



**INSTITUTO
FEDERAL**
Norte de Minas Gerais

Introdução a Sistemas Inteligentes

Modelos Machine Learning
Classificação - KNN

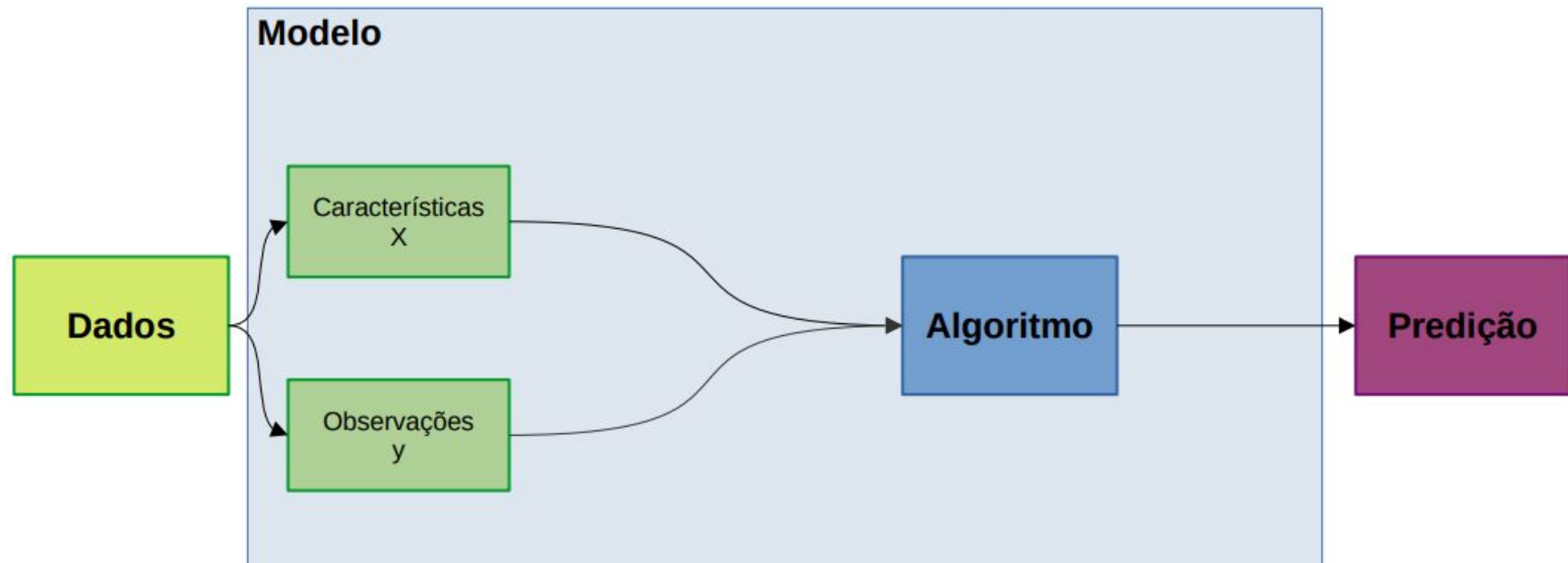
Prof^a. Suzana Mota



Modelos Machine Learning



Machine Learning



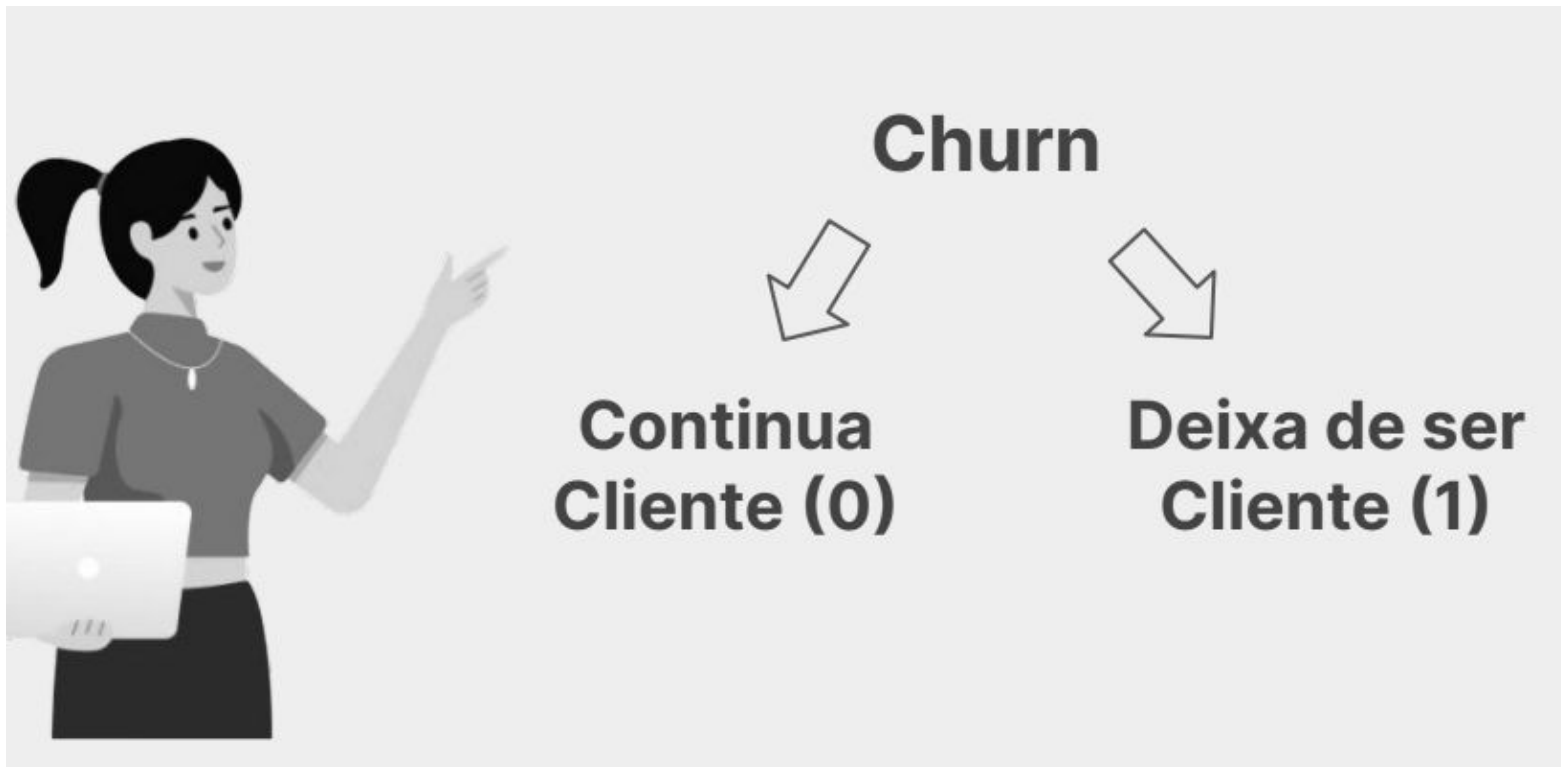
Classificação

Os dados pertencem a classes.

O objetivo é encontrar padrões que permitam **determinar se um dado pertence a uma ou outra classe.**



Classificação





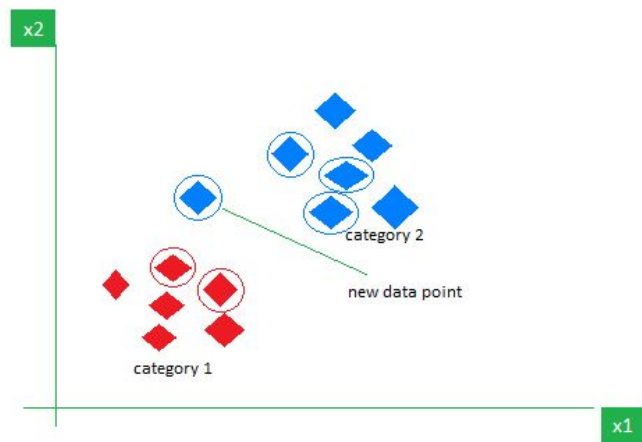
KNN

K-Nearest Neighbors

KNN Nearest Neighbors

KNN é um algoritmo de aprendizado supervisionado usado para classificação e regressão.

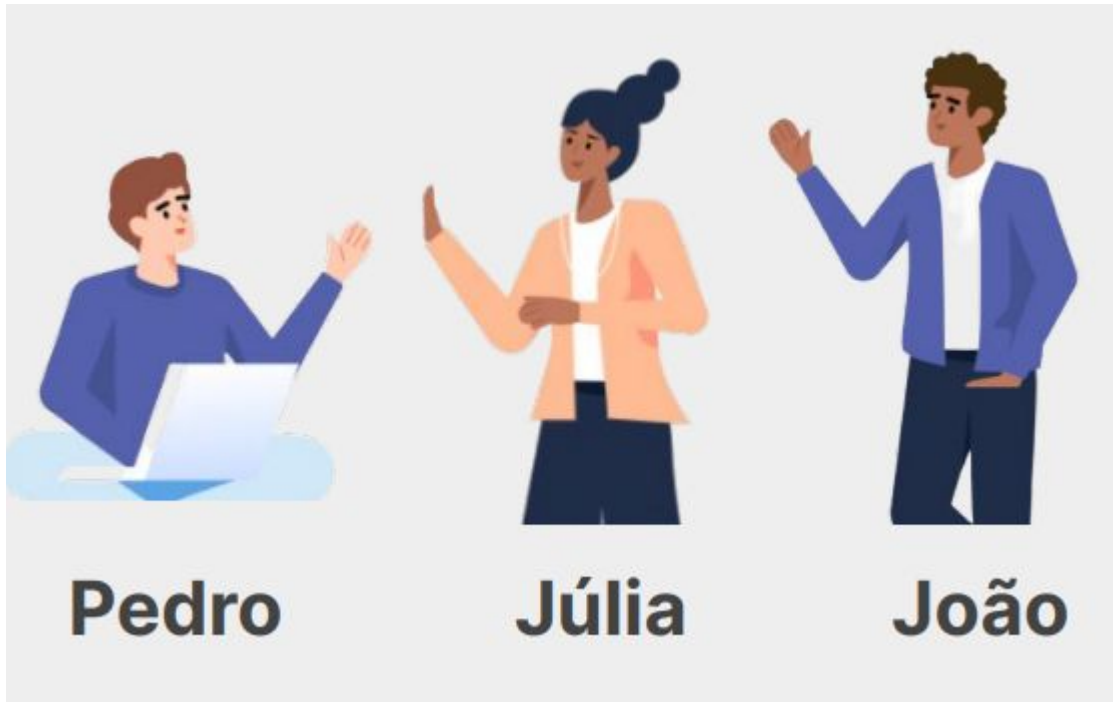
Baseia-se na ideia de que objetos semelhantes estão próximos uns dos outros no espaço de características.



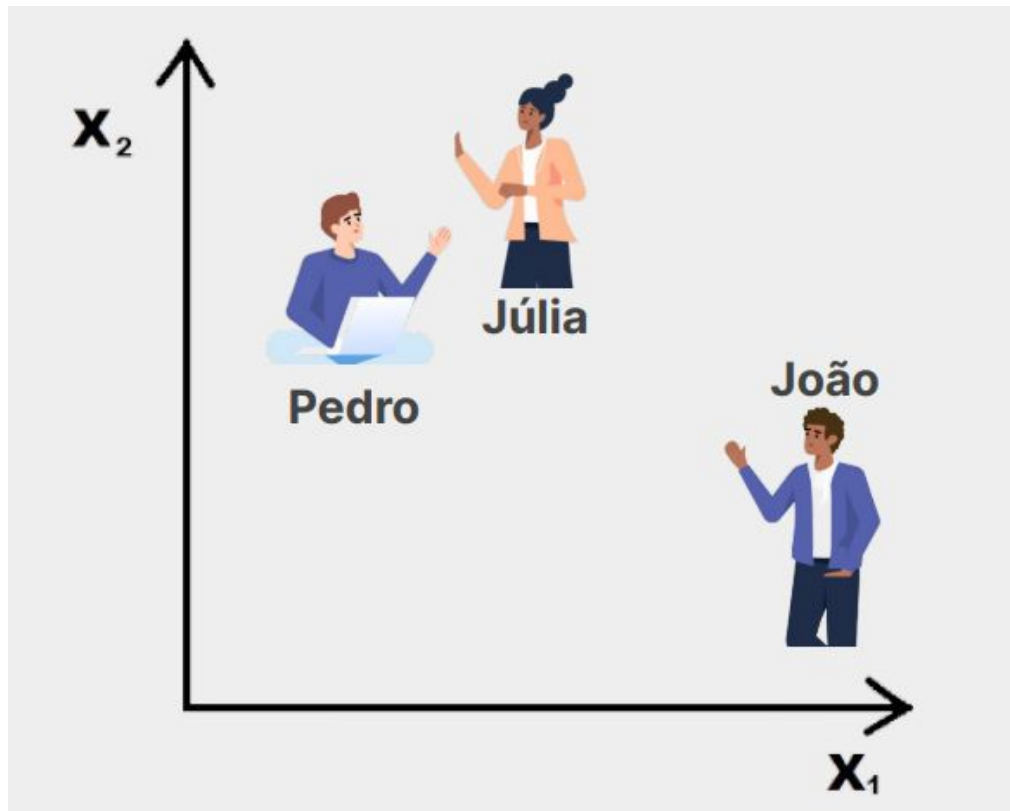
Como o KNN funciona?

1. Armazena o conjunto de dados de treinamento.
2. Para uma nova entrada, calcula a distância entre essa entrada e todos os pontos de dados do conjunto de treinamento.
3. Identifica os K vizinhos mais próximos (pontos mais próximos no espaço).
 - a. Classificação: Atribui a classe mais comum entre os K vizinhos.
 - b. Regressão: Retorna a média dos valores dos K vizinhos.

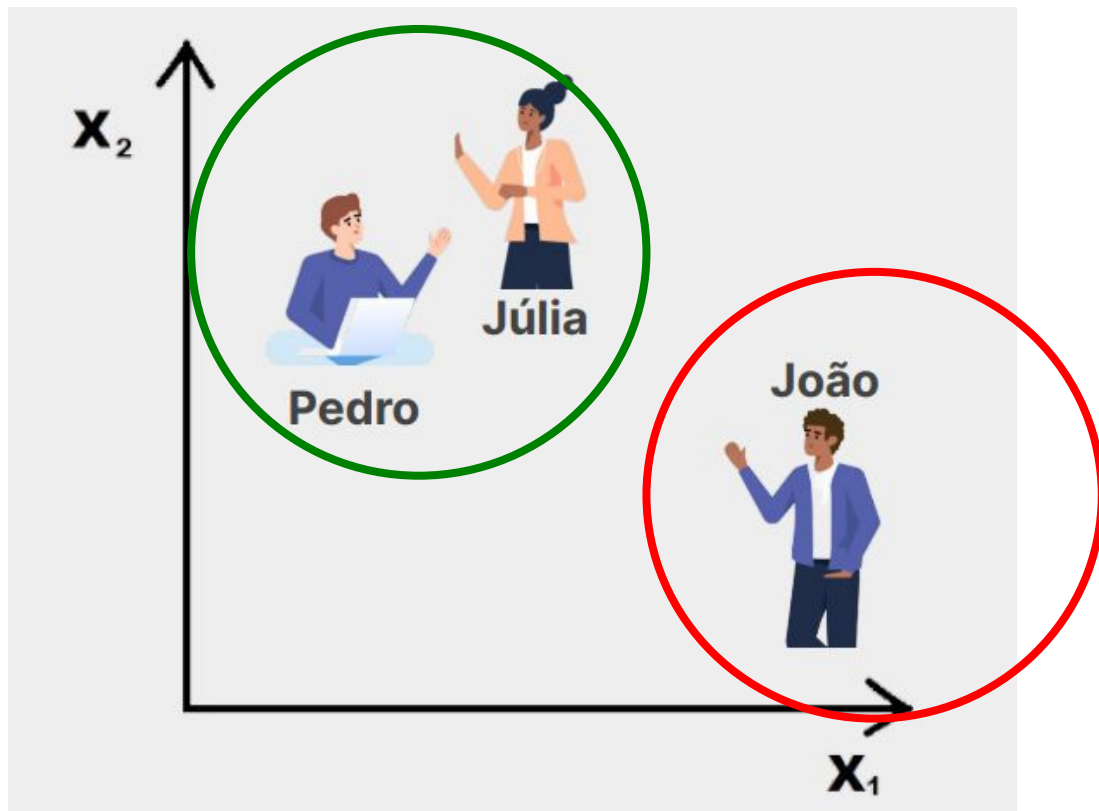
Classificação



Classificação

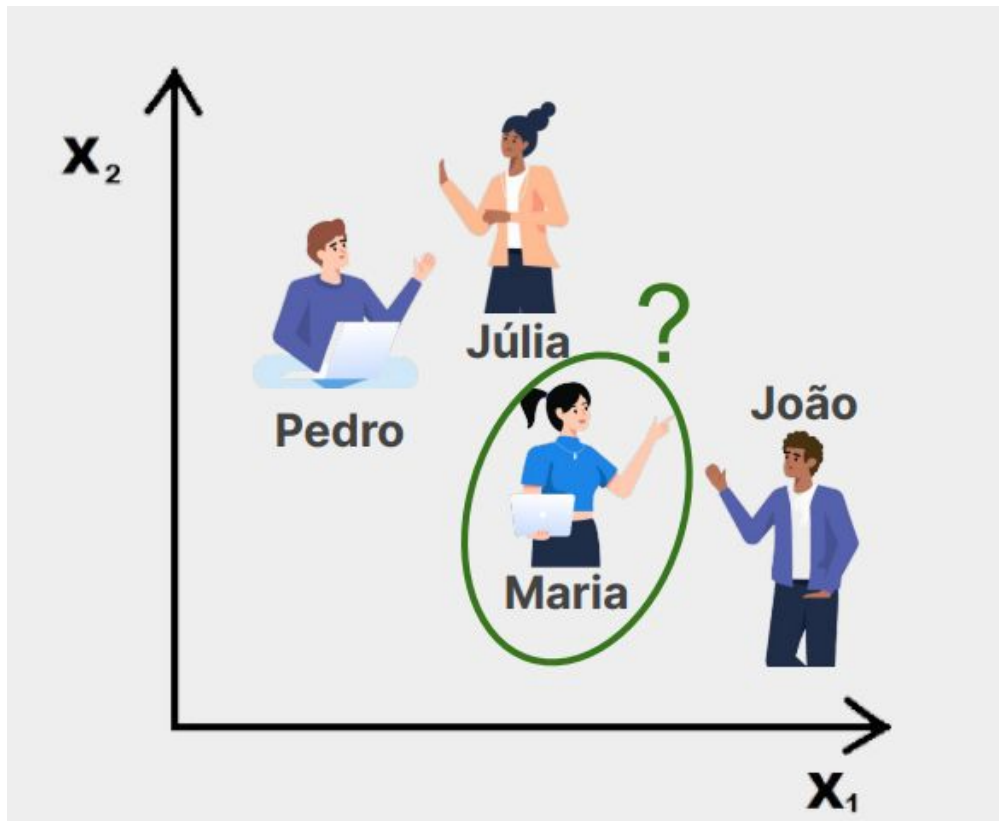


Classificação

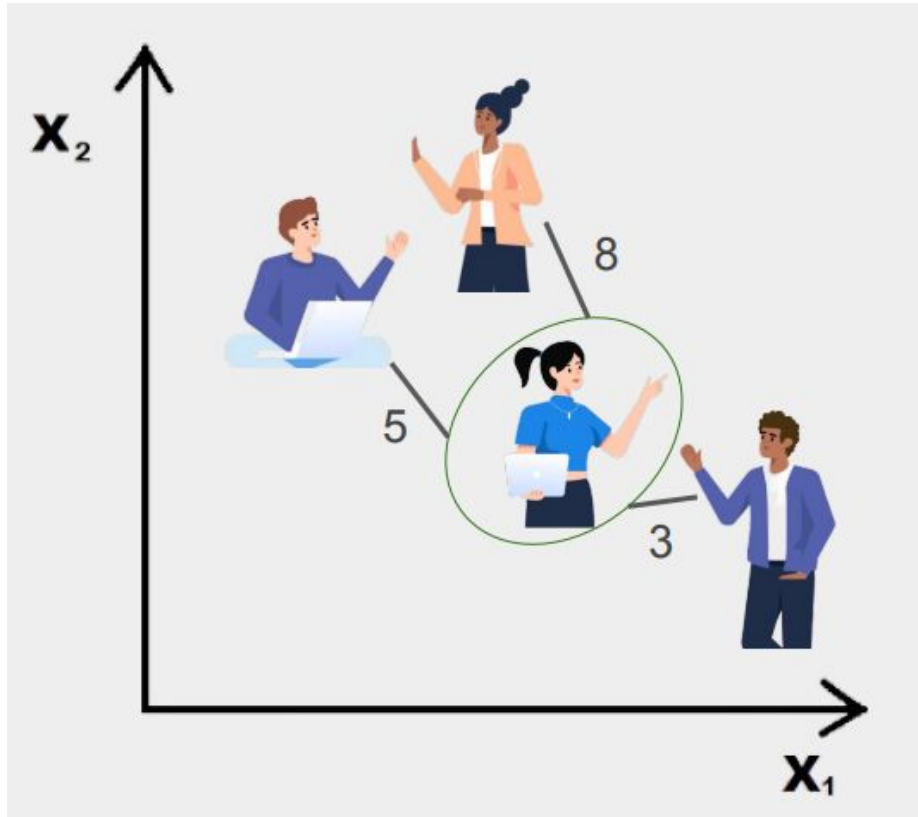


CHURN
DEIXAR DE ASSINAR ALGUM
PRODUTO OU SERVIÇO

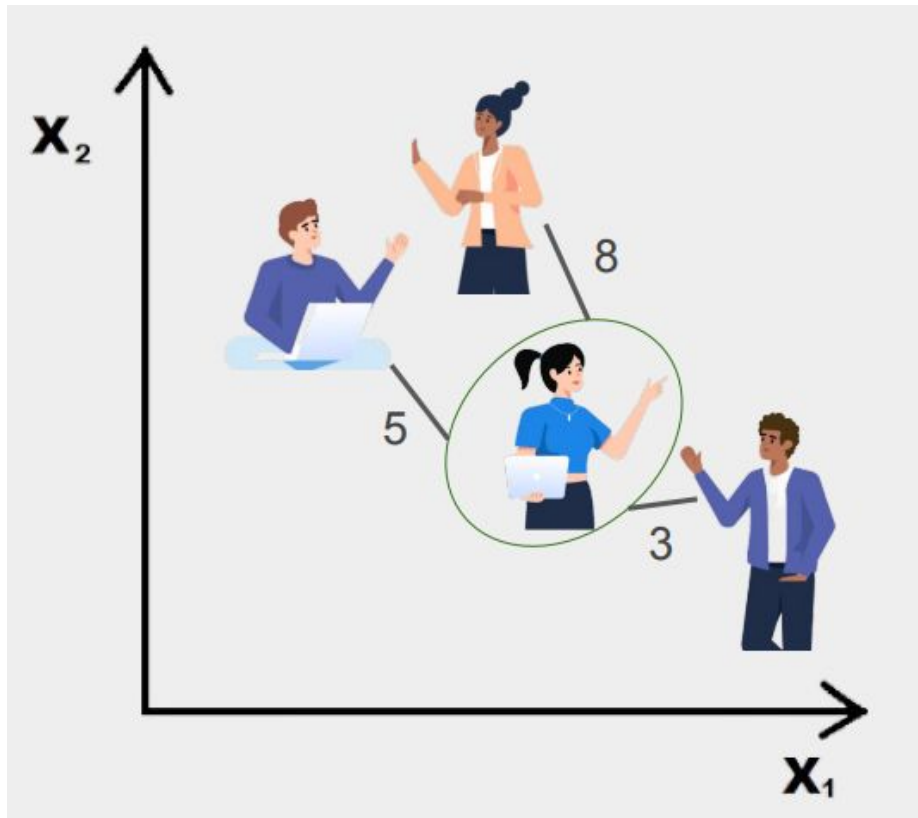
Classificação



Classificação



Classificação



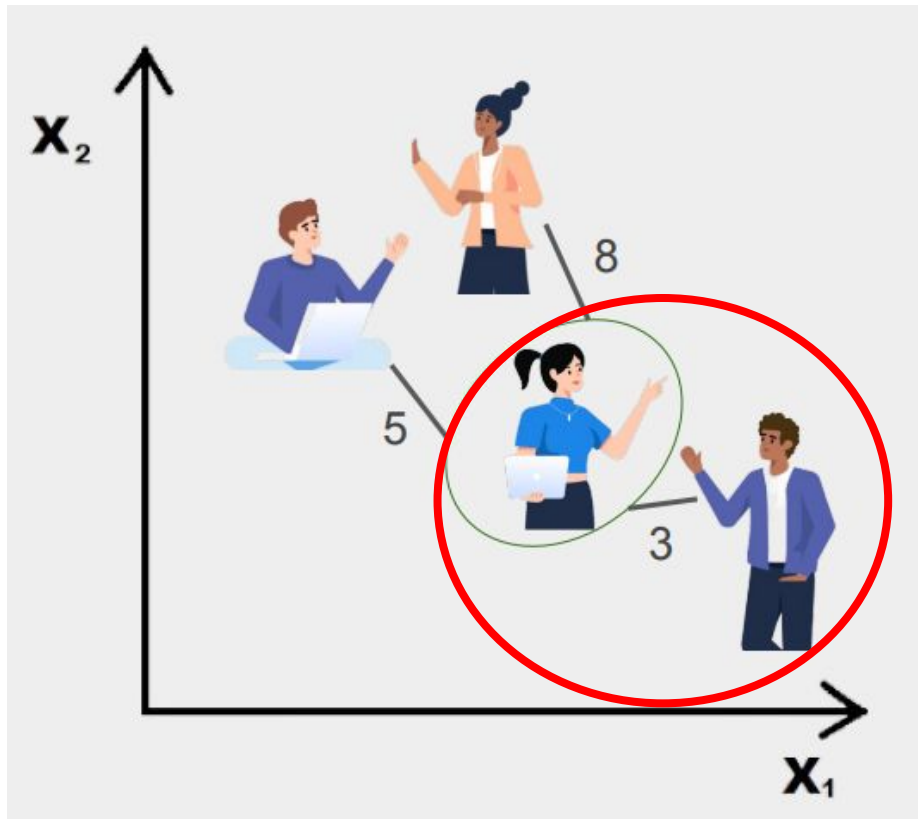
$K = 1$ (Distância de 1 vizinho mais próximo)

João 3

Pedro 5

Júlia 8

Classificação



$K = 1$ (Distância de 1 vizinho mais próximo)

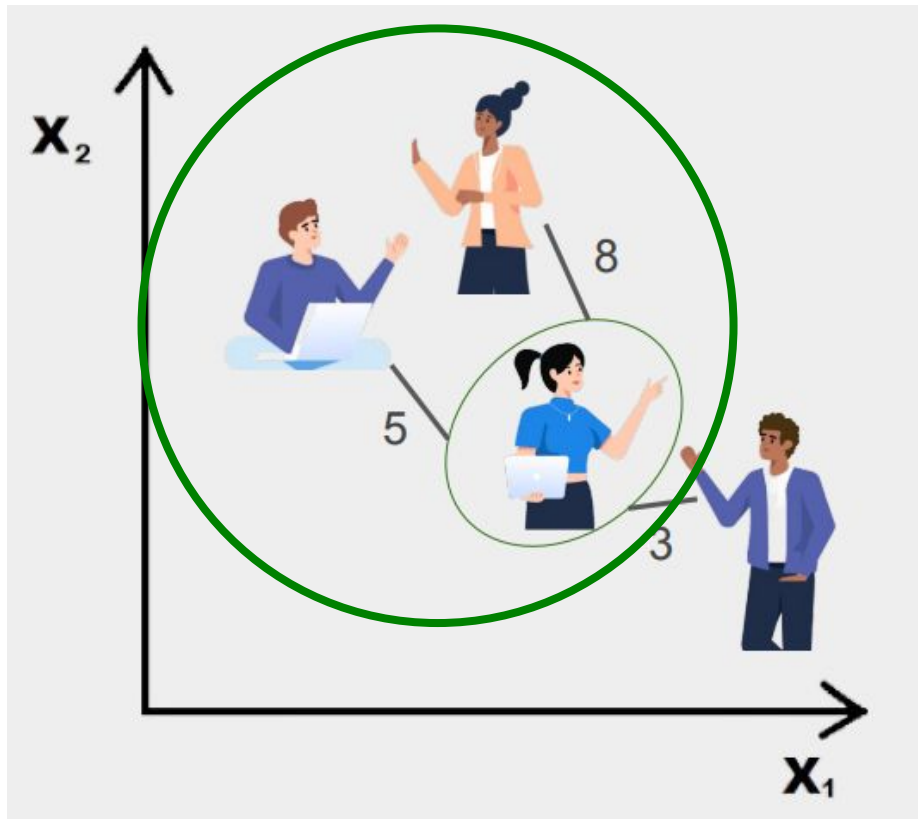
João 3

Pedro 5

Júlia 8

Provavelmente Maria dará Churn!

Classificação



$K = 3$ (Distância de 3 vizinho mais próximos)

João 3

Pedro 5

Júlia 8

KNN Nearest Neighbors

K é o número de vizinhos considerados para a classificação ou regressão.

K pequeno: Mais sensível a ruídos, pode resultar em overfitting.

K grande: Gera uma decisão mais "suavizada", mas pode ser impreciso se for muito grande (Underfitting).

Distâncias Usadas


- Distância Euclidiana

$$d(x, y) = \sqrt{\sum_{k=1}^n (x_k - y_k)^2}$$

- Distância de Manhattan

- Distância de Minkows

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

A large green circle is partially visible in the top right corner of the slide.

Conceitos Importantes

Overfitting (sobreajuste)

É um problema comum em modelos de machine learning, que ocorre quando **o modelo se ajusta muito bem aos dados de treinamento, mas tem desempenho ruim em novos dados.**



Causas de Overfitting



Modelo excessivamente complexo:

- Modelos muito complexos, como redes neurais profundas ou árvores de decisão muito profundas, tendem a capturar padrões irrelevantes.

Poucos dados de treinamento:

- Quando o conjunto de dados de treinamento é muito pequeno, o modelo pode aprender os detalhes específicos desse conjunto, mas não consegue generalizar para novos dados.

Ruído nos dados:

- O modelo pode aprender padrões com base em ruídos ou outliers dos dados, o que não é desejável.

Treinando um Modelo





<https://scikit-learn.org/stable/>

<https://scikit-learn.org/dev/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

Divisão de Dados

Os dados precisam ser divididos em treino e teste!

Para que não ocorra o overfitting, ou seja, para que eu tenha certeza que o Modelo treinado foi capaz de identificar padrões em dados que ele não conhece!



70%

80%

30%

20%

Dados de Treino

Conjunto de dados utilizado para treinar o modelo.

O modelo aprende os padrões e as relações presentes nesses dados.

Dados de Teste

Conjunto de dados separados para avaliar o desempenho do modelo.

Os dados de teste simulam novos dados reais, ajudando a verificar se o modelo consegue generalizar para dados que ele ainda não viu.

Por que separar os dados?

Prevenção de Overfitting:

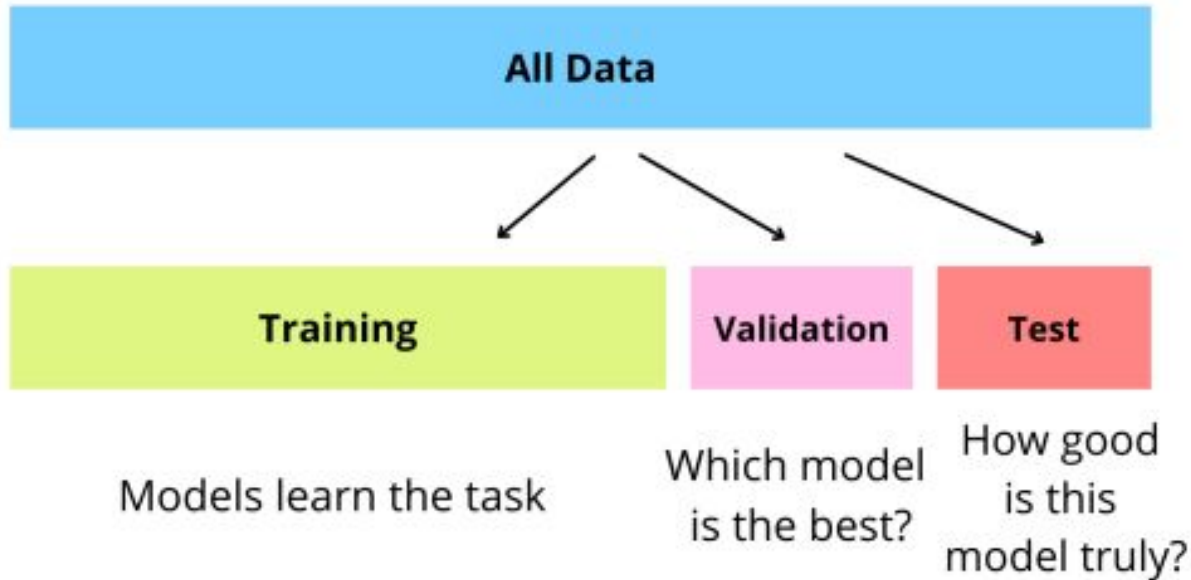
- Ao usar dados de treino para ajustar o modelo e dados de teste para avaliar, evitamos que o modelo se ajuste demais aos dados de treino (overfitting).

Avaliação Realista:

- Dados de teste fornecem uma medida mais realista de como o modelo funcionará no mundo real com dados não vistos.



Por que separar os dados?



Separando os dados

```
from sklearn.model_selection import train_test_split
```

```
X_treino, X_teste, y_treino, y_teste = train_test_split(X_normalizado, y, test_size=0.3, random_state=123)
```

30% TESTE

SEMENTE
RANDÔMICA

X_TREINO

X_TESTE

Features

Y_TREINO

Y_TESTE

Labels

“Respostas Certas”

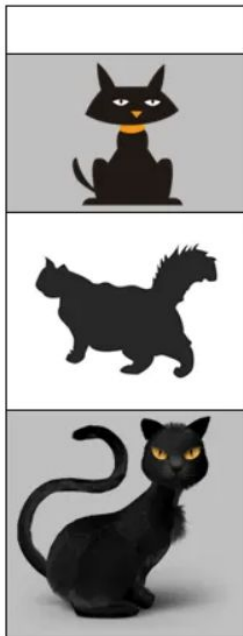
X OU Y?

É fofinho?	Tem orelhinha pequena?	Faz miau?
1	1	1
1	0	1
0	1	1

X OU Y?

É fofinho?	Tem orelhinha pequena?	Faz miau?
1	1	1
1	0	1
0	1	1

X
features



Y
labels

Treinando o Modelo

```
from sklearn.neighbors import KNeighborsClassifier
```

```
#instanciar o modelo (criamos o modelo)  
#por padrão são 5 vizinhos  
knn = KNeighborsClassifier(metric='euclidean')
```

```
#treinando o modelo com os dados de treino  
knn.fit(X_treino, y_treino)
```

Testando o Modelo

```
from sklearn.metrics import accuracy_score
```

```
#testando o modelo com os dados de teste  
predito_knn = knn.predict(X_teste)
```

```
acuracia = accuracy_score(y_teste, predito_knn)  
print(f'Acurácia do modelo KNN: {acuracia:.2f}')
```

Acurácia do modelo KNN: 0.81

Mudando parâmetros

```
from sklearn.neighbors import KNeighborsClassifier
```

```
k = 2  
knn = KNeighborsClassifier(n_neighbors=k)
```

```
#treinando o modelo com os dados de treino  
knn.fit(X_treino, y_treino)
```

***Machine Learning**

***Algorithms**

***Maths**

What the hell is this?

