



Aula: Introdução ao Aprendizado de Máquina (Machine Learning)



Machine Learning (Aprendizado de Máquina) é uma área da Inteligência Artificial que permite que sistemas aprendam padrões a partir de dados e tomem decisões sem serem explicitamente programados. Existem três principais paradigmas:

1. Aprendizado Supervisionado (Supervised Learning)

No **aprendizado supervisionado**, o modelo é treinado com **dados rotulados** — ou seja, cada entrada possui uma saída desejada conhecida.

- **Objetivo:** Aprender uma função que mapeia entradas para saídas.
- **Dados:** Conjunto de exemplos com input e output (X, Y).
- **Tarefas comuns:**
- **Classificação:** Prever categorias (ex: detectar spam em e-mails).
- **Regressão:** Prever valores contínuos (ex: previsão de preço de imóveis).
- **Exemplo de algoritmo:**
- Regressão Linear, Regressão Logística
- Árvores de Decisão
- Random Forest
- Redes Neurais

Vantagens: Alta precisão se houver muitos dados rotulados.

Desvantagens: Exige dados rotulados, que podem ser caros ou demorados de obter.

2. Aprendizado Não Supervisionado (Unsupervised Learning)

No **aprendizado não supervisionado**, os dados **não possuem rótulos**. O objetivo é descobrir padrões ou estruturas escondidas nos dados.

- **Objetivo:** Encontrar estrutura ou agrupamentos nos dados.
- **Dados:** Conjunto de entradas (X) sem saídas conhecidas.
- **Tarefas comuns:**
 - **Clusterização (agrupamento):** Agrupar dados semelhantes (ex: segmentação de clientes).
 - **Redução de Dimensionalidade:** Reduzir número de variáveis mantendo informação relevante (ex: PCA).
- **Exemplo de algoritmo:**
 - K-Means
 - DBSCAN
 - PCA (Principal Component Analysis)
 - Autoencoders

Vantagens: Não precisa de dados rotulados.

Desvantagens: Difícil avaliar a qualidade dos resultados.

3. Aprendizado por Reforço (Reinforcement Learning)

No **aprendizado por reforço**, um agente aprende **interagindo com o ambiente** e recebendo **recompensas ou punições**.

- **Objetivo:** Aprender uma política de ações que maximize a recompensa acumulada.

- **Dados:** Feedback em forma de recompensa ou penalidade após ações.
- **Exemplos de aplicação:**
 - Jogos (ex: AlphaGo, xadrez)
 - Robótica (ex: ensinar robôs a andar)
 - Sistemas de recomendação
- **Componentes principais:**
 - **Agente:** Quem toma decisões
 - **Ambiente:** Onde o agente atua
 - **Ação:** Decisão tomada pelo agente
 - **Recompensa:** Feedback recebido após a ação
- **Exemplo de algoritmo:**
 - Q-Learning
 - Deep Q-Network (DQN)
 - Policy Gradient

Vantagens: Aprende sozinho a partir de tentativa e erro.

Desvantagens: Pode exigir muito tempo de treinamento e recursos computacionais.

Resumo rápido:

Tipo	Dados	Objetivo	Exemplos de uso
Supervisionado	Com rótulos	Prever saída de entrada	Classificação (spam, diagnóstico), Regressão (preço de imóveis, previsão de vendas)
Não Supervisionado	Sem rótulos	Descobrir padrões	Clusterização (segmentação de clientes), Redução de

Tipo	Dados	Objetivo	Exemplos de uso
			Dimensionalidade (PCA), Detecção de anomalias
Reforço	Feedback do ambiente	Maximizar recompensa	Jogos (AlphaGo), Robótica, Sistemas de recomendação, Veículos autônomos

◆ **O que é Machine Learning?**

Machine Learning (ML) é um campo da Inteligência Artificial que permite que sistemas aprendam padrões a partir de dados, **sem serem explicitamente programados**.

Exemplo:

📧 Um filtro de spam aprende, a partir de e-mails anteriores, a reconhecer quais são spam ou não.

◆ **Conceitos-chave**

| Conceito | Descrição | Exemplo |

|-----|-----|-----|

| **Dados** | Conjunto de informações coletadas para análise | Tabela com e-mails e suas classificações |

| **Features (atributos)** | Características ou variáveis usadas para descrever os dados | Número de palavras, presença de links |

| **Labels (rótulos)** | O resultado esperado que queremos prever | Spam (1) / Não Spam (0) |

| **Modelo** | O algoritmo que aprende o padrão entre features e labels | O classificador de spam treinado |

🧠 **Resumo:**

Machine Learning = Dados + Features + Labels → Modelo que aprende padrões

2. Tipos de Aprendizado

2.1 Aprendizado Supervisionado

- O modelo **aprende com exemplos rotulados** (dados + respostas corretas).
- Usado para **classificação** e **regressão**.

Exemplo:

Treinar um modelo com imagens de gatos 🐱 e cachorros 🐶 para que ele aprenda a diferenciar.

Algoritmos comuns:

- KNN
 - SVM
 - Regressão Linear / Logística
 - Decision Trees
-

2.2 Aprendizado Não Supervisionado

- O modelo **não recebe rótulos** — ele precisa **encontrar padrões sozinho**.
- Usado para **agrupamento (clustering)** e **redução de dimensionalidade**.

Exemplo:

Agrupar clientes por comportamento de compra sem saber quem é quem.

Algoritmos comuns:

- K-means
- PCA (Análise de Componentes Principais)

2.3 Aprendizado por Reforço

- Um agente **aprende com tentativa e erro**, recebendo **recompensas** ou **punições**.
- Usado em jogos, robótica e controle autônomo.

Exemplo:

Um robô aprende a andar ajustando seus passos até ganhar a maior recompensa possível (ficar em pé por mais tempo).

3. Principais Algoritmos (25 – 45 min)

♦ 3.1 KNN (K-Nearest Neighbors)

- **Tipo:** supervisionado
- **Ideia:** um novo ponto é classificado com base nos **K vizinhos mais próximos**.
- **Vantagem:** simples e intuitivo
- **Desvantagem:** lento para grandes volumes de dados

Exemplo:

Para prever se uma fruta é maçã 🍏 ou laranja 🍊, olhe para as frutas mais próximas em tamanho e cor.

♦ 3.2 SVM (Support Vector Machine)

- **Tipo:** supervisionado
- **Ideia:** encontrar a **melhor fronteira (hiperplano)** que separa as classes.
- **Vantagem:** eficaz em dados de alta dimensão

- **Desvantagem:** sensível a outliers e parâmetros

Exemplo:

Separar pontos vermelhos e azuis desenhando a linha que deixa a maior margem entre eles.

♦ 3.3 K-means

- **Tipo:** não supervisionado
- **Ideia:** agrupar dados em **K grupos**, minimizando a distância dos pontos ao centro (centroide).
- **Vantagem:** rápido e fácil de interpretar
- **Desvantagem:** requer definir **K** previamente

Exemplo:

Segmentar clientes em grupos de comportamento (baixo, médio e alto consumo).

Demonstração curta (opcional – 5 min)

Usando Python (pode ser mostrado em slides ou Jupyter Notebook):

```
from sklearn.datasets import load_iris

from sklearn.cluster import KMeans

import pandas as pd

data = load_iris()

df = pd.DataFrame(data.data, columns=data.feature_names)
```



```
kmeans = KMeans(n_clusters=3)

df['cluster'] = kmeans.fit_predict(df)

print(df.head())
```

4. Pipeline de Machine Learning (45 – 60 min)

Um projeto de ML segue **etapas estruturadas**, conhecidas como **pipeline**.

4.1 🕒 Coleta de Dados

- Obter informações de fontes como bancos de dados, APIs ou sensores.
- Garantir a **qualidade**, **relevância** e **consistência** dos dados coletados.
- Nesta etapa, também é importante definir o **escopo do problema** e quais dados realmente importam.

📌 *Exemplo:* Coletar dados de vendas de uma loja online para prever o volume de pedidos.

4.2 🛠️ Pré-processamento

Etapa essencial para preparar os dados antes do treinamento do modelo.

Passos comuns:

- Remover valores ausentes (NaN) ou duplicados
- Normalizar e padronizar variáveis numéricas
- Converter variáveis categóricas em numéricas (ex: one-hot encoding)

- Selecionar apenas as **features relevantes**

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X_scaled = scaler.fit_transform(df[data.feature_names])
```

4.3 🖌️ Treinamento e Validação

Nesta etapa, o modelo é treinado para aprender os padrões presentes nos dados.

Etapas principais:

1. Dividir os dados em **conjuntos de treino e teste**.
2. Treinar o modelo usando os dados de treino.
3. Avaliar o desempenho usando os dados de teste.
4. Utilizar métricas de qualidade (acurácia, precisão, recall, F1-score, etc.).

Exemplo em Python:

```
from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

from sklearn.neighbors import KNeighborsClassifier
```

Divisão entre treino (80%) e teste (20%)

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, data.target,
```


Treinamento do modelo KNN

```
model = KNeighborsClassifier(n_neighbors=3)

model.fit(X_train, y_train)
```

Predição e avaliação

```
y_pred = model.predict(X_test)

print("Acurácia:", accuracy_score(y_test, y_pred))
```

4.4 Ajuste e Iteração

Depois do primeiro treinamento, é comum **ajustar e iterar** para melhorar o desempenho do modelo.

Passos principais:

- Alterar **hiperparâmetros** (por exemplo, k no KNN ou c no SVM).
- Testar diferentes **algoritmos**.
- Avaliar com **validação cruzada (cross-validation)** para medir desempenho médio.
- Repetir o processo até atingir **equilíbrio entre precisão e generalização**.

Exemplo em Python:

```
from sklearn.model_selection import cross_val_score
```

Avaliação usando validação cruzada

```
scores = cross_val_score(model, X_scaled, data.target, cv=5)

print("Acurácia média:", scores.mean())
```

Conclusão

- Machine Learning permite **aprender padrões a partir de dados**.
- Existem **três tipos principais de aprendizado**:
 - Supervisionado
 - Não supervisionado
 - Por reforço
- Algoritmos introdutórios importantes: **KNN, SVM e K-means**.
- Todo projeto de ML segue um **pipeline estruturado**:

Resumo:

O sucesso em ML depende tanto da qualidade dos dados quanto do processo de experimentação.

Atividade de Fixação

1. Dê um exemplo prático de cada tipo de aprendizado.
2. Classifique o problema:

Prever o preço de uma casa com base em tamanho e localização.

É supervisionado, não supervisionado ou por reforço?

3. Qual é a diferença entre **feature** e **label**?
4. O que é **overfitting** e como evitá-lo?

5. Descreva as principais etapas do **pipeline de ML** com suas palavras.

