# Introdução ao Machine Learning

Machine Learning é o estudo do aprendizado de máquina, essa ciência elabora algoritmos para que a máquina encontre relações entre dados de entrada e saída ou só entrada, e após estabelecer relações possa prever situações.

Dentro ciência de IA existem algumas metodologias como:

# Aprendizado supervisionado

Quando possuímos dados de entrada e saída e portanto podemos treinar o algoritmo afim de fazer previsões assertivas. Existem ainda uma relação que indica a previsão de variáveis dependentes com base na análise de variáveis independentes. Nesse caso, dizemos que os **dados são anotados** com as respostas ou classes a serem previstas.

Dentre as técnicas mais conhecidas para resolver problemas de aprendizado supervisionado estão regressão linear, regressão logística, redes neurais artificiais, máquina se suporte vetorial (ou máquinas kernel), árvores de decisão, k-vizinhos mais próximos e Bayes ingênuo. Aprendizado de máquina supervisionado é a área que concentra a maioria das aplicações bem sucedidas e onde a maioria dos problemas já estão bem definidos.

Ex: campeonato de futebol, com base na relação dos dados de entrada e saída é possível prever possíveis resultados futuros dos jogos:

|  |  |
| --- | --- |
| Dados de entrada | Dados de saída |
| Composição de um time de futebol | 0 se perdeu ou empatou |
| Características de cada jogador | 1 se ganhou |
| Número de Gol |  |

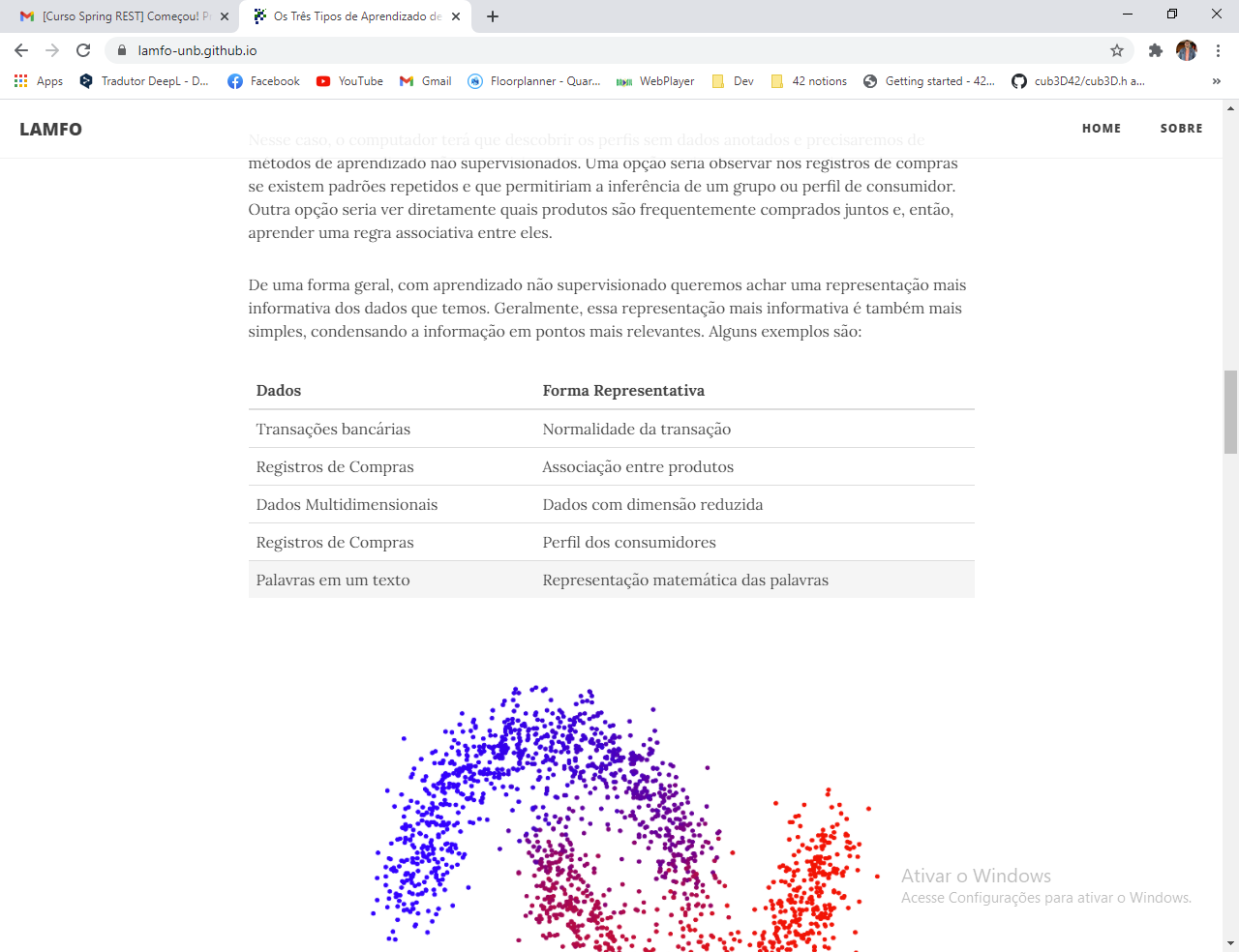


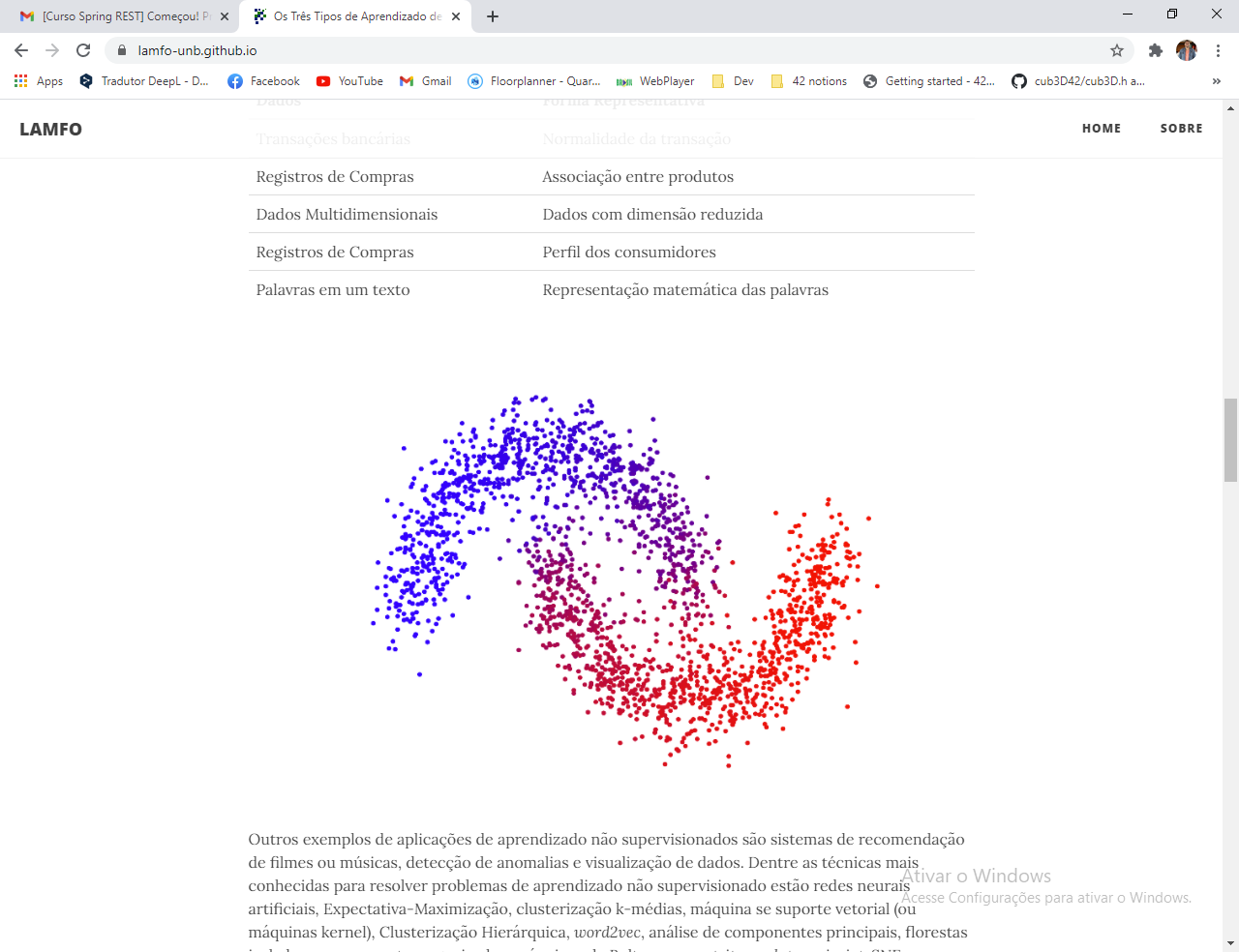
# Aprendizado não supervisionado

Mas nem todos os problemas podem ser resolvidos desta forma. Em alguns casos, conseguir dados anotados é extremamente custoso ou até mesmo impossível. Por exemplo, imagine que você é dono de um comércio e quer conhecer o perfil dos seus consumidores. Pode haver um perfil de consumidor que sempre compra vinho e queijo ou que compra carne e carvão ou ainda leite em pó e fralda. Se esse for o caso, colocar esses produtos em prateleiras distantes pode aumentar vendas, já que aumentará o tempo e o percurso do cliente no mercado. No entanto, nesse caso não estamos anotado para cada compra à qual perfil o consumidor pertence. Mais ainda, sequer sabemos quantos perfis de consumidores há.

Nesse caso, o computador terá que descobrir os perfis sem dados anotados e precisaremos de métodos de aprendizado não supervisionados. Uma opção seria observar nos registros de compras se existem padrões repetidos e que permitiriam a inferência de um grupo ou perfil de consumidor. Outra opção seria ver diretamente quais produtos são frequentemente comprados juntos e, então, aprender uma regra associativa entre eles.

De uma forma geral, com aprendizado não supervisionado queremos achar uma representação mais informativa dos dados que temos. Geralmente, essa representação mais informativa é também mais simples, condensando a informação em pontos mais relevantes. Alguns exemplos são:





Outros exemplos de aplicações de aprendizado não supervisionados são sistemas de recomendação de filmes ou músicas, detecção de anomalias e visualização de dados. Dentre as técnicas mais conhecidas para resolver problemas de aprendizado não supervisionado estão redes neurais artificiais, Expectativa-Maximização, clusterização k-médias, máquina se suporte vetorial (ou máquinas kernel), Clusterização Hierárquica, word2vec, análise de componentes principais, florestas isoladoras, mapas auto-organizados, máquinas de Boltzmann restritas, eclat, apriori, t-SNE. Problemas de aprendizado não supervisionado são consideravelmente mais complicados do que problemas de aprendizado supervisionado, principalmente porque não temos a reposta anotada nos dados. Como consequência, é extremamente complicado e controverso avaliar um modelo de aprendizado não supervisionado e esse tipo de modelo está na fronteira do conhecimento em aprendizado de máquina.

# Aprendizagem por reforço

A terceira abordagem de aprendizagem de máquinas é a chamada “aprendizagem por reforço”, em que a máquina tenta aprender qual é a melhor ação a ser tomada, dependendo das circunstâncias na qual essa ação será executada.

O futuro é uma variável aleatória: como não se sabe a priori o que irá acontecer, é desejável uma abordagem que leve em consideração essa incerteza, e consiga incorporar as eventuais mudanças no ambiente do processo de tomada da melhor decisão. Essa ideia de fato deriva do conceito de “aprendizagem por reforço” da psicologia, no qual uma **recompensa** ou **punição** é dada a um agente, dependendo da decisão tomada; com o tempo e a repetição dos experimentos, espera-se que o agente consiga associar as ações que geram maior recompensa para cada situação que o ambiente apresenta, e passe a evitar as ações que geram punição ou recompensa menor. Na psicologia, essa abordagem é chama de behaviorismo e tem B. F. Skinner como um dos principais expoentes, um famoso psicólogo que, dentre outros experimentos, usou a ideia de recompensas e punições para treinar pombos para conduzir mísseis na Segunda Guerra Mundial.

Essa mesma ideia é vista no aprendizado de máquinas: a máquina observa um “estado da natureza” dentre o conjunto de cenários futuros possíveis e, com base nisso, escolhe uma ação a se tomar e recebe a recompensa associada a essa ação específica e nesse estado específico obtendo, assim, a informação dessa combinação. O processo se repete até que a máquina seja capaz de escolher a melhor ação a se tomar para cada um dos cenários possíveis a serem observados no futuro.

Ilustrando com um exemplo bem simples: suponha que você queira adestrar seu cão a se sentar ao seu comando por essa abordagem. De primeira, dificilmente o animal executará o comando requerido, e você responde a isso dando um “reforço negativo” (punição), repreendendo-o verbalmente, com suas expressões faciais ou mesmo com uma pancada de jornal (ou algo mais hostil, a depender do seu temperamento…). Quando o cão se aproxima do que deveria fazer, você pode dar “reforços positivos” como sinais de aprovação ou incentivo. Se o cão de fato sentar após o comando, você lhe dá uma recompensa - um biscoitinho, por exemplo. Com várias repetições desse mesmo experimento, espera-se que, com o tempo, o cão passe a associar a relação de “causa-efeito” entre o comando e a recompensa a ser recebida, e com isso “aprenda” a obedecer a esse comando. O famoso experimento do “cão de Pavlov” ilustra bem esse paradigma de aprendizagem. Ivan Pavlov foi um cientista russo notório por apresentar a ideia do “reflexo condicionado”, baseado no seguinte experimento: apresentando um pedaço de carne a um cão, o animal passa a salivar, desejando o alimento. Em vez de apresentar apenas a carne, Pavlov soava uma campainha sempre que isso acontecia; com a repetição, o cão passava a associar os dois “estímulos” (carne e campainha) e salivar assim que ouve a campainha.

Essa ideia é bastante versátil quando a transportamos para o âmbito da ciência de dados: em vez de adestrar cachorros, podemos por exemplo construir uma máquina que monta portfolios no mercado financeiro e que ajusta a combinação de ativos comprados/vendidos a depender da “recompensa” (retorno financeiro) da carteira anterior e da evolução (“estados”) do mercado. Ou ainda um automóvel que dirige “sozinho”, que toma decisões dependendo do cenário que observa ao redor, recebendo recompensas negativas quando colide com o ambiente ou com outros veículos, e com repetidas etapas, aos poucos “aprenda” a contornar os obstáculos.

Vamos mais além: voltando ao exemplo do adestramento de cães, suponha que, depois de tê-lo ensinado a sentar com sucesso, você queira fazer um teste de obediência - fazer o cão ficar sentado enquanto você anda para trás, até chegar a uma distância de cinco metros dele. Caso o cão se sente, você dá a ele o biscoitinho (assim como antes); mas caso ele não se levante até você ficar a cinco metros dele, você dá a ele uma recompensa maior (um pedaço de bife, por exemplo). Como o cão poderia ganhar a recompensa maior? Note que agora o cão tem uma “escolha difícil”: ele pode simplesmente sentar (já que ele já aprendeu essa tarefa) e ganhar sempre o biscoito, assim como pode “explorar” novas possibilidades - no caso, ficar parado mesmo com o dono longe - para ver se eventualmente não há uma recompensa maior ainda. O dono continuará fornecendo reforços positivos e negativos enquanto anda para trás, elogiando-o se ficar parado e repreendendo-o caso saia da posição; mas será que o animal está disposto a abrir mão da recompensa que ele já tem “garantida”?

Essa situação ilustra um trade-off: o animal pode escolher por “testar” novas combinações de ações ótimas em uma sequência de “estados” não realizadas anteriormente em busca de uma recompensa maior, mas não imediata (o chamado **“exploration”**); ou simplesmente se ater à recompensa que se obtém através de uma ação já conhecida (o chamado **“exploitation”**). Veja como essa ideia pode se estender facilmente para outros contextos; em finanças, por exemplo, reflete bem o grau de aversão ao risco de um agente econômico - o quanto ele está disposto a correr mais riscos em busca de retornos maiores, ou está disposto a correr menos risco e obter um retorno menor, porém mais seguro.

# Aprendizado Supervisionado - Regressão

Buscam prever um valor contínuo, com base em um conjunto de dados rotulados, ou seja deseja-se prever um valor numérico. Ex: consumo de um carro, tendo distância percorrida e velocidade.

É uma relação matemática.

### Variáveis independente

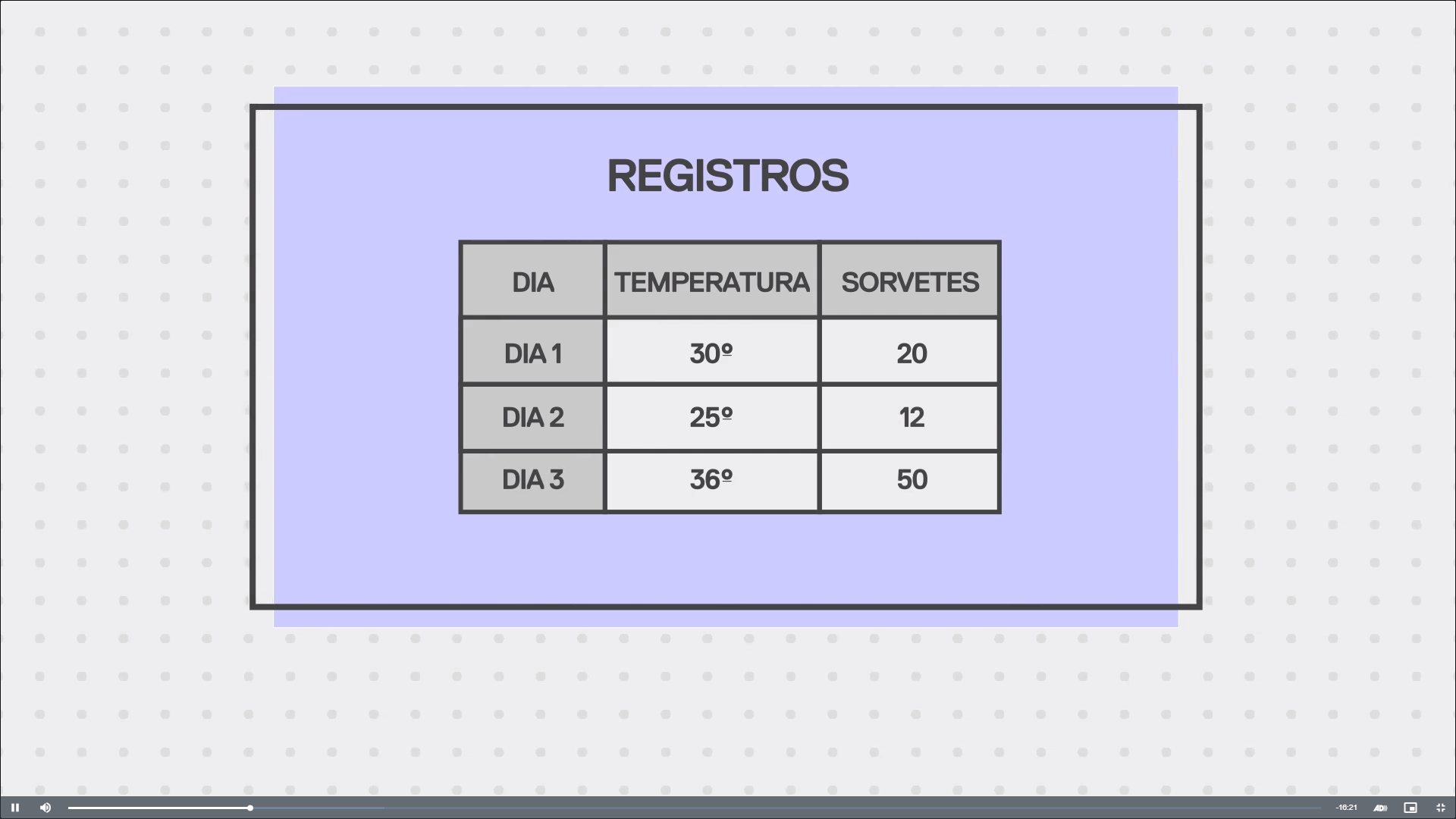
Não dependem de nenhuma informação para fazer sentido) Ex: Idade de uma pessoa.

### Variáveis dependentes

precisam de outras informações, dados de entrada para serem exibidas) Ex: valor do seguro de um carro, onde precisamos do ano do carro e idade do motorista para gerarmos.

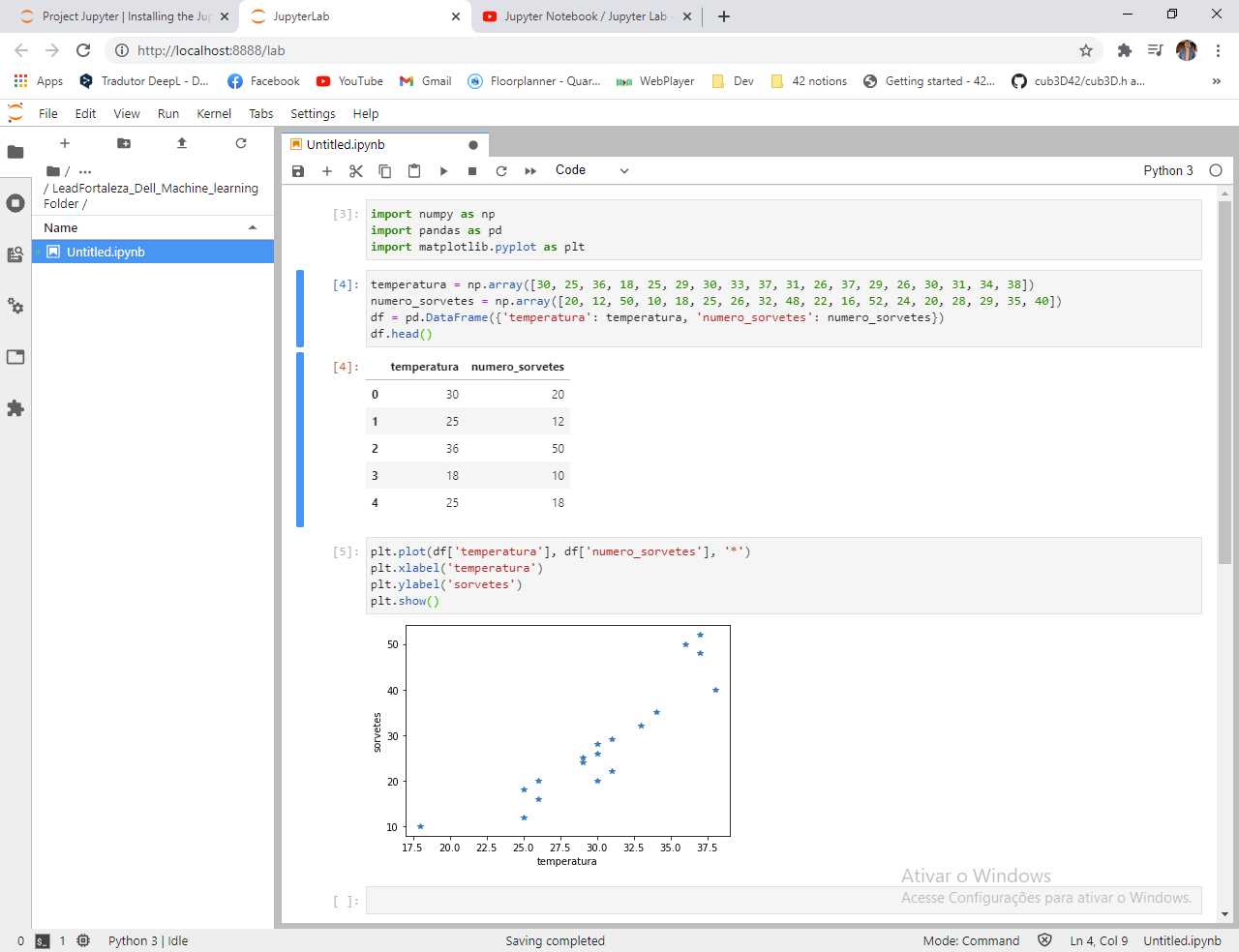
Para expressarmos uma variável independente utilizamos o X e a para variáveis independentes o Y. F(X) ou F(Y)

Ex: Vendedor de sorvete deseja verificar a relação entre temperatura e número de sorvetes vendidos. Nesse caso a temperatura é uma variável independente, já o número de sorvetes é dependente:



## Regressão Simples

É quando uma única variável independente é usada para obter uma variável dependente. Ex Sorveteiro acima.

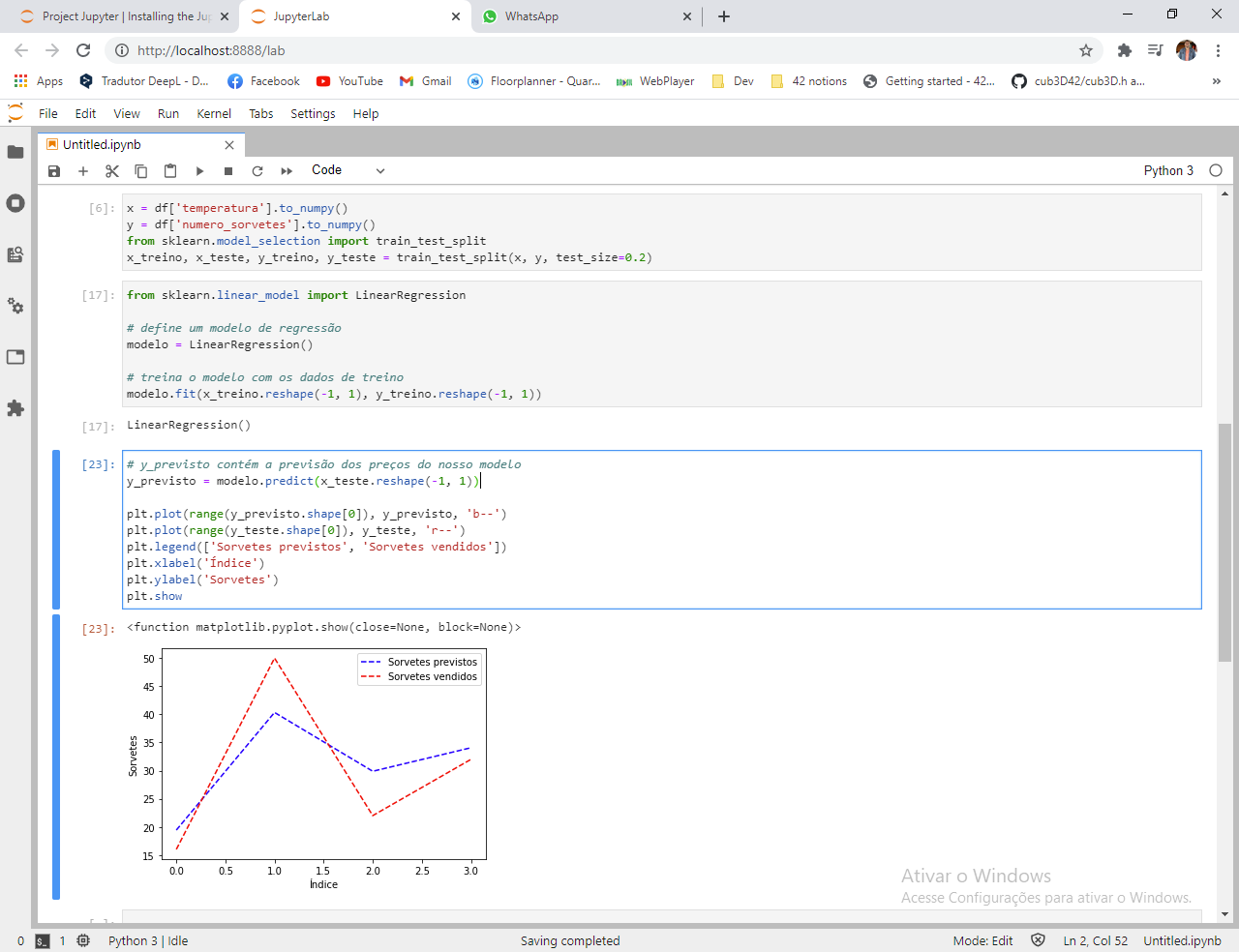


No ex. abaixo temos a utilização das bibliotecas:

1. **Numpy** - np.array – criar os array
2. **Pandas**- pd.DataFrame[definimos as colunas das tabelas] e df.head()[para mostrar a tabela]
3. **Matplotlib** - criar gráficos, plt.plot(defini as variáveis do gráfico x, y, e caracter), plt.xlabel(‘temperatura’), plt.ylabel(‘sorvetes’), plt.show() mostrar o gráfico

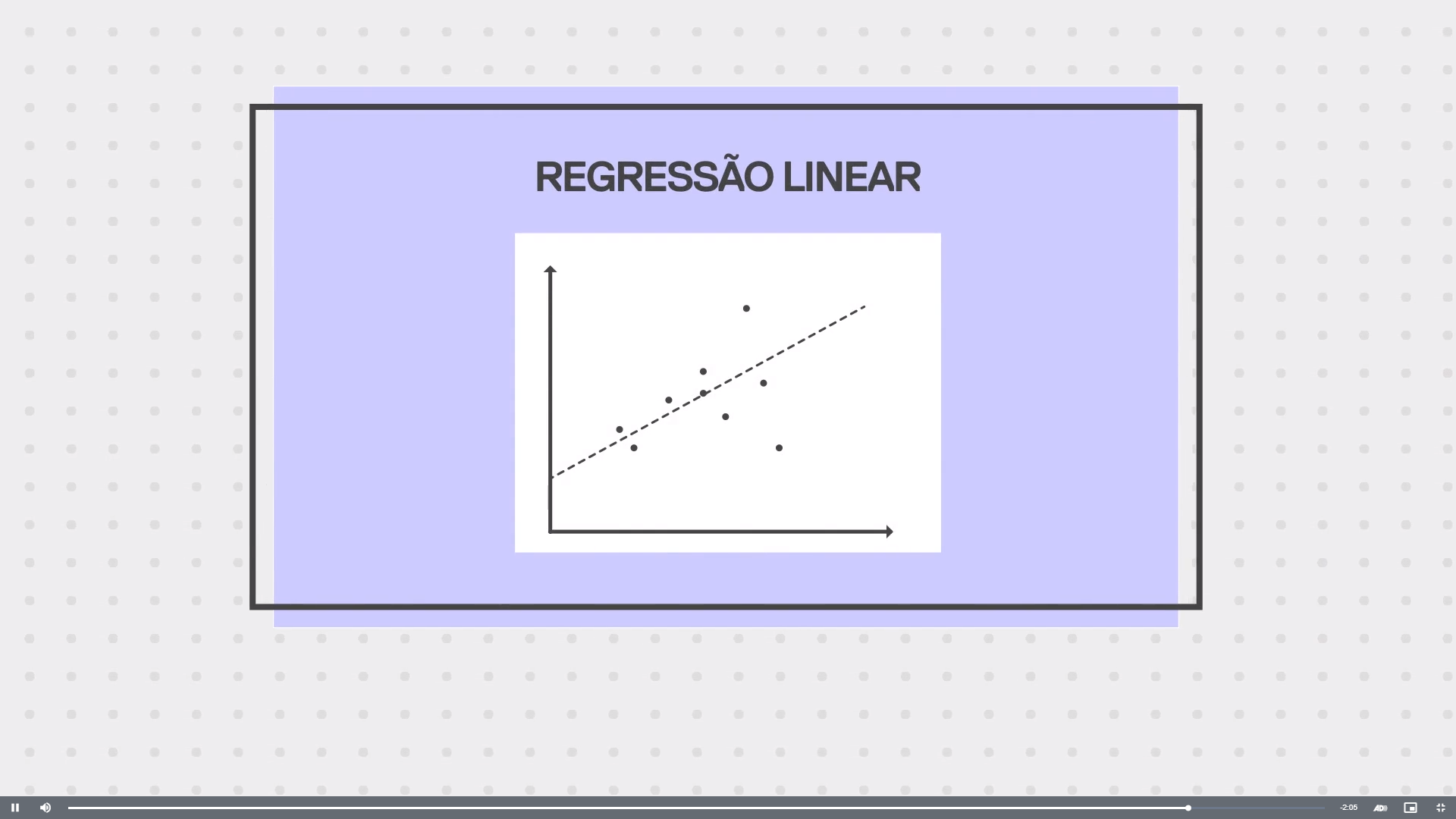
# 

Construindo um modelo Regressão Simples:



**- Linear:**

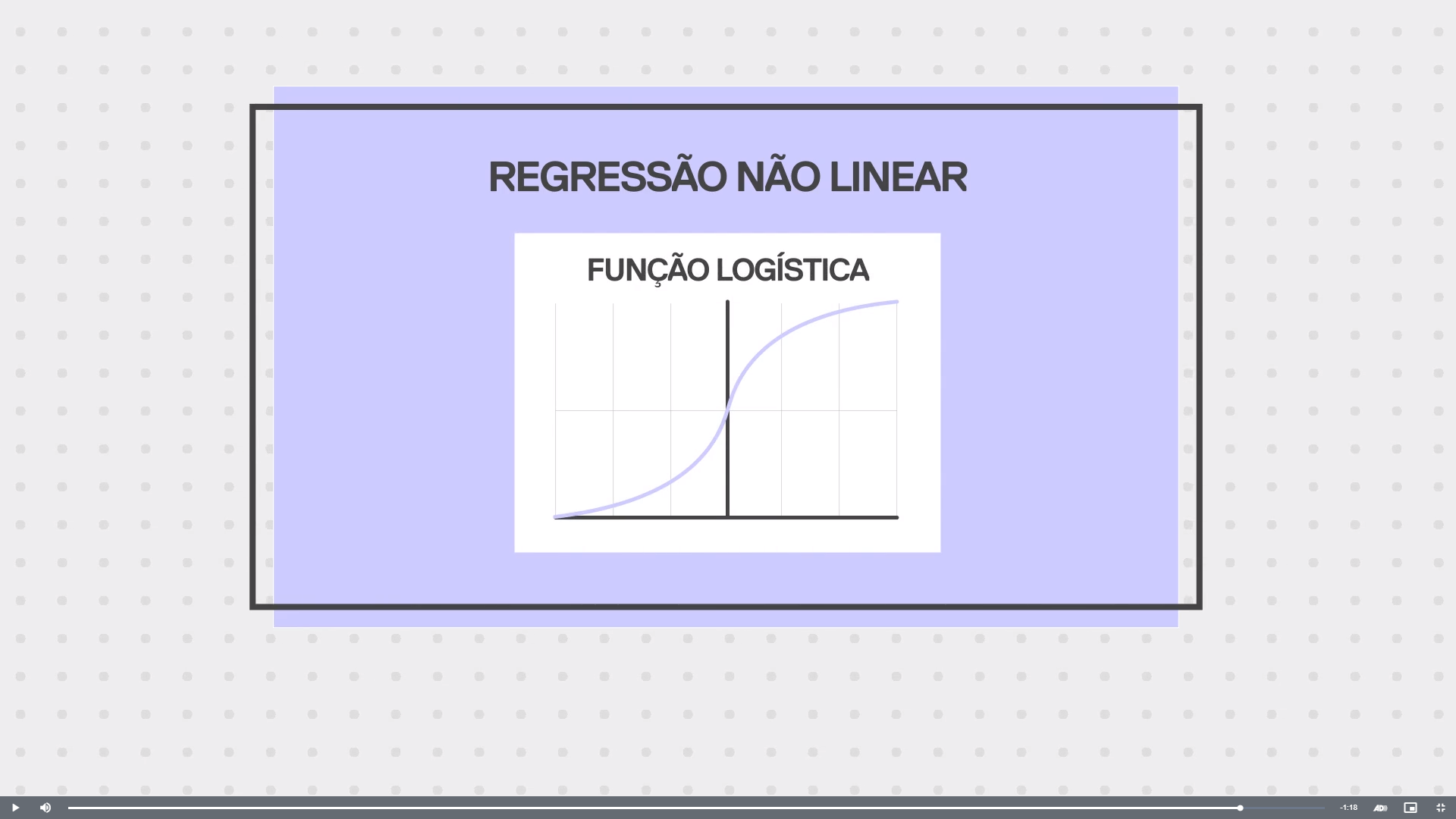
Ocorre quando tem uma demonstração linear em seus parâmetros:



Ex: Y = 10 + 5x.

**- Não Linear:**

Quando a demonstração não é linear



Ex. Y = 1/(1 + EXP(-x))

## Regressão Múltipla

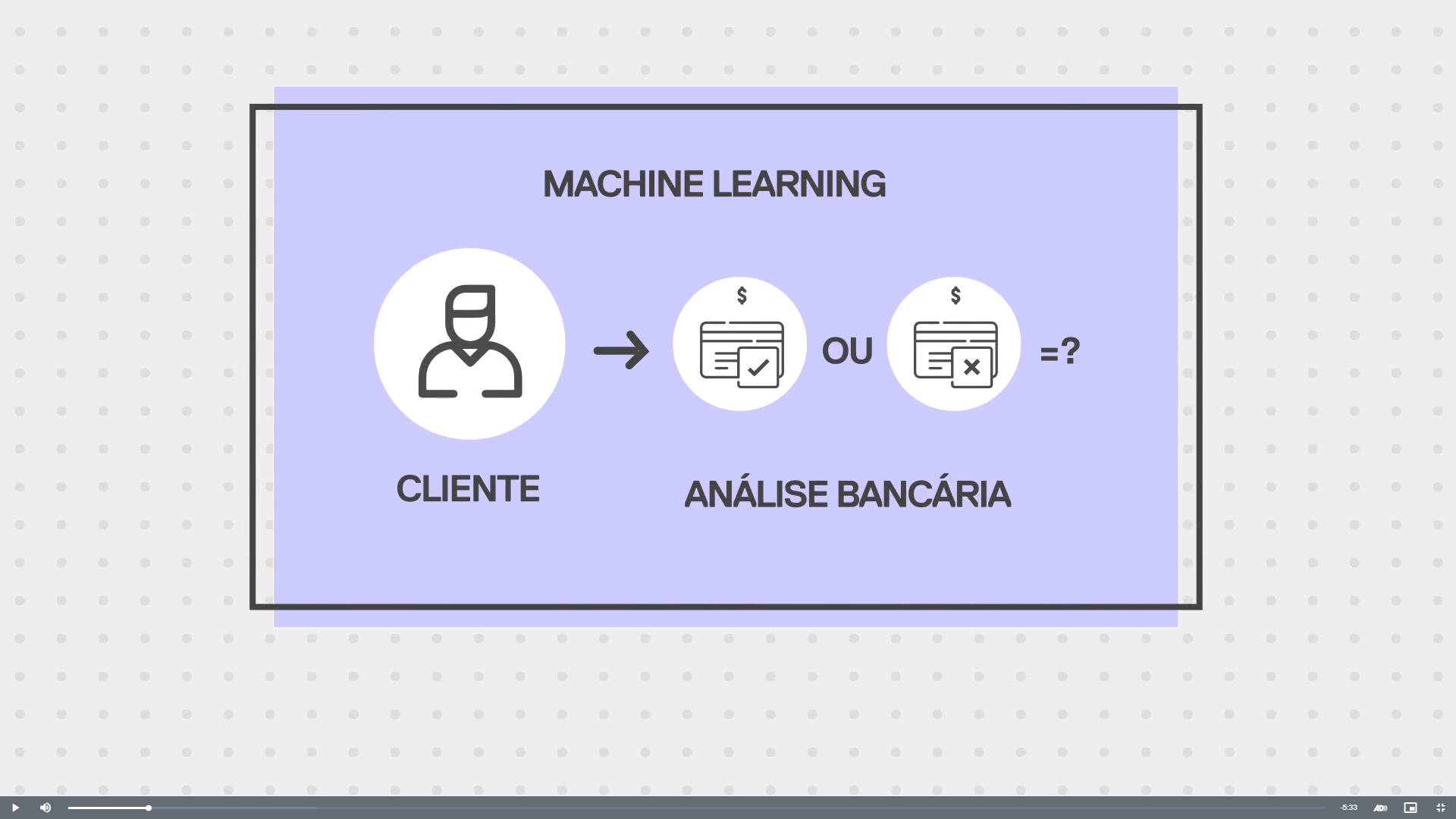
É quando mais de duas ou mais variáveis independentes são usadas para obter uma ou mais variáveis dependentes. Ex: Seguro do carro, baseado na idade e ano do carro.

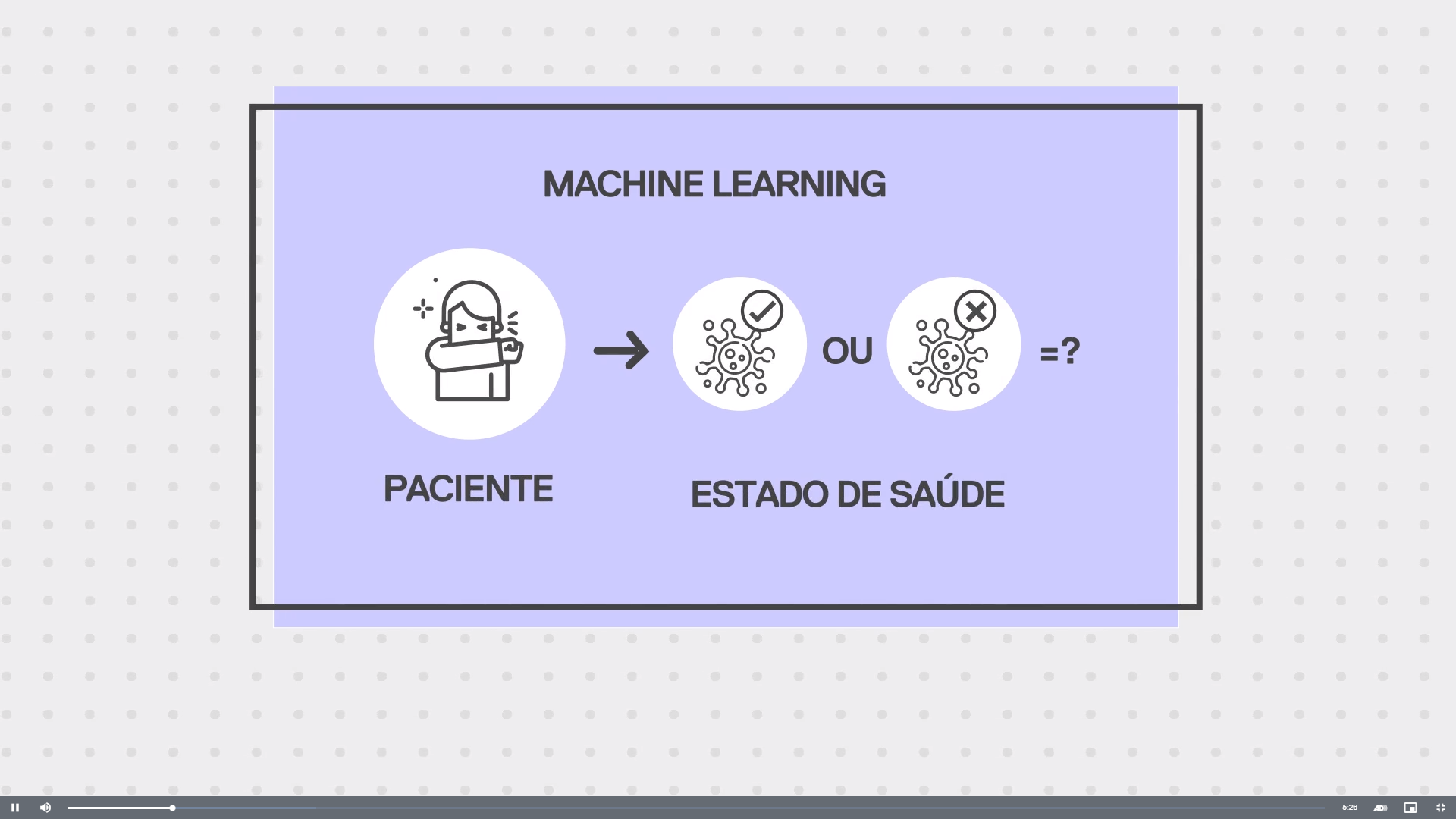
- Linear:

- Não Linear:

# Aprendizado Supervisionado – Classificação

Busca-se associar variáveis de entrada, em categorias distintas. Ex pessoa pode ou não ter crédito aprovado.

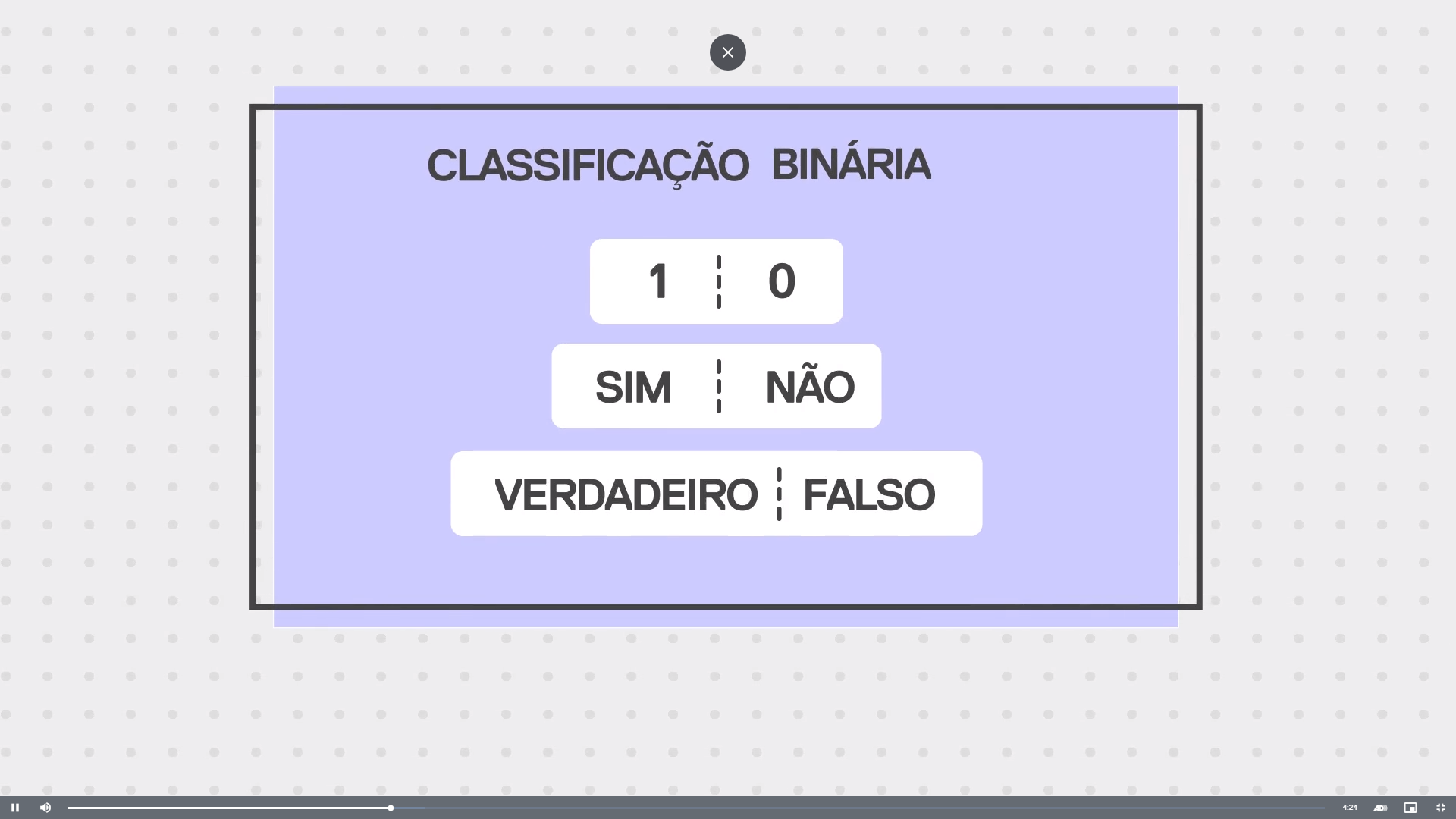




A classificação é uma métrica que tenta classificar ou determinar uma nova informação, com base em um conjunto de informações previamente conhecidos.



## Classificação binária



Ex Modelo se uma time vai ou não ganhar uma partida.

## Classificação Multiclasse

É Quando o Modelo tem que identificar dentre 3 ou mais classificações os seus itens, utilizando os dados.

O modelo de classificação ou Modelo Classificador, tem necessidade de realizar 2 etapas:

1. Treino – Consiste em passar uma parte dos dados em forma de tabela, como forma de entrada. Em seguida o algoritmo fará diversas interações, em busca de padrões de similaridade.
2. Teste de Algoritmo – São atribuídos ao algoritmo os dados que não foram utilizados na etapa anterior, assim é possível identificar a assertividade do modelo.

Modelos de algoritmos de Classificação:

* **K-NEARREST NEIGHBOR (KNN)**
* **DECISION TREES (ÁRVORE DE DECISÃO)**
* **REGRESSÃO LOGÍSTICA**

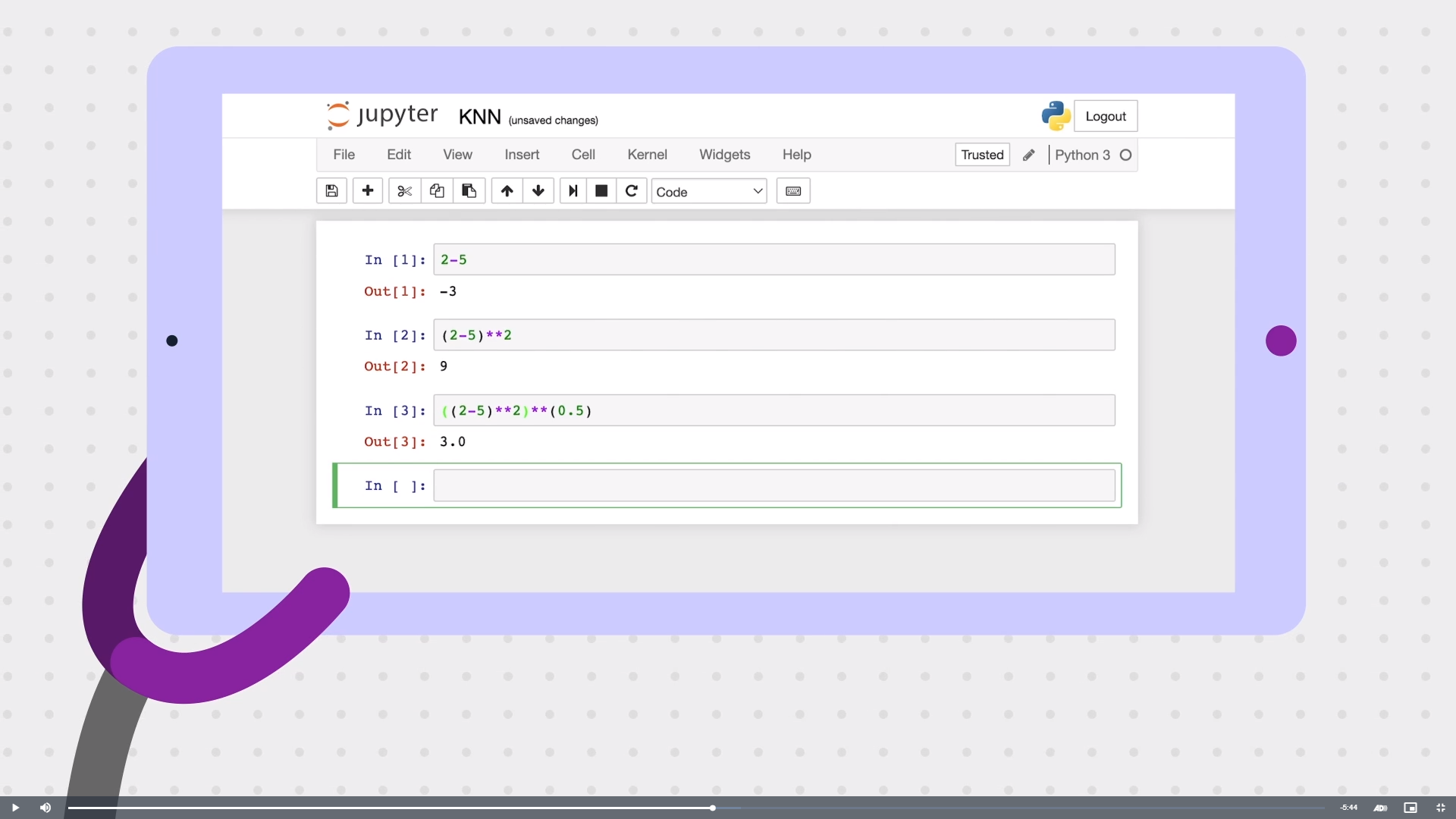
### -K-NEARREST NEIGHBOR (KNN – BUSCAR VIZINHOS MAIS PRÓXIMOS)

É um modelo muito intuitivo, que se vale na ideia simples de classificar elementos, ou executar a regressão de alguns elementos, a partir da distância entre eles.

Ex: Um modelo que precise calcular a semelhança entre elementos, tal semelhança pode ser identifica a partir de uma distância.

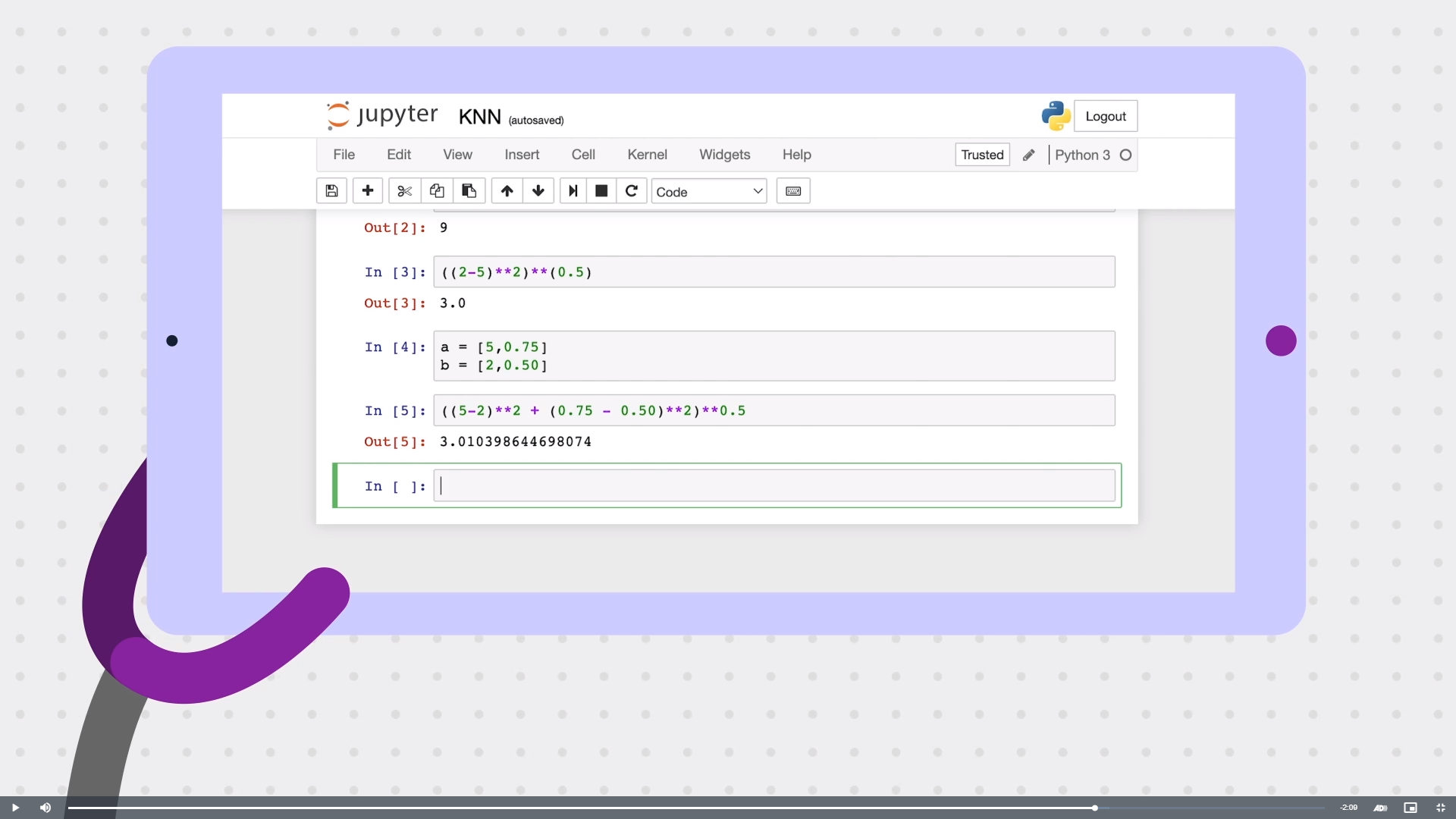
Distância mediana, teoremas de Pitágoras = H² = C1² + C2².

Como distâncias negativas não são interessantes para a análise, precisamos positivar o valor para isso podemos elevar a distância ao quadrado e após isso retirar a raiz quadrada,



Assim eliminamos a possibilidade de encontrarmos valores negativos.

Sendo assim agora precisamos mensurar a distância com mais de 2 atributos.



# Jupyter notebook

## Bibliotecas

Numpy – cria arrays e trabalha com números

Panda – cria tabela

Matplotlib – cria gráficos