



PROJETO </> EDUCAÇÃO

DO FUTURO

Data Science 1

AULA 01



AULA 01

CONTEÚDOS

- Aprendizagem de Máquina
- Porque aprender Machine Learning
- Terminologia
- Métodos Preditivos
- Tipos de aprendizagem de máquina
- Introdução a biblioteca Pandas



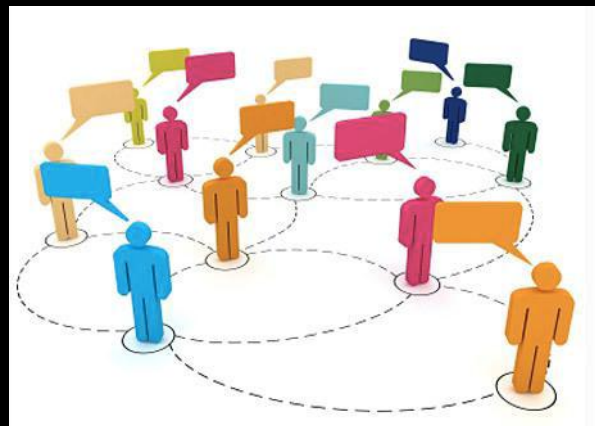
Utilização



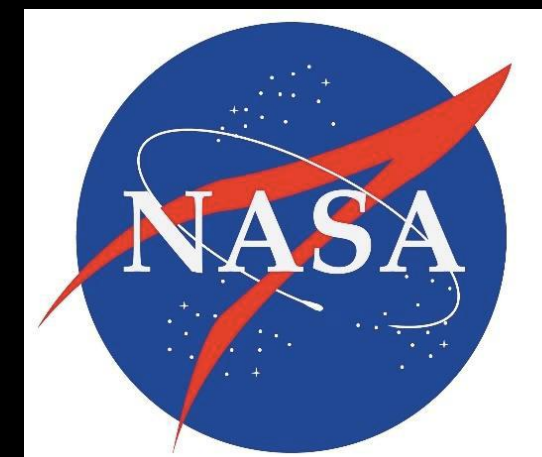
facebook Ads

Google

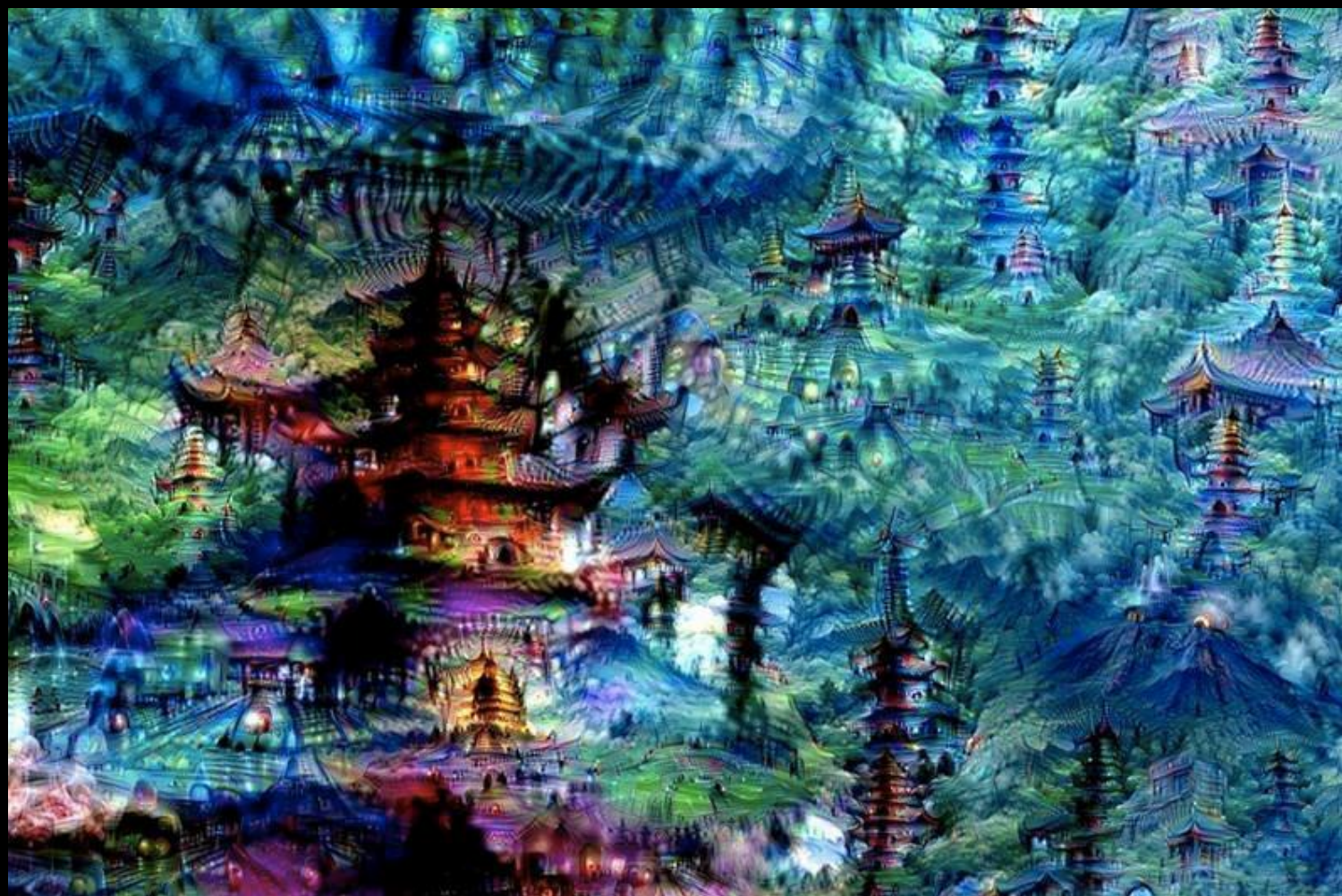
NETFLIX



amazon.com[®]
eHarmony[®]



Geração de imagens



Geração de imagens



<https://thispersondoesnotexist.com/>



Geração de texto



ChatGPT

Koala



grammarly





1. Introdução à Ciência de Dados

Ciência de Dados é uma área que une matemática, estatística, programação e conhecimento do domínio para coletar, organizar, analisar e extrair valor de dados.

💡 É como pegar um monte de dados soltos e transformá-los em respostas úteis para decisões reais.





2. O processo de ciência de dados

1. Coleta de dados
2. Limpeza e organização
3. Exploração e visualização
4. Modelagem (Machine Learning)
5. Validação dos resultados
6. Apresentação e tomada de decisão



3. Conexão com áreas do dia a dia

Área	Aplicação com dados
Marketing	Segmentação de clientes, campanhas personalizadas
Saúde	Diagnósticos preditivos, gestão hospitalar
Finanças	Deteção de fraudes, precificação de seguros
Logística	Roteirização e otimização de entregas
Governo	Previsão de epidemias, distribuição de recursos públicos
RH	Análise de produtividade, retenção de talentos




4. Perfil do cientista de dados

As 3 principais habilidades que se cruzam:

 Estatística e matemática: saber interpretar dados.

 Programação (Python): saber transformar dados em análises e modelos.

 Conhecimento do domínio: saber o que perguntar e como interpretar.

Ninguém nasce sabendo tudo isso. É uma jornada. E hoje, a gente começa com a ferramenta central: o Pandas, para ler e explorar dados de forma estruturada.



 Ligando a Teoria à Prática:

Python e as Bibliotecas de Ciência de Dados

Agora que entendemos o que é Ciência de Dados, vamos ver como ela é feita na prática. E o principal ambiente que usamos é o Python.





Por que usamos Python?

É uma linguagem simples e legível – fácil de aprender.

Tem uma enorme comunidade ativa com bibliotecas prontas.

É extremamente versátil: serve tanto para automação quanto para análise avançada e inteligência artificial.

O segredo do Python em Ciência de Dados não está só na linguagem. Está nas bibliotecas que ele oferece.





Bibliotecas Fundamentais da Análise de Dados

1. 🐼 Pandas – Para manipulação de dados tabulares
Usamos para ler arquivos, organizar colunas, fazer filtros, agrupamentos e estatísticas.

É como o "Excel turbinado" do Python.

2.  NumPy – Para cálculos com vetores, matrizes e funções matemáticas

Ele é o núcleo matemático. Trabalha com arrays e oferece funções estatísticas, lógicas e de álgebra.





Bibliotecas Fundamentais da Análise de Dados

3. Matplotlib – A biblioteca base de visualizações

Permite criar gráficos personalizados, de linha, barra, pizza, etc.

4. Seaborn – Visualizações estatísticas mais bonitas e simples

Funciona em cima do Matplotlib. Ideal para exploração visual dos dados com pouco código.

5. Plotly Express – Gráficos interativos e modernos

Para dashboards e visualizações mais sofisticadas. Os gráficos são interativos (passar o mouse, clicar, dar zoom, etc.).





Como elas se encaixam no fluxo da análise de dados

📁 Coleta dos dados



🐼 Pandas → organiza e limpa os dados

📊 NumPy → faz os cálculos matemáticos

📈 Matplotlib / Seaborn / Plotly → mostram os resultados



Revisão antes de iniciar com as bibliotecas

Listas – Guardando conjuntos de dados

```
# Exemplo de lista com nomes
nomes = ["Ana", "Bruno", "Carlos", "Daniela", "Eduardo"]

# Acessando um valor
print(nomes[0])  # Ana
print(nomes[2])  # Carlos
```

👉 Pense em uma coluna de dados com vários valores. No Pandas, você vai ver muito isso!



Revisão antes de iniciar com as bibliotecas

◆ 2. Dicionários – Dados em pares (chave e valor)

Dicionários são usados para representar informações estruturadas. E eles se parecem muito com tabelas.

```
peessoa = {  
    "nome": "Ana",  
    "idade": 23,  
    "cidade": "São Paulo"  
}
```

```
print(peessoa["nome"])    # Ana  
print(peessoa["idade"])  # 23
```

👉 Um DataFrame do Pandas pode ser criado a partir de um dicionário de listas, onde cada chave representa o nome da coluna.



Revisão antes de iniciar com as bibliotecas

◆ 3. Funções – Reaproveitando lógica

Funções são blocos de código que podemos reaproveitar.

```
def saudacao(nome):  
    return f"Olá, {nome}"
```

```
print(saudacao("Carlos")) # Olá, Carlos
```

👉 No Pandas, vamos usar métodos, que funcionam como funções aplicadas aos dados:



Revisão antes de iniciar com as bibliotecas

◆ 4. Objetos e Métodos

No Pandas, quase tudo é um objeto. Por exemplo, um DataFrame é um objeto com diversos métodos (ações).

```
# exemplo simples de string, que também é um objeto  
texto = "ciência de dados"  
print(texto.upper()) # transforma em maiúsculas
```

👉 No Pandas, vamos usar métodos, que funcionam como funções aplicadas aos dados:



Revisão antes de iniciar com as bibliotecas

Exercícios

1. Dada a lista de nomes, crie uma nova lista com apenas os nomes com mais de 5 letras

```
nomes = ["Ana", "Bruno", "Carla", "Daniela", "Eva", "Fernanda", "Igor"]
```

Resultado esperado: ["Bruno", "Carla", "Daniela", "Fernanda"]

2. Dada a lista de notas de alunos, calcule a média apenas dos alunos que tiraram nota maior ou igual a 7

```
notas = [5.5, 8.2, 6.0, 7.0, 9.5, 3.0, 7.5]
```

Resultado esperado: média das notas maiores ou iguais a 7



Revisão antes de iniciar com as bibliotecas

Exercícios

```
# 3. Escreva uma função que recebe uma lista de notas e retorna:  
# a média e o conceito (A: >= 9, B: >= 7, C: >= 5, D: <5)
```

```
def analisar_notas(lista_notas):  
    pass
```

```
# Exemplo:  
# analisar_notas([7, 8, 9]) → (8.0, "B")
```

```
# 4. Dada a lista de notas de alunos, calcule a média apenas dos alunos  
# que tiraram nota maior ou igual a 7
```

```
notas = [5.5, 8.2, 6.0, 7.0, 9.5, 3.0, 7.5]
```

```
# Resultado esperado: média das notas maiores ou iguais a 7
```



Revisão antes de iniciar com as bibliotecas

Exercícios

5. Dada uma lista de dicionários com informações de pessoas, calcule a média das idades.

```
dados = [  
    {"nome": "Ana", "idade": 23},  
    {"nome": "Bruno", "idade": 25},  
    {"nome": "Carlos", "idade": 30},  
    {"nome": "Diana", "idade": 20}  
]  
# Resultado esperado: média das idades (24.5)
```

Exemplo:

analisar_notas([7, 8, 9]) → (8.0, "B")

6. A partir da mesma lista acima, filtre apenas as pessoas com idade acima de 24.



1. Introdução ao Pandas

 Introdução Conceitual: Como o Pandas interpreta e organiza os dados
Estruturas de dados poderosas e flexíveis (Series e DataFrames)

O pandas é uma biblioteca do Python projetada para manipulação e análise de dados tabulares e rotulados. Ele é uma das ferramentas mais poderosas na ciência de dados por permitir trabalhar com dados estruturados de forma eficiente e intuitiva, muito parecido com o que fazemos em planilhas (como Excel) ou tabelas em bancos de dados.



1. Introdução ao Pandas

`pd.Series()` — O que é?

A serie é uma estrutura do pandas, que armazena uma sequência de dados com rótulos (índices). Ela é semelhante a uma lista ou a um vetor do NumPy, mas com a vantagem de possuir um índice explícito e funcionalidades avançadas.

 Sintaxe básica:

```
pd.Series(data=None, index=None, dtype=None, name=None,  
copy=False, fastpath=False)
```



1. Introdução ao Pandas

✓ Parâmetros mais importantes:

Parâmetro	Descrição
<code>data</code>	Os dados a serem armazenados na Series. Pode ser: lista, array, dicionário, escalar (valor único), etc.
<code>index</code>	Define os rótulos das posições (índices) dos dados. Se não for fornecido, será gerado automaticamente de 0 até n-1.
<code>dtype</code>	Tipo de dado da Series (ex: <code>int</code> , <code>float</code> , <code>str</code>). Se omitido, o <code>pandas</code> tenta inferir.
<code>name</code>	Um nome opcional para a Series (útil em operações com DataFrame).
<code>copy</code>	Se <code>True</code> , força a cópia dos dados mesmo se já forem uma estrutura compatível.



1. Introdução ao Pandas

`pd.Series()` — O que é?

A serie é uma estrutura do pandas, que armazena uma sequência de dados com rótulos (índices). Ela é semelhante a uma lista ou a um vetor do NumPy, mas com a vantagem de possuir um índice explícito e funcionalidades avançadas.

 Sintaxe básica:

```
pd.Series(data=None, index=None, dtype=None, name=None,  
copy=False, fastpath=False)
```



1. Introdução ao Pandas

Exemplos práticos

1. Criando uma Series a partir de uma lista
2. Definindo um índice personalizado
3. Criando a partir de um dicionário
4. Criando a partir de um valor escalar (repetido)
5. Usando o parâmetro dtype e name



1. Introdução ao Pandas

✓ O que é um DataFrame?

Um DataFrame é a estrutura bidimensional (tabela) principal do pandas. Ele é parecido com:

Uma planilha do Excel

Uma tabela de banco de dados

Uma matriz rotulada



1. Introdução ao Pandas

✓ O que é um DataFrame?

Cada coluna de um DataFrame é uma Series, e você pode trabalhar com elas individualmente ou com a tabela inteira de forma vetorial e poderosa.

■ Sintaxe: `pd.DataFrame()`

```
pd.DataFrame(data=None, index=None, columns=None, dtype=None, copy=False)
```



1. Introdução ao Pandas

Parâmetros principais:

Parâmetro	O que faz
<code>data</code>	Os dados em si. Pode ser: lista de listas, lista de dicionários, dicionário de listas, array NumPy, Series, ou outro DataFrame
<code>index</code>	Rótulos das linhas. Se omitido, será criado um índice de 0 até n-1.
<code>columns</code>	Rótulos das colunas. Se omitido, o Pandas tentará inferir.
<code>dtype</code>	Tipo de dado geral (opcional). Ex: <code>float</code> , <code>int</code> , <code>str</code>
<code>copy</code>	Se <code>True</code> , força uma cópia dos dados.



1. Introdução ao Pandas

Parâmetros principais:

 Exemplos

1. Criando um DataFrame a partir de um dicionário de listas
2. Definindo um índice personalizado
3. Criando a partir de uma lista de dicionários
5. Com uma única Series (coluna)



1. 📦 Introdução ao Pandas

📌 Analogias

Pandas	Conceito real
<code>Series</code>	Coluna de uma planilha (com índice)
<code>DataFrame</code>	Planilha inteira ou tabela SQL
<code>index</code>	Cabeçalho vertical (linhas)
<code>columns</code>	Cabeçalho horizontal (colunas)



1. Introdução ao Pandas

Parâmetros principais:

 Exemplos

1. Criando um DataFrame a partir de um dicionário de listas
2. Definindo um índice personalizado
3. Criando a partir de uma lista de dicionários
5. Com uma única Series (coluna)



1. Introdução ao Pandas

Atividades :

Atividade 1: Crie uma lista com os nomes de cinco frutas e transforme essa lista em uma Series do pandas.

Atividade 2: Crie uma lista com os valores de temperatura ao longo de 7 dias. Crie uma Series que use os dias da semana como índice.

Atividade 3: Crie uma Series usando um dicionário onde as chaves são nomes de alunos e os valores são suas notas finais.

Atividade 4: Crie uma Series com os números de 1 a 5 e defina índices personalizados em formato de letras.



1. Introdução ao Pandas

Atividades :

Atividade 5: Crie um dicionário com os dados de três pessoas contendo nome, idade e cidade. Use esse dicionário para criar um DataFrame.

Atividade 6: Crie uma lista de dicionários representando três livros, com as chaves: "título", "autor", "ano". Use essa lista para criar um DataFrame.

Atividade 7: Crie um DataFrame a partir de uma lista de listas. Os dados devem representar [nome, idade] de três pessoas. Defina os nomes das colunas como "Nome" e "Idade".

Atividade 8: Crie um DataFrame usando um array do NumPy com números inteiros de 1 a 9, em uma matriz 3x3. Defina os nomes das colunas como "A", "B", "C".




1. Introdução ao Pandas

Atividades :

Atividade 9: Crie um DataFrame a partir de um dicionário de Series. Cada chave do dicionário deve representar uma coluna.





(85) 98524-9935  youthspace

 contato@youthidiomas.com.br

<https://www.youthspace.com.br/>

2023

