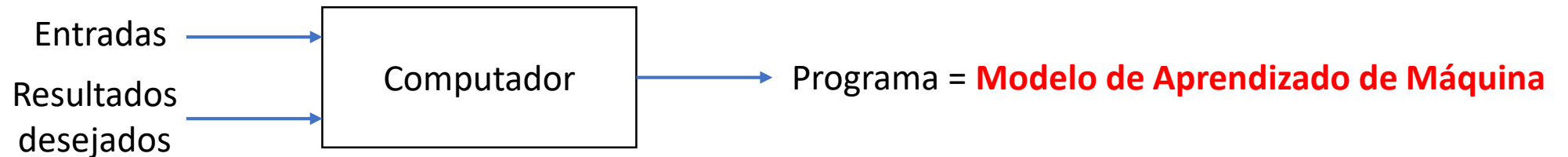
# O que é o Aprendizado de Máquina?

- “... sem serem explicitamente programados.”

## Programação Tradicional



## Aprendizado de Máquina



# Por que ML se tornou tão difundido?

Alguns dos principais motivos são:

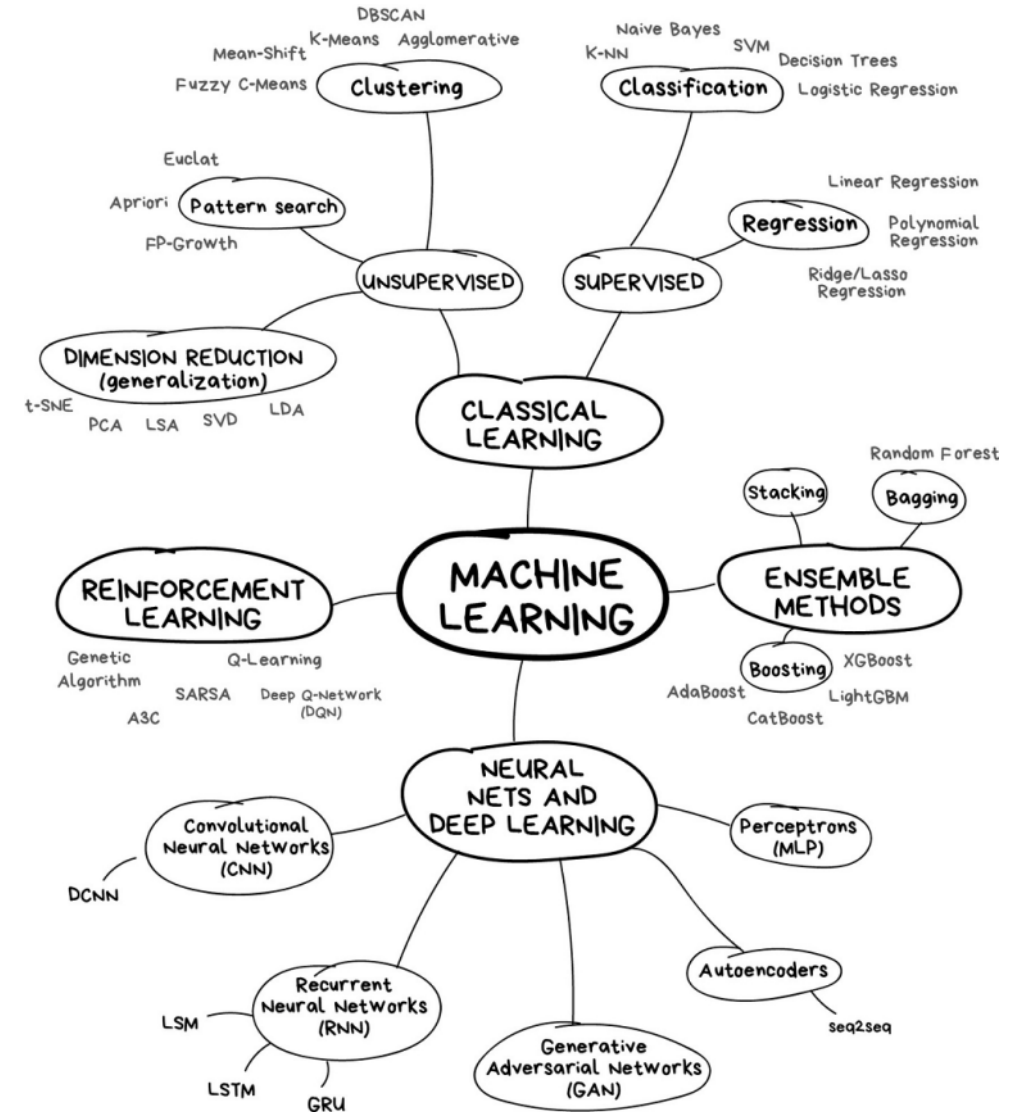
- Vivemos na era da informação. Nessa era, um volume sem precedentes de dados (de tera a petabytes) está disponível, impossibilitando sua análise por nós seres humanos.
- Porém, para modelos de ML isso não é um problema e sim uma solução, pois quanto mais dados melhor será o aprendizado.
- Hoje em dia, dados são preciosíssimos e a extração de novas informações (úteis) vale ouro.
- O surgimento de recursos computacionais poderosos tais como GPUs, FPGAs e CPUs com múltiplos cores.
- Surgimento de novas e eficientes estratégias/técnicas de treinamento (i.e., aprendizagem), e.g., deep-learning, reinforment-learning, etc.
- Existência de frameworks e bibliotecas poderosas que facilitam o desenvolvimento de soluções com ML.



# Tipos de Aprendizado de Máquina

Os algoritmos de aprendizado de máquina podem ser agrupados nas seguintes categorias:

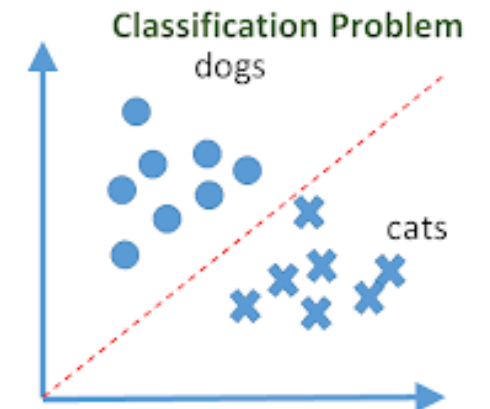
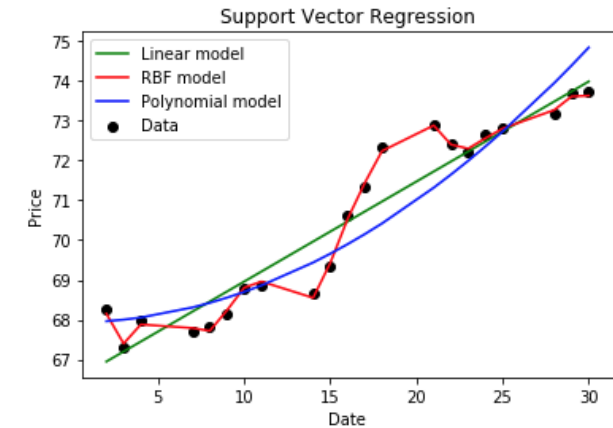
- Supervisionado
- Não-Supervisionado
- Semi-Supervisionado
- Por Reforço
- Metaheurístico





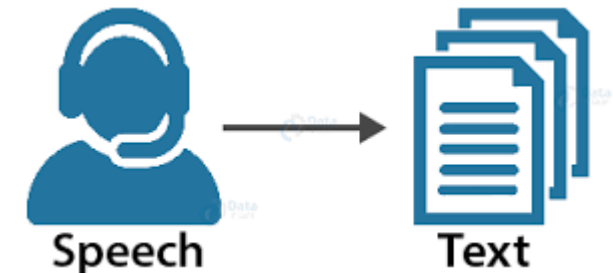
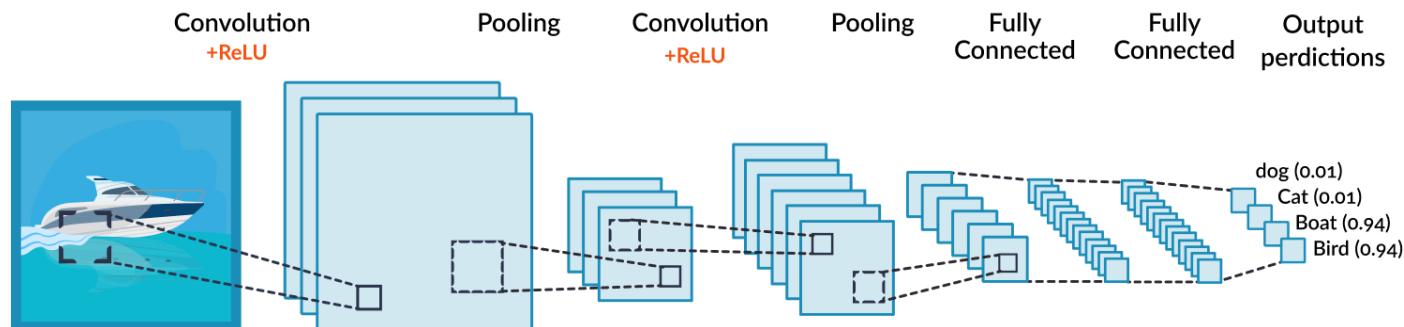
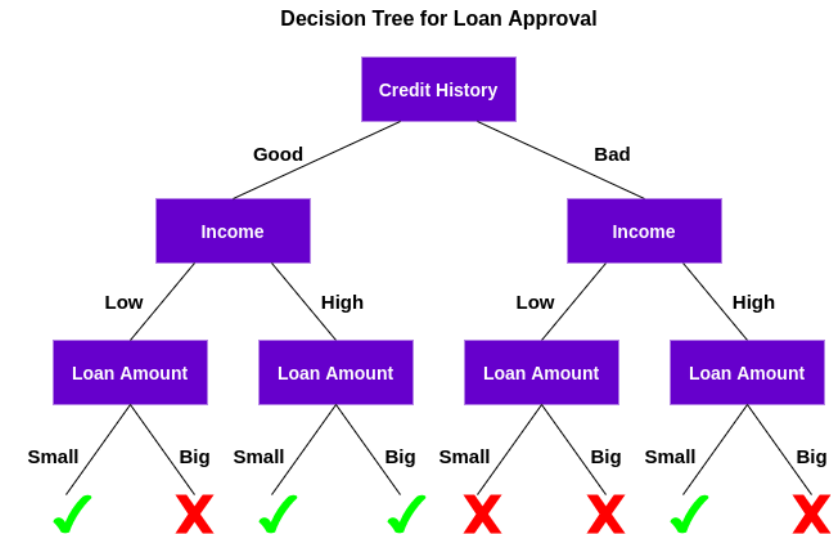
# Aprendizado Supervisionado

- No aprendizado supervisionado a máquina sabe o que aprender, ou seja, ela tem acesso às respostas esperadas.
- Neste tipo de aprendizado, os dados ou exemplos de treinamento incluem os **atributos**,  $x$ , que são a entrada do algoritmo de ML e as **soluções desejadas**,  $y$ , (i.e., as respostas corretas), chamadas de **rótulos** (ou *labels*, do Inglês).
- **Tarefa:** os modelos supervisionados de ML devem **aprender** uma função que mapeie as entradas  $x$  nas saídas  $y$ , ou seja,  $y = f(x)$ .
- Esse tipo de aprendizado pode ser dividido em problemas de **Regressão** e **Classificação**.
  - **Regressão:** o rótulo,  $y$ , pertence a um **conjunto infinito** de valores, i.e., números reais.
  - **Classificação:** o rótulo,  $y$ , pertence a um **conjunto finito** de valores, i.e., conjunto finito de classes.



# Principais Algoritmos para Aprendizado Supervisionado

- Regressão linear.
- Regressão logística.
- Árvores de Decisão (*Decision Trees*).
- Florestas Aleatórias (*Random Forests*).
- k vizinhos mais próximos (*k-nearest neighbors* - k-NN).
- Máquinas de Vetores de Suporte (*Support Vector Machines* - SVMs).
- Redes Neurais Artificiais.





# Aprendizado Não-Supervisionado

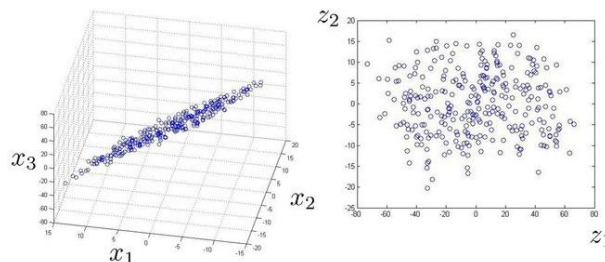
- Neste tipo de aprendizado, as máquinas não são informadas sobre o que aprender. Elas só recebem os exemplos de treinamento,  $x$ .
- Neste caso, os algoritmos ***aprendem/descobrem padrões*** (muitas vezes ocultos) presentes nos dados de entrada sem a presença de rótulos.
- **Tarefa:** os modelos devem ***aprender/descobrir*** padrões desconhecidos se baseando apenas nos exemplos de entrada.
- Trata problemas de clusterização, redução de dimensionalidade, detecção de anomalias (*outliers*) e aprendizado de regras de associação.

Clusterização

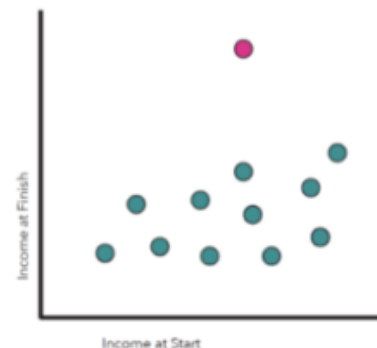


Redução de dimensionalidade

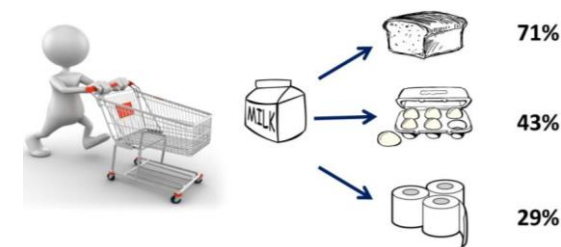
Reduce data from 3D to 2D



Detecção de Anomalias



Regras de associação

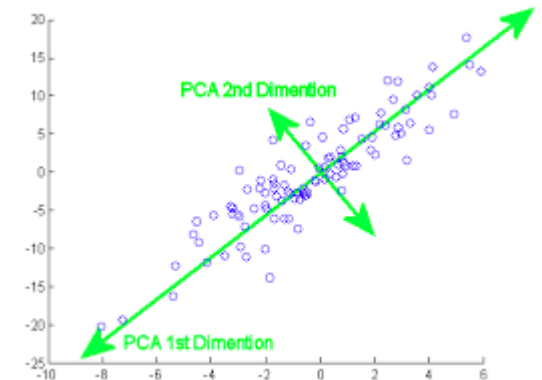
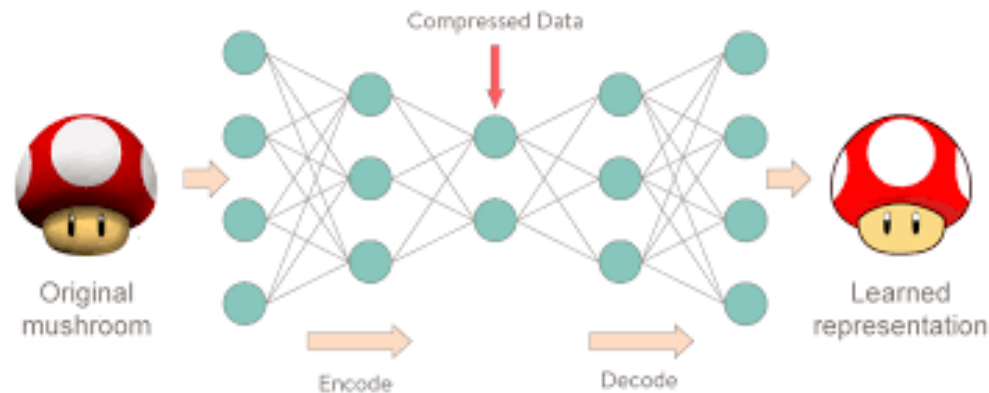
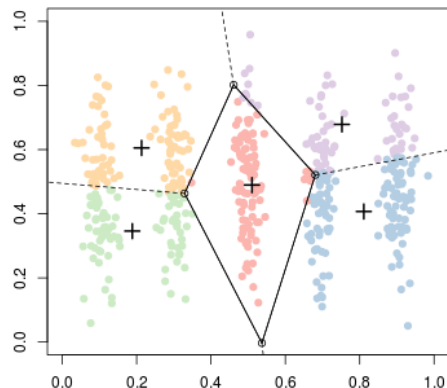


Of transactions that included milk:

- 71% included bread
- 43% included eggs
- 29% included toilet paper

# Principais Algoritmos para Aprendizado Não-Supervisionado

- k-médias (*k-means*).
  - Particiona os atributos em  $k$  clusters (ou grupos) distintos com base na distância ao centroide de um cluster.
- Redes Neurais Artificiais, e.g., auto-encoders.
  - Os autoencoders são usados para redução ou aumento de dimensionalidade.
- Análise de Componentes Principais (*Principal Component Analysis - PCA*).
  - Redução de dimensionalidade.



# Aprendizado Semi-Supervisionado

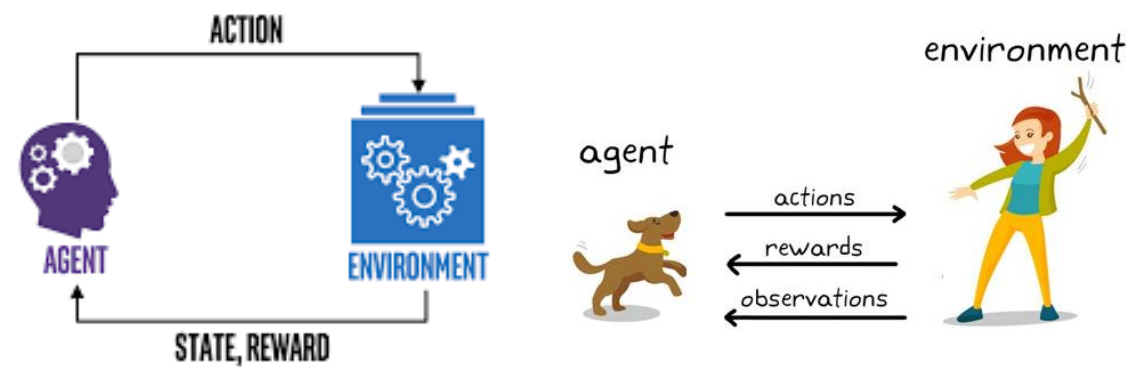
- Neste tipo de aprendizado, as máquinas tem acesso a exemplos com e sem rótulos.
- Geralmente envolve uma ***pequena quantidade de dados*** rotulados e uma ***grande quantidade de dados não-rotulados***.
- É de grande ajuda em casos onde se ter uma grande quantidade de dados rotulados é muito demorado, caro ou complexo.
- Algoritmos de aprendizagem semi-supervisionada são o resultado da combinação de algoritmos supervisionados e não-supervisionados.
- Uma maneira de realizar aprendizado semi-supervisionado é combinar, por exemplo, algoritmos de ***clustering*** e ***classificação***.

# Aprendizado Semi-Supervisionado

- **Exemplo:** Como **classificaríamos** milhões de textos **não-rotulados** da internet em categorias como economia, esportes, política, entretenimento, etc.?
- Poderíamos usar **clustering** para agrupar a quantidade massiva de textos e usar apenas os exemplos mais representativos de cada **cluster** (quantidade bem menor de textos) para **rotular manualmente**.
- Esses **exemplos rotulados** são usados para treinar um **classificador**.
- Após o treinamento, o **classificador** classifica automaticamente todos os textos.



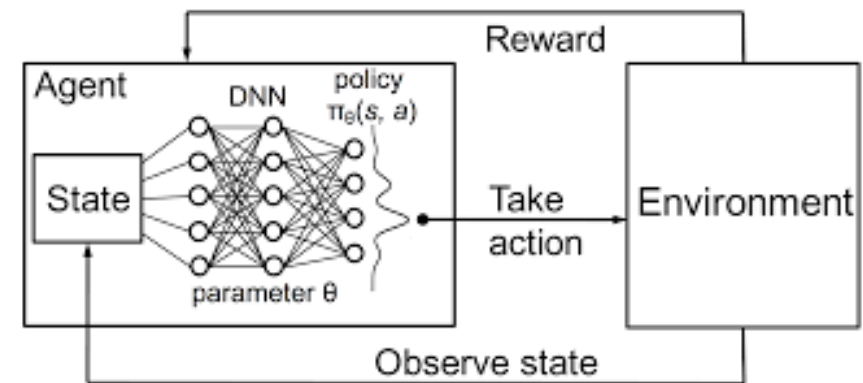
# Aprendizado Por Reforço



- Abordagem totalmente diferente das anteriores pois ***não temos exemplos de treinamento***, sejam eles rotulados ou não.
- O algoritmo de aprendizado por reforço, chamado de ***agente*** nesse contexto, aprende como se comportar em um ***ambiente*** através de interações do tipo ***tentativa e erro***.
- O ***agente*** observa o ***estado*** do ***ambiente*** em que está inserido, seleciona e executa ***ações*** e recebe uma ***recompensa*** (ou ***reforço***) em consequência das ***ações*** tomadas.
- Seguindo estes passos, o agente deve aprender por si só qual a melhor ***estratégia***, chamada de ***política***, para obter a maior recompensa possível ao longo do tempo.
- Uma ***política*** define qual ***ação*** o ***agente*** deve escolher quando estiver em uma determinada situação, ou seja, o ***estado*** do ***ambiente***.
- Uma ***política*** é uma função que mapeia os ***estados*** do ***ambiente*** em ***ações*** que o ***agente*** deve tomar.

# Principais Algoritmos de Aprendizado Por Reforço

- Q-Learning
  - Usado para encontrar uma **política** ótima de seleção de **ações** usando a **função-Q**.
  - **Q**, ou **valor-Q**, representa a **qualidade** de uma dada **ação** em um determinado **estado**.
- Deep Q-Learning
  - Junção de Deep Learning + Q-Learning.
  - Redes neurais profundas possibilitam que Q-Learning seja aplicado a problemas com número gigantesco de **estados** e **ações**.
  - O Q-Learning tabela a **função-Q**, já o Deep Q-Learning encontra uma **função** que aproxime a **função-Q**.



# Aprendizado Metaheurístico

- Uma **metaheurística** é um método **heurístico** usado para resolver de forma genérica problemas de otimização.
  - **Heurística** é um método ou processo criado com o objetivo de encontrar soluções, de **forma rápida** e muitas vezes **sub-ótimas**, para um problema.
- **Metaheurísticas** são geralmente aplicadas a problemas para os quais não se conhece um algoritmo eficiente (e.g., problemas NP-completos).
- Características das metaheurísticas:
  - não garantem que uma solução globalmente ótima seja encontrada, mas podem encontrar uma solução suficientemente boa.
  - são estratégias que orientam o processo de busca.
  - não são específicas do problema, ou seja, são genéricas.
  - funcionam bem mesmo com capacidade de computação limitada.



# Principais Algoritmos de Aprendizado Metaheurístico

- Algoritmo Genético (*Genetic Algorithm* - GA).
  - Inspirados pelo processo de seleção natural.
- Otimização por enxame de partículas (*Particle Swarm Optimization* - PSO).
  - Inspirado no comportamento de cardumes de peixes e de bandos de pássaros
- Otimização da colônia de formigas (*Ant Colony Optimization* - ACO).
  - Inspirado no comportamento das formigas ao saírem de sua colônia para encontrar comida.

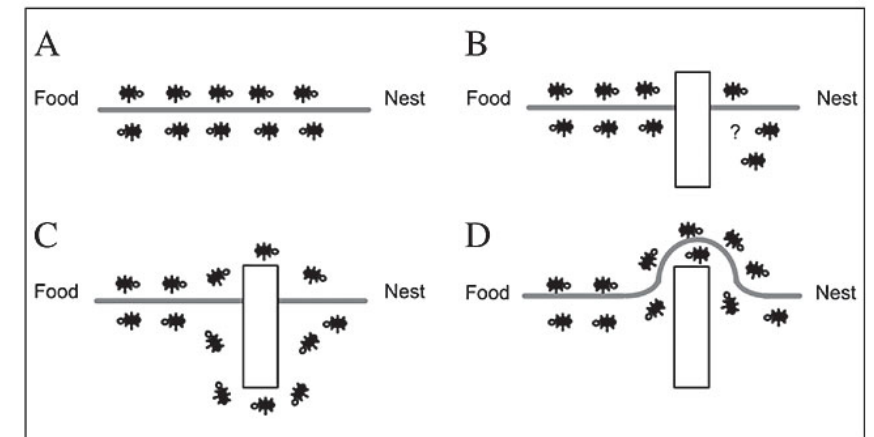
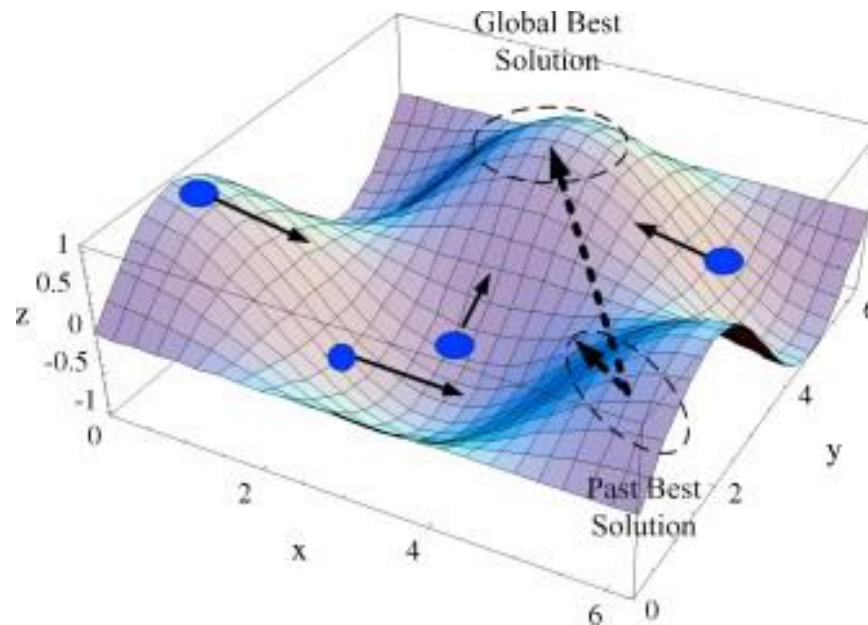
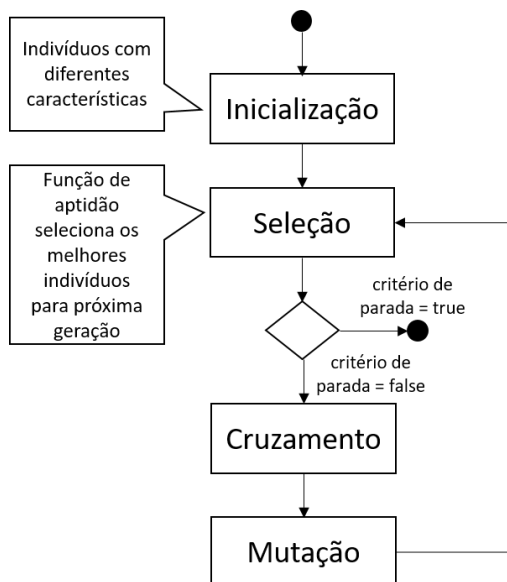


Figure 2. A. Ants in a pheromone trail between nest and food; B. an obstacle interrupts the trail; C. ants find two paths to go around the obstacle; D. a new pheromone trail is formed along the shorter path.



# Tipos de Treinamento

Uma outra forma de se classificar algoritmos de ML é com relação se eles podem ser treinados incrementalmente ou não. Assim, os algoritmos podem ser divididos em algoritmos com treinamento:

- **incremental (online).**
- **em batelada (batch).**

# Treinamento incremental

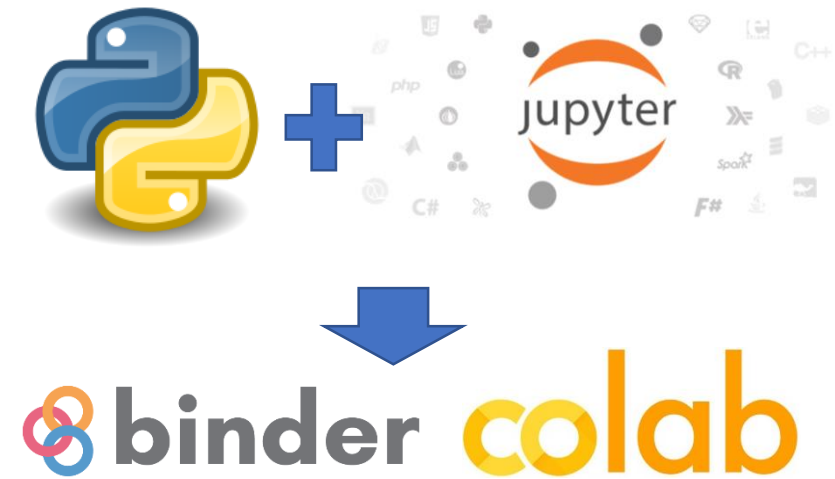
- Neste tipo de treinamento, o algoritmo ***aprende incrementalmente***:
  - Os exemplos de treinamento são apresentados ***sequencialmente um-a-um*** ou em ***pequenos grupos*** chamados de mini-batches (ou mini-lotes).
- Cada iteração de treinamento é rápida possibilitando que o sistema aprenda sobre novos dados à medida que eles chegam.
- Ótima opção para casos onde os dados chegam como um fluxo contínuo ou se tem recursos computacionais limitados.
- Entretanto, como não há pré-processamento/análise, dados corrompidos ou com problemas afetam a performance do sistema.

# Treinamento em batelada

- Neste tipo de treinamento, o algoritmo é ***treinado com todos os exemplos disponíveis***.
- É um tipo de treinamento simples, de fácil implementação e obtém ótimos resultados.
- Dados podem ser pré-processados/analísados, evitando assim, dados corrompidos ou com problemas.
- O treinamento é demorado e utiliza muitos recursos computacionais (e.g., CPU, memória) quando comparado ao treinamento incremental.
- Para treinar com novos exemplos é necessário iniciar o treinamento do zero.
- Se a quantidade de dados do conjunto de treinamento for muito grande pode ser impossível treinar em batelada.

# Executando códigos na nuvem

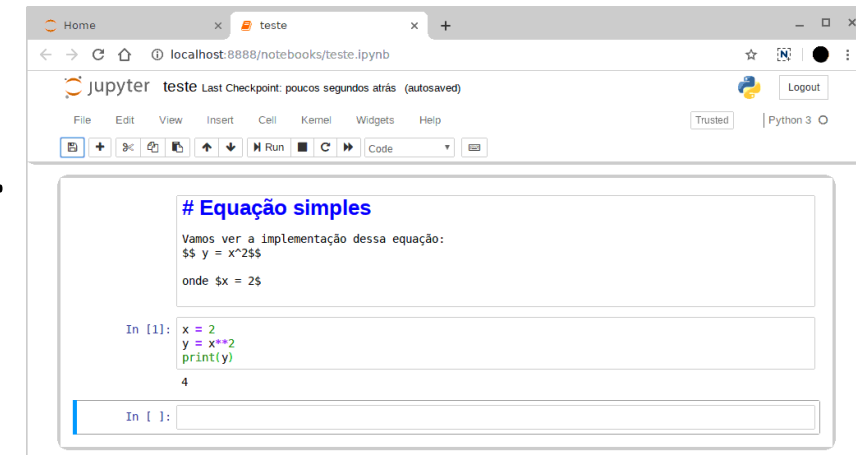
- Durante o curso, usaremos **Python** como linguagem de programação.
- Utilizaremos **notebooks Jupyter** para programar.
  - Eles são documentos virtuais usados para criar e compartilhar código juntamente com equações, gráficos e texto.
  - **Notebooks** permitem uma maneira interativa de programar e documentar o código.
- Para executar estes **notebooks**, utilizaremos o **Binder** ou **Google Colaboratory**, que são ambientes computacionais (i.e., servidores) interativos e gratuitos.
- Portanto, **vocês não precisam instalar nada**, apenas terem um navegador web e conexão com a internet.



# Binder



- **Binder**: aplicação web gratuita que permite a criação e edição de *notebooks* em navegadores web.
- Suporta a execução de várias linguagens de programação: Python, C++, C#, PHP, Julia, R, etc.
- Algumas desvantagens do **Jupyter** são:
  - Poucos servidores disponíveis.
  - Depois de algum tempo inativo, a máquina virtual executando seu *notebook* se desconecta e você pode perder seu código.
- URL (através do Jupyter): <https://jupyter.org/>



# Google Colaboratory (Colab)



- **Colab**: outra aplicação web gratuita, baseada no Jupyter, que permite a criação e edição de ***notebooks*** em navegadores web.
- É um produto da Google.
- Por hora, suporta apenas a execução de códigos escritos em Python.
- Vantagens sobre o Jupyter:
  - Maior número de servidores.
  - Inicialização e processamento do código mais rápidos.
  - Fornece GPUs e TPUs gratuitamente.
  - Compartilhamento de notebooks entre usuários é mais fácil.
  - Notebooks podem ser salvos no seu Google Drive, evitando que você perca seu código.
- URL: <https://colab.research.google.com/>

# Exemplos de uso dos notebooks com Binder e Colab



# Histograma

```
import numpy as np
import matplotlib.pyplot as plt
```

```
np.random.seed(42) # Reseta o gerador PN.
```

```
N = 1000000 # Número de exemplos.
```

```
x = np.random.rand(N,1) # vetor coluna x, com dimensão Nx1.
w = np.sqrt(0.01)*np.random.randn(N,1) # vetor coluna w, com dimensão Nx1.
```

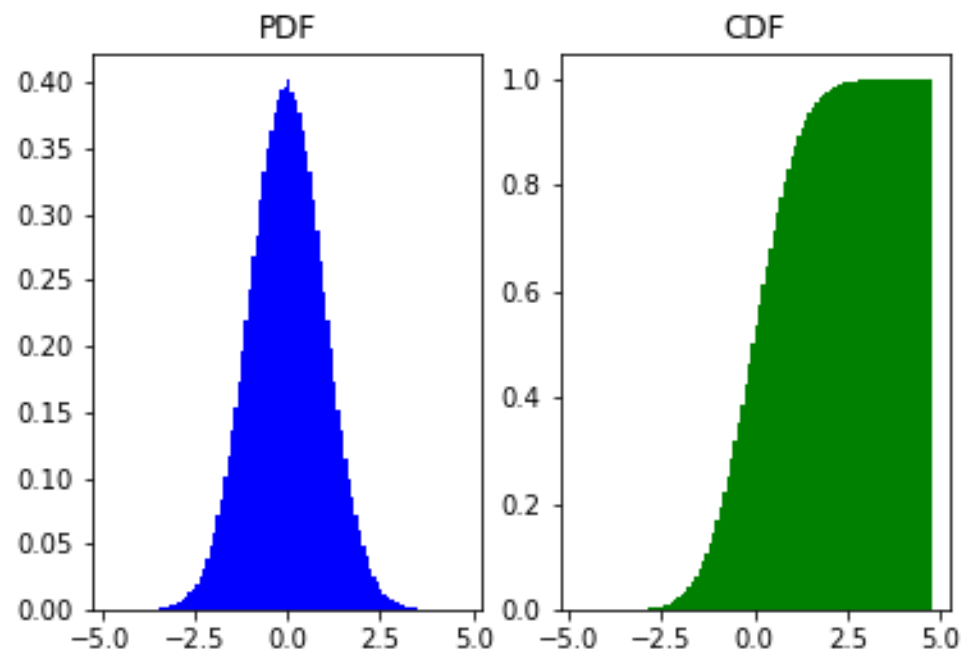
```
y = 1 + 2*x # Função original ou verdadeira.
y_noisy = y + w # Versão ruidosa de y.
```

```
plt.plot(x, y_noisy, '.b', label='Função ruidosa')
plt.plot(x, y, 'k', label='Função original')
plt.xlabel('x', fontsize=14)
plt.ylabel('y', fontsize=14)
plt.legend()
plt.show()
```

```
# histograma (pdf)
plt.subplot(1, 2, 1)
plt.title('PDF')
plt.hist(y_noisy, bins=100, density=True, color='b')
# CDF empírica
plt.subplot(1, 2, 2)
plt.title('CDF')
plt.hist(y_noisy, bins=100, density=True, color='g', cumulative=True)

plt.savefig('histogram.png') # salva figura em arquivo
```

## Exemplo (binder): Histograma.ipynb



## Exemplo (colab): Histograma.ipynb



# Figura 3D

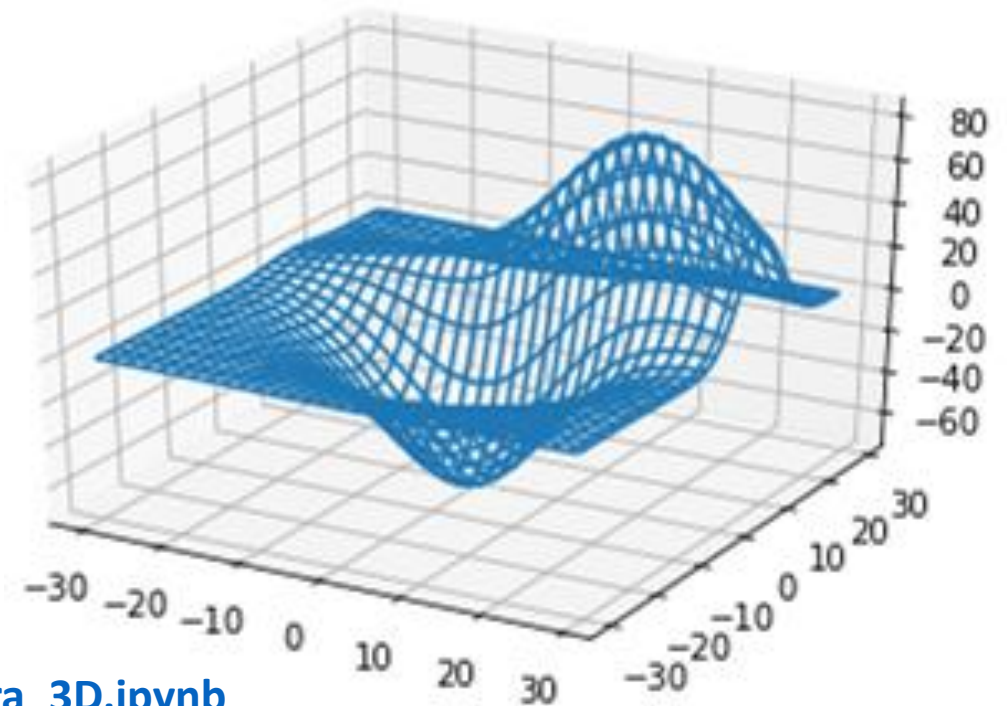
```
import matplotlib.pyplot as plt
# facilita visualizacao de figuras 3D
from mpl_toolkits.mplot3d import axes3d # graficos 3D sao habilitados importando axes3d
```

```
# para figuras interativas usar "notebook" ao inves de "inline"
%matplotlib notebook
```

```
ax = plt.subplot(111, projection='3d')
X, Y, Z = axes3d.get_test_data(0.1)
ax.plot_wireframe(X, Y, Z)
```

```
# salva figura em arquivo
plt.savefig('figura3d.png')
```

[Exemplo \(binder\): Figura 3D.ipynb](#)



[Exemplo \(colab\): Figura 3D.ipynb](#)

# Ajuste de curva com Redes Neurais

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neural_network import MLPRegressor # importa classe MLPRegressor do modulo neural network

%matplotlib inline

x = np.arange(-10, 10, 0.1)

# dados originais
y = 12 + 3 * np.exp(-0.05*x) + 1.4 * np.sin(1.2*x) + 2.1 * np.sin(-2.2*x + 3)

# faz com que o gerador de numeros aleatorios sempre forneça os mesmos valores
np.random.seed(42)

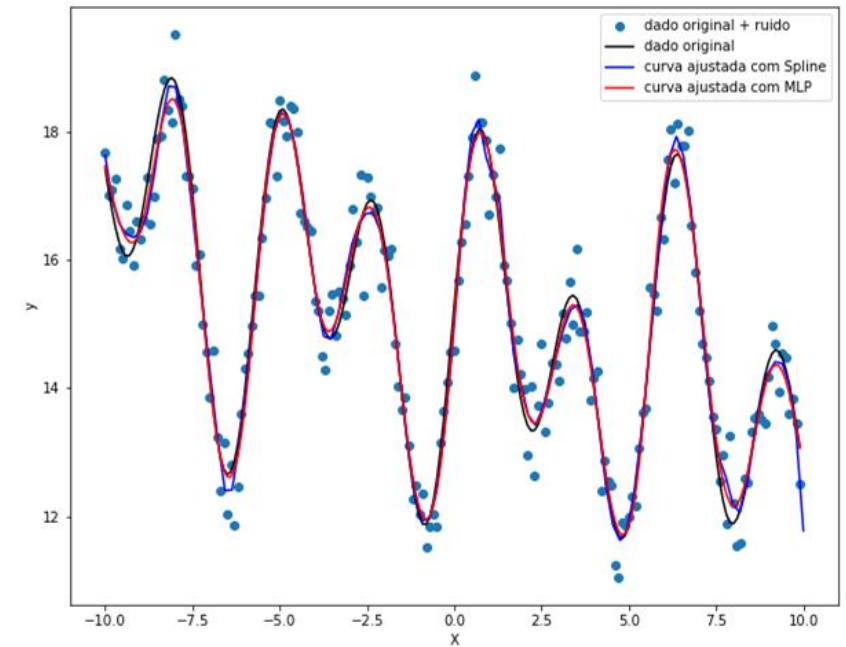
# adicionando ruido aos dados originais
y_noise = y + np.random.normal(0, 0.5, size = len(y))

# trata o ajuste de curva como um problema de regressao e treina um modelo para que se ajuste aos dados.
mlp = MLPRegressor(hidden_layer_sizes=(50,25,10), max_iter=10000, solver='lbfgs', alpha=0.9, activation='tanh')
yfit = mlp.fit(x[:, None], y_noise).predict(x[:, None])

plt.figure()
plt.plot(x, y_noise, 'o', label = 'dados original + ruido')
plt.plot(x, y, 'k', label = 'dados original')
plt.plot(x, yfit, '-r', label = 'curva ajustada com MLP', zorder = 10)
plt.legend()
plt.xlabel('X')
plt.ylabel('y')

# salva figura em arquivo
plt.savefig('mlp_regression.png')
```

[Exemplo \(binder\): Ajuste de curva com Redes Neurais.ipynb](#)



[Exemplo \(colab\): Ajuste de curva com Redes Neurais.ipynb](#)

# Referências

- [1] Stuart Russell and Peter Norvig, *“Artificial Intelligence: A Modern Approach,”* Prentice Hall Series in Artificial Intelligence, 3rd ed., 2015.
- [2] Aurélien Géron, *“Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems”*, 1st ed., O'Reilly Media, 2017.
- [3] Joseph Misiti, *“Awesome Machine-Learning,”* on-line data base with several free and/or open-source books (<https://github.com/josephmisiti/awesome-machine-learning>).
- [4] Andriy Burkov, *“The Hundred-Page Machine-Learning Book,”* Andriy Burkov 2019.
- [5] C. M. Bishop, *“Pattern Recognition and Machine Learning,”* Springer, 1st ed., 2006.
- [6] S. Haykin, *“Neural Networks and Learning Machines,”* Prentice Hall, 3ª ed., 2008.
- [7] Coleção de livros,  
<https://drive.google.com/drive/folders/1lylIMu1w6POBhrVnw11yqXXy6BjC439j?usp=sharing>

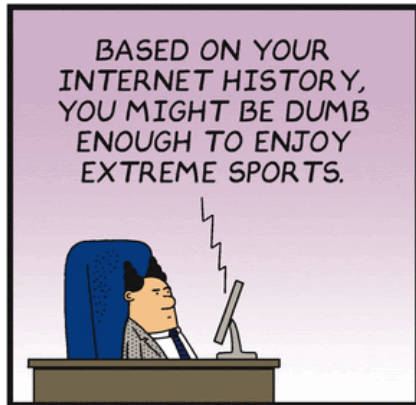
# Avisos

- Entregas de exercícios devem ser feitas no MS Teams.
  - Se atentem às datas/horários de entrega no MS Teams.
- Todo material do curso será disponibilizado no MS Teams e no GitHub:
  - [https://github.com/zz4fap/t319\\_aprendizado\\_de\\_maquina](https://github.com/zz4fap/t319_aprendizado_de_maquina)
- Horário de Atendimento
  - Professor: Segundas-feiras das 18:30 às 19:30 e Quartas-feiras das 15:30 às 16:30 via MS Teams.
  - Monitora (Bruna): Todas as Quintas-feiras das 17:30 às 18:30.

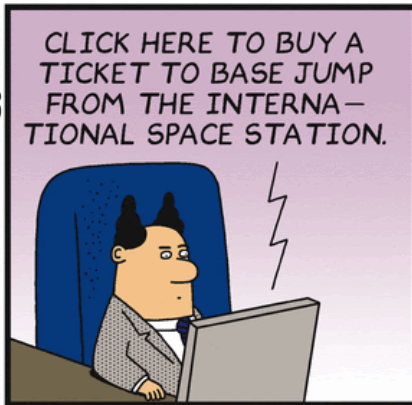
# Tarefas

- **Quiz:** “*T319 - Quiz - Introdução*” que se encontra no MS Teams.
- **Exercício Prático:** [Laboratório #1](#).
  - Pode ser baixado do MS Teams ou do GitHub.
  - Pode ser respondido através do link acima (na nuvem) ou localmente.
  - [Instruções para resolução e entrega dos laboratórios](#).
  - **Laboratórios podem ser feitos em grupo, mas as entregas devem ser individuais.**

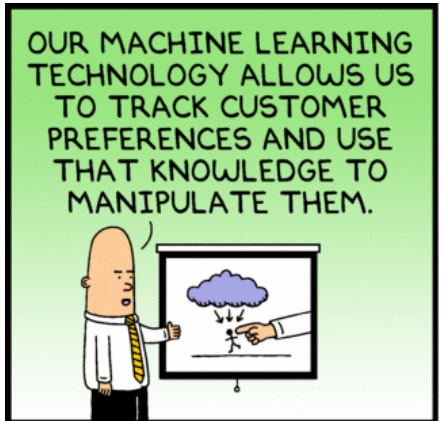
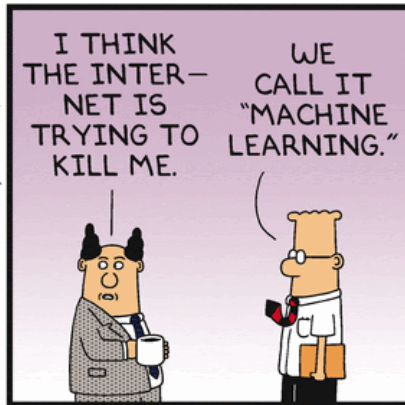
Obrigado!



Dilbert.com DilbertCartoonist@gmail.com



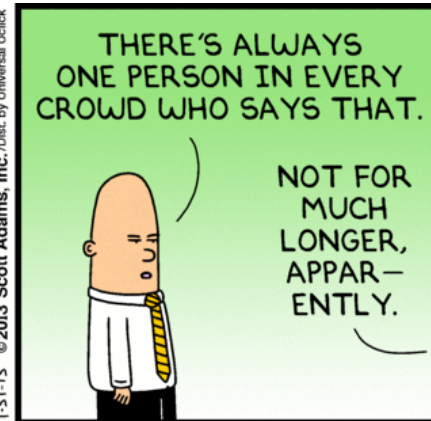
2-2-13 ©2013 Scott Adams, Inc./Dist. by Universal Uclick



Dilbert.com DilbertCartoonist@gmail.com



1-31-13 ©2013 Scott Adams, Inc./Dist. by Universal Uclick



www.dilbert.com scottadams@aol.com



©2003 United Feature Syndicate, Inc.

