**Douglas Pereira de Araujo**

**Felipe Dal Molin**

**Gabriela Germana Da Silva**

**Samuel Regis Nascimento Barbosa**

**Projeto Aplicado III**

# Algoritmo de Recomendação para indicação de Livros em Python

**São Paulo**

**2023**

# Sumário

**Cronograma**

| Etapa | Descrição da atividade | Prazo |
|---|---|---|
| Etapa 1 | Montagem do grupo<br>Escolha da temática<br>Escolha do Dataset<br>Organização dos documentos no repositório | 10/08 a 27/08 |
| Etapa 2 | Elaboração da proposta analítica<br>Tratar e preparar a base de dados<br>Definir a técnica para o treinamento do modelo<br>Validação do Modelo<br>Descrever o referencial teórico para a elaboração do projeto.<br>Apresentação dos Scripts da Análise Exploratória em Python<br>Construção gráfica dos resultados | 28/08 a 21/09 |
| Etapa 3 | Ajustar o pipeline de treinamento para o resultado<br>Reavaliar o desempenho do modelo.<br>Descrever a metodologia aplicada. | 23/09 a 20/10 |
| Etapa 4 | Analisar os resultados obtidos<br>Descrever e documentar os resultados<br>Finalização do data Storytelling<br>Descrever e documentar as conclusões e os trabalhos futuros.<br>Ajuste do relatório final<br>Realizar a gravação da apresentação do projeto em vídeo<br>Organizar todos os documentos nos repositórios<br>Entregar todos os arquivos do projeto<br>Vídeo de apresentação do projeto | 24/10 a 01/11 |

**Bibliotecas utilizadas no modelo:**

- ✓ Pandas;
- ✓ Numpy;
- ✓ Matplotlib;
- ✓ Seaborn;
- ✓ Scipy;
- ✓ Requests;
- ✓ Sklearn;
- ✓ Pillow;
- ✓ Warnings

**Introdução**

Descoberta de Conhecimento em Bancos de Dados (KDD), ou Knowledge Discovery in Databases, é um processo interdisciplinar que abrange diversas etapas, desde a seleção e preparação dos dados até a interpretação dos resultados obtidos. É um campo que combina elementos da mineração de dados, aprendizado de máquina, estatísticas e outras disciplinas relacionadas, visando extrair informações úteis e ocultas a partir de grandes conjuntos de dados. O processo de KDD envolve a identificação de padrões, tendências e relações nos dados, resultando na geração de conhecimento que pode ser aplicado em tomadas de decisão, previsões e outras análises significativas.

Os algoritmos de recomendação têm um papel fundamental na análise de dados, especialmente no contexto do comércio eletrônico, plataformas de streaming, redes sociais e muitos outros sistemas onde há uma abundância de opções disponíveis para os usuários. Esses algoritmos buscam prever as preferências ou interesses dos usuários e, com base nessa previsão, sugerir itens ou conteúdos que possam ser do interesse deles. A popularidade e a importância dos algoritmos de recomendação cresceram exponencialmente à medida que as empresas buscam personalizar a experiência do usuário e melhorar a satisfação.

Nesse contexto da análise de dados, escolhemos a aplicação de algoritmos de recomendação para livros, que tem desempenhado um papel significativo em aprimorar a experiência dos leitores e promover a descoberta literária. Esses sistemas de recomendação para livros não apenas facilitam a descoberta de novas obras, mas também contribuem para a construção de uma comunidade de leitores ao redor de interesses comuns. Ao combinar os princípios de KDD e algoritmos de recomendação, esses sistemas proporcionam uma experiência de leitura mais enriquecedora e diversificada, conectando leitores a livros que poderiam passar despercebidos em meio a uma vasta oferta literária.

Os princípios e os processos envolvidos na descoberta de padrões úteis nos dados, abrangem duas técnicas de identificação das informações. Os algoritmos supervisionados são projetados para trabalhar com dados rotulados, ou seja, conjuntos de dados em que cada exemplo é associado a um rótulo ou classe conhecida. Em contraste, os algoritmos não supervisionados lidam com dados não

rotulados, explorando a estrutura subjacente dos dados para identificar padrões, agrupamentos e características intrínsecas. Eles buscam organizar os dados de maneira significativa, revelando informações sobre similaridades e diferenças entre os exemplos sem a orientação de rótulos predefinidos. Enquanto os algoritmos supervisionados são aplicados em tarefas de classificação e regressão, os algoritmos não supervisionados são frequentemente utilizados em tarefas de clusterização proporcionando uma compreensão mais profunda dos dados e suas relações.

Para o projeto proposto foi escolhido o método k-NN, um sistema que visa entender os gostos e preferências de leitores individuais, utilizando dados como histórico de leitura, classificações e interações com outros livros. Com base nessas informações o algoritmo busca aprender padrões e relações entre títulos, autores e gêneros nos dados de treinamento para fazer previsões ou classificações precisas em novos dados, com base nas informações aprendidas durante o treinamento.

O k-NN (k-vizinhos mais próximos) é um método utilizado em aplicações de classificação que considera que os registros do conjunto de dados correspondem a pontos no espaço, onde cada atributo representa uma dimensão desse espaço. Quando um novo registro precisa ser classificado, ele é comparado com todos os registros do conjunto de treinamento para identificar os k vizinhos mais próximos. A classe do novo registro é determinada pela classe mais frequente entre esses vizinhos mais próximos, podendo variar de acordo com a métrica escolhida. O valor de k é um parâmetro de entrada do método, e as métricas mais comuns utilizadas são a Euclidiana e a de Manhattan. A escolha do valor de k depende do conjunto de dados e pode ser determinada por técnicas como validação cruzada e bootstrap e precisam ser considerados pois impactam na sensibilidade ao ruído e na definição das fronteiras entre as classes.

## Metodologia do Projeto de Recomendação de Livros

### Importação das Bibliotecas

```
# Importação de Bibliotecas
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings

# Configurações Iniciais
warnings.filterwarnings('ignore')
pd.set_option('display.max_rows', 100)
pd.set_option('display.max_columns', 50)
plt.rcParams['figure.figsize'] = (15, 6)
plt.style.use('seaborn-darkgrid')
```

### Coleta de Dados:

Foram coletados três conjuntos de dados em formato CSV: Books.csv, Ratings.csv e Users.csv.

O conjunto de dados Books.csv contém informações sobre os livros, incluindo título, autor, ano de publicação, editora e URLs da imagem, o conjunto de dados Ratings.csv registra as avaliações dos usuários para os livros e o conjunto de dados Users.csv contém informações demográficas sobre os usuários, como localização e idade.

```
# Ler os dados
Dados_Livros = pd.read_csv('Books.csv')
Dados_Avaliacao = pd.read_csv('Ratings.csv')
Dados_Usuario = pd.read_csv('Users.csv')

# Dimensçao [ Linhas, Colunas ]
Dados_Livros.shape, Dados_Avaliacao.shape, Dados_Usuario.shape
```

**Modelagem dos dados**

Na fase de modelagem do nosso projeto, um passo crucial foi combinar diferentes conjuntos de dados. Essa etapa, chamada de cruzamento de dados, foi essencial para criar o modelo que recomendará livros aos usuários.

Cruzamento dos Dados

1º Cruzamento:

No primeiro cruzamento de dados, juntamos informações sobre livros e avaliações dos usuários. Esses dados foram combinados para criar uma base que relaciona livros e as opiniões dos leitores. Isso foi feito usando o número de identificação dos livros (ISBN) como chave para conectar as informações.

2º Cruzamento:

No segundo cruzamento de dados, adicionamos informações sobre os próprios usuários. Combinamos os dados já consolidados no primeiro cruzamento com os perfis dos leitores. Ao conectar essas informações usando os IDs dos usuários, conseguimos entender melhor as preferências individuais de leitura. Isso tornou nosso modelo de recomendação altamente personalizado, levando em conta as preferências únicas de cada usuário.

Esses cruzamentos criaram uma tabela única que se tornou a base do nosso modelo de recomendação de livros. Esta tabela contém informações sobre livros, avaliações de usuários e os perfis dos leitores.

```
# Cruzamentos dos dados

# 1° Cruzamento
Tab_Cruzada = Dados_Livros.merge( Dados_Avaliacao, how='inner',
on='ISBN')

# 2° Cruzamento
Tab_Cruzada = Tab_Cruzada.merge( Dados_Usuario, how='inner', on='User-
ID')

# Dimensão
Tab_Cruzada.shape
```

## 2. Exploração de Dados

Uma visualização inicial dos conjuntos de dados foi realizada para entender a estrutura e o conteúdo dos dados. Foi identificado que a abordagem de filtragem colaborativa poderia ser usada para criar o sistema de recomendação, dadas as avaliações dos usuários disponíveis.

```
# Verificar
Tab_Cruzada.head()
```

| | ISBN | Book-Title | Book-Author | Year-Of-Publication | Publisher | Image-URL-S | Image-URL-M | |
|---|---|---|---|---|---|---|---|---|
| 0 | 0195153448 | Classical Mythology | Mark P. O. Morford | 2002 | Oxford University Press | http://images.amazon.com/images/P/0195153448.0... | http://images.amazon.com/images/P/0195153448.0... | http://imag |
| 1 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://imag |
| 2 | 0060973129 | Decision in Normandy | Carlo D'Este | 1991 | HarperPerennial | http://images.amazon.com/images/P/0060973129.0... | http://images.amazon.com/images/P/0060973129.0... | http://imag |
| 3 | 0374157065 | Flu: The Story of the Great Influenza Pandemic... | Gina Bari Kolata | 1999 | Farrar Straus Giroux | http://images.amazon.com/images/P/0374157065.0... | http://images.amazon.com/images/P/0374157065.0... | http://imag |
| 4 | 0393045218 | The Mummies of Urumchi | E. J. W. Barber | 1999 | W. W. Norton & Company | http://images.amazon.com/images/P/0393045218.0... | http://images.amazon.com/images/P/0393045218.0... | http://imag |

```
# Formato das Colunas
Tab_Cruzada.dtypes
```

```
ISBN                  object
Book-Title            object
Book-Author           object
Year-Of-Publication  float64
Publisher             object
Image-URL-S           object
Image-URL-M           object
Image-URL-L           object
User-ID                int64
Book-Rating            int64
Location              object
Age                  float64
dtype: object
```

```
# Verificando
Tab_Cruzada['Location'].head(5)
```

```
0     stockton, california, usa
1      timmins, ontario, canada
2      timmins, ontario, canada
3      timmins, ontario, canada
4      timmins, ontario, canada
Name: Location, dtype: object
```

```
# Verificando
Tab_Cruzada['Location'].tail(5)
```

```
1031131                        venice, florida, usa
1031132                     tioga, pennsylvania, usa
1031133                        madrid, madrid, spain
1031134                   grand prairie, texas, usa
1031135    bielefeld, nordrhein-westfalen, germany
Name: Location, dtype: object
```

```
# Tecnica de tratamento de texto
def Extrair_Pais( Regiao ):
  '''
    Função para extrair o nome do pais na coluna região
  '''
  # Incluindo a inforção
  Registro = Regiao
  # Fatiar
  Registro = Regiao.split(',')
  # Buscar
  Fracao = Registro[-1].upper()
  #Retorno
  return Fracao

# Criando a coluna
Tab_Cruzada['Pais'] = Tab_Cruzada['Location'].apply( Extrair_Pais )

# Verificando
Tab_Cruzada.head()
```

| | ISBN | Book-Title | Book-Author | Year-Of-Publication | Publisher | Image-URL-S | Image-URL-M | |
|---|---|---|---|---|---|---|---|---|
| 0 | 0195153448 | Classical Mythology | Mark P. O. Morford | 2002.0 | Oxford University Press | http://images.amazon.com/images/P/0195153448.0... | http://images.amazon.com/images/P/0195153448.0... | http://image |
| 1 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001.0 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://image |
| 2 | 0060973129 | Decision in Normandy | Carlo D'Este | 1991.0 | HarperPerennial | http://images.amazon.com/images/P/0060973129.0... | http://images.amazon.com/images/P/0060973129.0... | http://image |
| 3 | 0374157065 | Flu: The Story of the Great Influenza Pandemic... | Gina Bari Kolata | 1999.0 | Farrar Straus Giroux | http://images.amazon.com/images/P/0374157065.0... | http://images.amazon.com/images/P/0374157065.0... | http://image |
| 4 | 0393045218 | The Mummies of Urumchi | E. J. W. Barber | 1999.0 | W. W. Norton & Company | http://images.amazon.com/images/P/0393045218.0... | http://images.amazon.com/images/P/0393045218.0... | http://image |

```
# Nulos
Tab_Cruzada.isnull().sum()
```

```
ISBN                      0
Book-Title                0
Book-Author               1
Year-Of-Publication       4
Publisher                 2
Image-URL-S               0
Image-URL-M               0
Image-URL-L               4
User-ID                   0
Book-Rating               0
Location                  0
Age                  277835
Pais                      0
dtype: int64
```

```
# Unicos
Tab_Cruzada.nunique()
```

```
ISBN                   270151
Book-Title             241071
Book-Author            101588
Year-Of-Publication       116
Publisher               16729
Image-URL-S            269842
Image-URL-M            269842
Image-URL-L            269839
User-ID                 92106
Book-Rating                11
Location                22480
Age                       141
Pais                      288
dtype: int64
```

```
# Renomar as colunas
Tab_Cruzada.rename(
    columns={
        'Book-Title' : 'Titulo',
        'Book-Author' : 'Autor',
        'Year-Of-Publication' : 'Ano_Publicacao',
        'Publisher' : 'Editora',
        'User-ID' :'Id_Cliente',
        'Book-Rating' : 'Avaliacao',
        'Location' : 'Localizacao',
        'Age' : 'Idade'
    }, inplace=True
)

# Verificar
Tab_Cruzada.columns
```

```
Index(['ISBN', 'Titulo', 'Autor', 'Ano_Publicacao', 'Editora', 'Image-URL-S',
       'Image-URL-M', 'Image-URL-L', 'Id_Cliente', 'Avaliacao', 'Localizacao',
       'Idade', 'Pais'],
      dtype='object')
```

**Análise Exploratória dos dados**

```
# Analise descritiva
Tab_Cruzada.describe()
```

| | Ano_Publicacao | Id_Cliente | Avaliacao | Idade |
|---|---|---|---|---|
| **count** | 1.031132e+06 | 1.031136e+06 | 1.031136e+06 | 753301.000000 |
| **mean** | 1.968195e+03 | 1.405945e+05 | 2.839051e+00 | 37.397648 |
| **std** | 2.311015e+02 | 8.052466e+04 | 3.854157e+00 | 14.098254 |
| **min** | 0.000000e+00 | 2.000000e+00 | 0.000000e+00 | 0.000000 |
| **25%** | 1.992000e+03 | 7.041500e+04 | 0.000000e+00 | 28.000000 |
| **50%** | 1.997000e+03 | 1.412100e+05 | 0.000000e+00 | 35.000000 |
| **75%** | 2.001000e+03 | 2.114260e+05 | 7.000000e+00 | 45.000000 |
| **max** | 2.050000e+03 | 2.788540e+05 | 1.000000e+01 | 244.000000 |

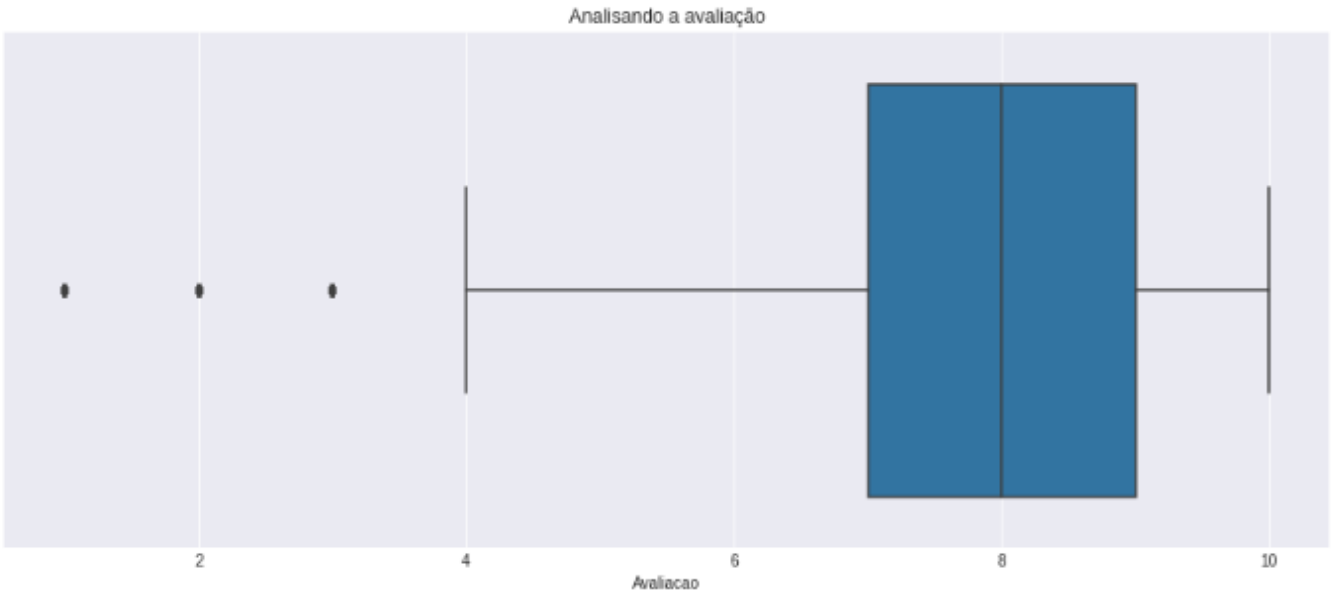
```
# Remover as avaliações zeradas
Tab_Cruzada = Tab_Cruzada.loc[ Tab_Cruzada['Avaliacao'] > 0 ]

# Veriicar
Tab_Cruzada.isnull().sum(), Tab_Cruzada.shape
```

```
(ISBN                  0
 Titulo                0
 Autor                 1
 Ano_Publicacao        1
 Editora               2
 Image-URL-S           0
 Image-URL-M           0
 Image-URL-L           1
 Id_Cliente            0
 Avaliacao             0
 Localizacao           0
 Idade            114221
 Pais                  0
 dtype: int64, (383842, 13))
```

```
# Verificar
Tab_Cruzada['Avaliacao'].describe()
```

```
count    383842.000000
mean          7.626701
std           1.841339
min           1.000000
25%           7.000000
50%           8.000000
75%           9.000000
max          10.000000
Name: Avaliacao, dtype: float64
```

```
# Analise grafica
plt.title('Analisando a avaliação')
sns.boxplot( data=Tab_Cruzada, x='Avaliacao');
```



```
# Analise
Analise = Tab_Cruzada.groupby( by=['Titulo'] ).agg(
    Quantidade = ('Titulo', 'count'),
    Media = ('Avaliacao', 'mean'),
    Max = ('Avaliacao', 'max'),
    Min = ('Avaliacao', 'min'),
    Mediana = ('Avaliacao', 'median'),
)

# Verificando
Analise.head()
```

| Titulo | Quantidade | Media | Max | Min | Mediana |
|---|---|---|---|---|---|
| A Light in the Storm: The Civil War Diary of Amelia Martin, Fenwick Island, Delaware, 1861 (Dear America) | 1 | 9.000000 | 9 | 9 | 9.0 |
| Ask Lily (Young Women of Faith: Lily Series, Book 5) | 1 | 8.000000 | 8 | 8 | 8.0 |
| Dark Justice | 1 | 10.000000 | 10 | 10 | 10.0 |
| Earth Prayers From around the World: 365 Prayers, Poems, and Invocations for Honoring the Earth | 7 | 7.142857 | 10 | 1 | 7.0 |
| Final Fantasy Anthology: Official Strategy Guide (Brady Games) | 2 | 10.000000 | 10 | 10 | 10.0 |

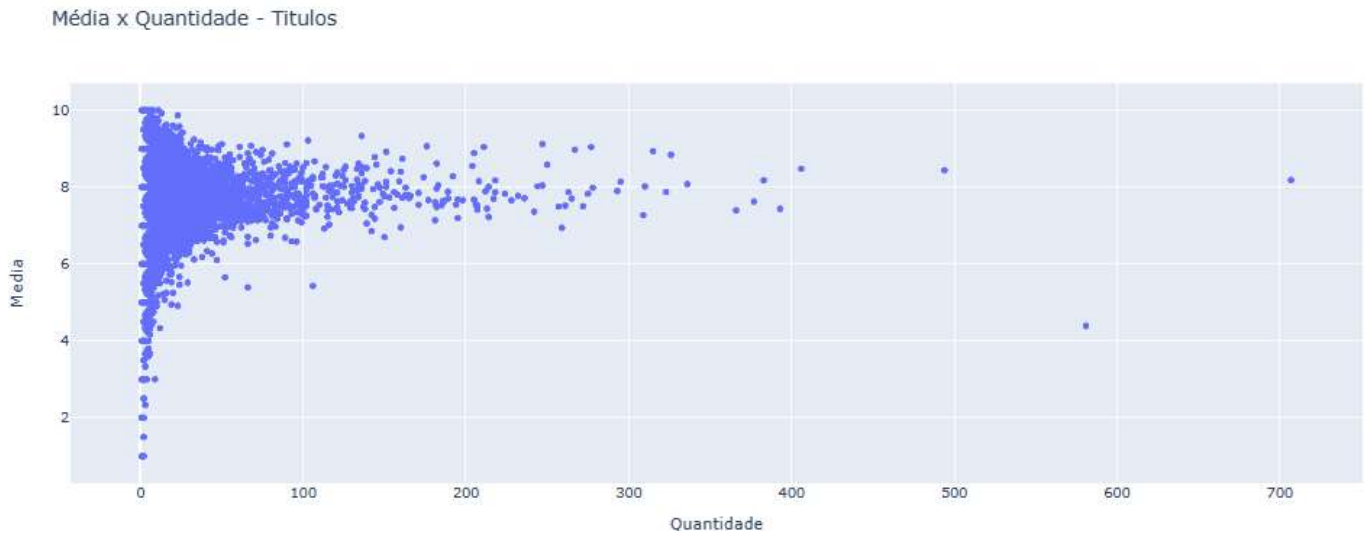
```
# Verificar
Analise.sort_values('Quantidade', ascending=False ).head()
```

| Titulo | Quantidade | Media | Max | Min | Mediana |
|---|---|---|---|---|---|
| The Lovely Bones: A Novel | 707 | 8.185290 | 10 | 1 | 8.0 |
| Wild Animus | 581 | 4.390706 | 10 | 1 | 4.0 |
| The Da Vinci Code | 494 | 8.439271 | 10 | 1 | 9.0 |
| The Secret Life of Bees | 406 | 8.477833 | 10 | 2 | 9.0 |
| The Nanny Diaries: A Novel | 393 | 7.437659 | 10 | 1 | 8.0 |

```
# Vericar
Analise.sort_values(['Media', 'Quantidade' ], ascending=False ).head()
```

| Titulo | Quantidade | Media | Max | Min | Mediana |
|---|---|---|---|---|---|
| **Postmarked Yesteryear: 30 Rare Holiday Postcards** | 11 | 10.0 | 10 | 10 | 10.0 |
| **The Sneetches and Other Stories** | 8 | 10.0 | 10 | 10 | 10.0 |
| **Natural California: A Postcard Book** | 7 | 10.0 | 10 | 10 | 10.0 |
| **Uncle John's Supremely Satisfying Bathroom Reader (Uncle John's Bathroom Reader)** | 7 | 10.0 | 10 | 10 | 10.0 |
| **Oh, the Thinks You Can Think! (I Can Read It All by Myself Beginner Books)** | 6 | 10.0 | 10 | 10 | 10.0 |

```
# Analise Qtd x Avaliacao

px.scatter(
    # Dados
    data_frame=Analise,
    # Parametros
    x='Quantidade', y='Media',
    # Titulo
    title='Média x Quantidade - Titulos',
    # Upgrade
    # marginal_y='rug', marginal_x='histogram'
)
```




,

```
# Correlação
Analise.corr()
```

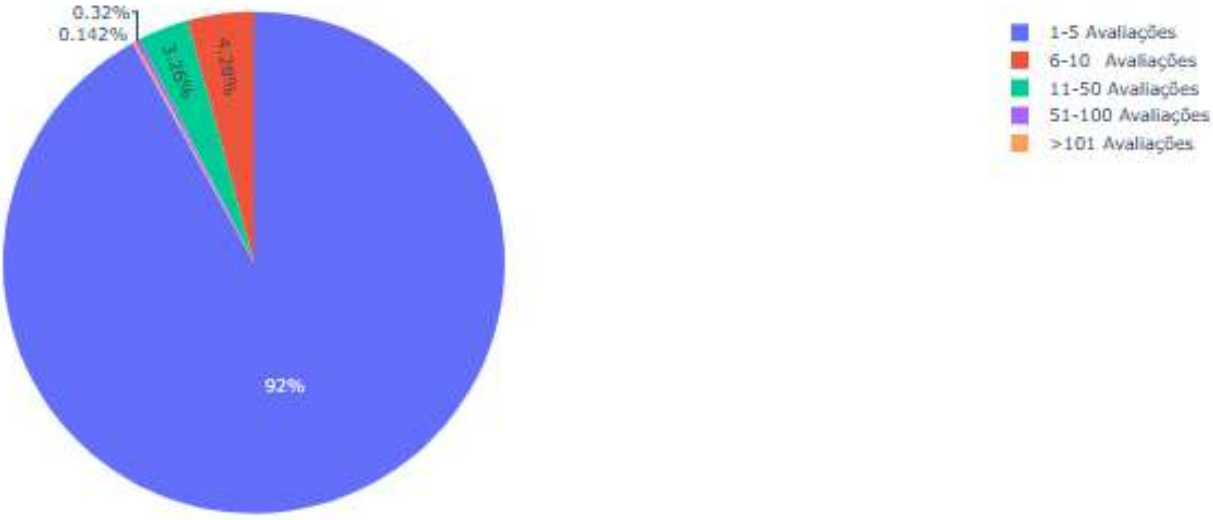
| | Quantidade | Media | Max | Min | Mediana |
|---|---|---|---|---|---|
| **Quantidade** | 1.000000 | 0.018880 | 0.175572 | -0.251497 | 0.036604 |
| **Media** | 0.018880 | 1.000000 | 0.889722 | 0.842385 | 0.989839 |
| **Max** | 0.175572 | 0.889722 | 1.000000 | 0.530760 | 0.887792 |
| **Min** | -0.251497 | 0.842385 | 0.530760 | 1.000000 | 0.804023 |
| **Mediana** | 0.036604 | 0.989839 | 0.887792 | 0.804023 | 1.000000 |

```
# Analise
Analise['Quantidade'].describe()
```

```
count    135567.000000
mean          2.831382
std           9.135691
min           1.000000
25%           1.000000
50%           1.000000
75%           2.000000
max         707.000000
Name: Quantidade, dtype: float64
```

```
#
def Classificao_Quantidade( Quantidade ):
  '''
    Agrupar a quantidade
  '''

  if int( Quantidade ) <= 5:
    return '1-5 Avaliações'

  elif int( Quantidade) <=10:
    return '6-10 Avaliações'

  elif int(Quantidade) <= 50:
    return '11-50 Avaliações'

  elif int(Quantidade) <= 100:
    return '51-100 Avaliações'

  else:
    return '>101 Avaliações'

# Aplicação
Pizza = Analise['Quantidade'].apply( Classificao_Quantidade ).value_counts( normalize=True )

# Tranformar em um DataFrame
Pizza = pd.DataFrame( Pizza ).reset_index()

# Plot
px.pie(
    # DAdos
    data_frame=Pizza,
    # Paramewtros
    names='index', values='Quantidade',
    # Titulo
    title='Divisão das Quantidades'
)
```

Distribuição das quantidades em cada categoria



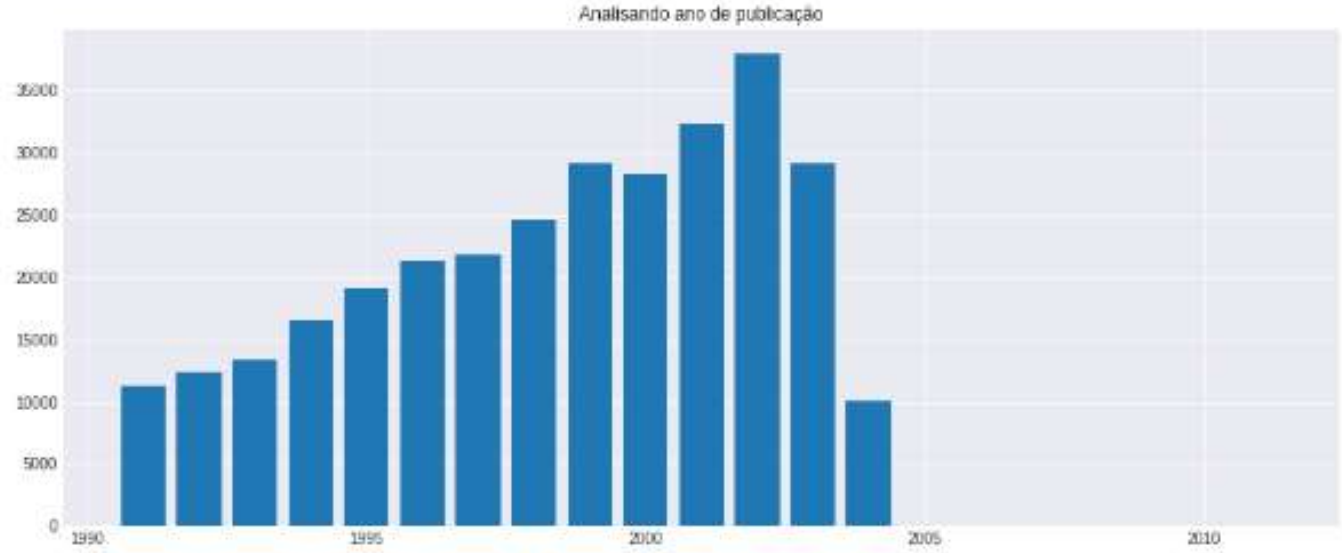

```
# Verificando
Pizza
```

| | index | Quantidade |
|---|---|---|
| 0 | 1-5 Avaliações | 0.920010 |
| 1 | 6-10 Avaliações | 0.042783 |
| 2 | 11-50 Avaliações | 0.032589 |
| 3 | 51-100 Avaliações | 0.003201 |
| 4 | >101 Avaliações | 0.001416 |

```
# Publicação
Anlaise_Ano = Tab_Cruzada['Ano_Publicacao'].value_counts().sort_index().reset_index()

# Verificando
Anlaise_Ano.describe()
```

| | index | Ano_Publicacao |
|---|---|---|
| count | 105.000000 | 105.000000 |
| mean | 1934.028571 | 3655.628571 |
| std | 210.090432 | 8051.797181 |
| min | 0.000000 | 1.000000 |
| 25% | 1937.000000 | 4.000000 |
| 50% | 1963.000000 | 63.000000 |
| 75% | 1989.000000 | 1652.000000 |
| max | 2050.000000 | 37986.000000 |

```
# Plot

# Filtrando o ano
Filtro = Anlaise_Ano.loc[ (Anlaise_Ano['index'] > 1990) & ( Anlaise_Ano['index'] < 2020 ) ]

# Plot
plt.title('Analisando ano de publicação')
plt.bar( Filtro['index'], Filtro['Ano_Publicacao'] );
```



```
Tab_Cruzada.columns
```

```
Index(['ISBN', 'Titulo', 'Autor', 'Ano_Publicacao', 'Editora', 'Image-URL-S',
       'Image-URL-M', 'Image-URL-L', 'Id_Cliente', 'Avaliacao', 'Localizacao',
       'Idade', 'Pais'],
      dtype='object')
```

```
# Ranikg dos Autores
Tab_Cruzada.groupby( by='Autor' ).agg(
    Quantidade = ('Avaliacao', 'count'),
    Media = ('Avaliacao', 'mean'),
).sort_values('Quantidade', ascending=False ).head(10)
```

| Autor | Quantidade | Media |
|---|---|---|
| Stephen King | 4639 | 7.815046 |
| Nora Roberts | 2938 | 7.629680 |
| John Grisham | 2550 | 7.523137 |
| James Patterson | 2387 | 7.697947 |
| J. K. Rowling | 1746 | 8.970218 |
| Mary Higgins Clark | 1677 | 7.503280 |
| Janet Evanovich | 1490 | 7.944966 |
| Dean R. Koontz | 1475 | 7.572203 |
| Anne Rice | 1245 | 7.387952 |
| Sue Grafton | 1235 | 7.722267 |

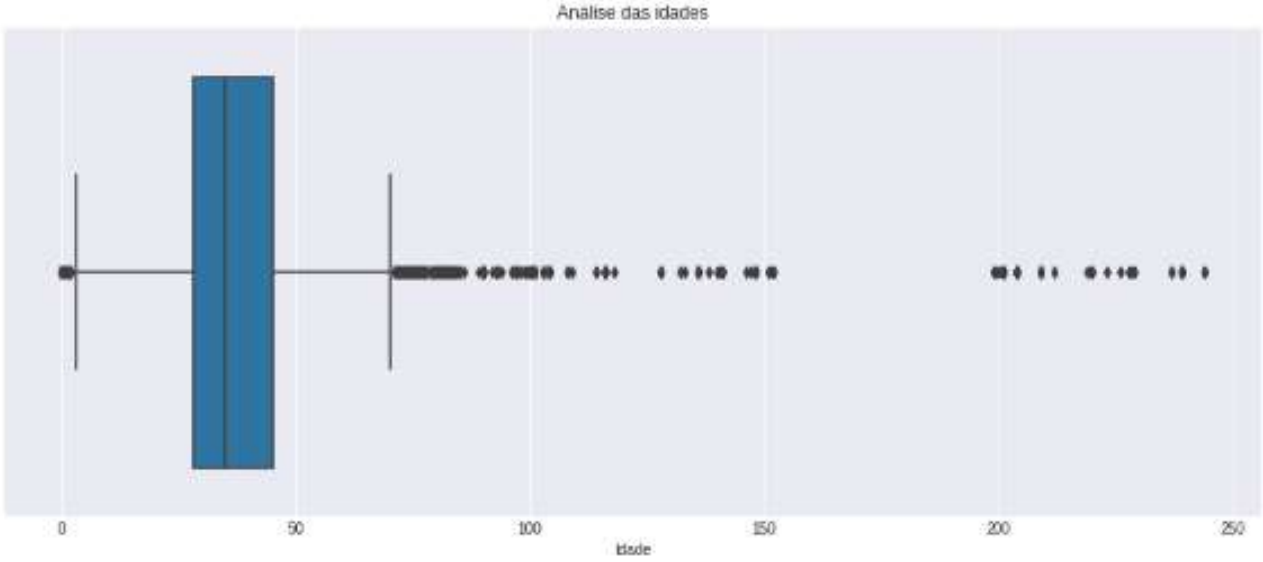
```
# Concetração das avaliações
Tab_Cruzada['Pais'].value_counts( normalize=True ).head(10) * 100
```

```
USA               68.378135
CANADA             9.267876
UNITED KINGDOM     3.854190
GERMANY            3.165625
                   2.737845
SPAIN              1.874990
AUSTRALIA          1.821322
N/A                1.811943
FRANCE             1.207789
PORTUGAL           0.897505
Name: Pais, dtype: float64
```

```
# Concetração das avaliações
Tab_Cruzada['Pais'].value_counts( normalize=True ).cumsum().head(10) * 100
```

```
USA                68.378135
CANADA             77.646011
UNITED KINGDOM     81.500201
GERMANY            84.665826
                   87.403671
SPAIN              89.278662
AUSTRALIA          91.099984
N/A                92.911927
FRANCE             94.119716
PORTUGAL           95.017221
Name: Pais, dtype: float64
```

```
# Idade
plt.title('Análise das idades')
sns.boxplot( data=Tab_Cruzada, x='Idade' );
```



**Construção do modelo**

O Aprendizado por Representação é a busca por melhores representações de dados em algoritmos, muitas vezes não supervisionados. Exemplos incluem Análise de Componentes Principais e Clustering. Esses métodos transformam dados de entrada para preservar informações úteis, útil como pré-processamento para classificação e previsão, permitindo reconstrução de dados de origens desconhecidas. Algoritmos buscam representações de baixa dimensão, esparsas ou hierarquias em níveis abstratos. Essencial para máquinas inteligentes que desvendam fatores subjacentes dos dados.

```
# Ajustar ( Avaliação dos Livros --> Tab_Cruzada )

# Ajustando a Tabela de Avaliacoes
Avaliacoes = Analise.reset_index().iloc[:, 0:2]

# Cruzando os dados
Tab_Final = Tab_Cruzada.merge( Avaliacoes, how='inner', on='Titulo' )

# VErificando
Tab_Final.head()
```

| | ISBN | Titulo | Autor | Ano_Publicacao | Editora | Image-URL-S | Image-URL-M | Image-URL-L | Id_Client |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001.0 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | |
| 1 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001.0 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | 1167 |
| 2 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001.0 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | 6754 |
| 3 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001.0 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | 11686 |
| 4 | 0002005018 | Clara Callan | Richard Bruce Wright | 2001.0 | HarperFlamingo Canada | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | http://images.amazon.com/images/P/0002005018.0... | 12362 |

```
# Filtrar
Livros_Avaliados = Tab_Final.loc[ Tab_Final['Quantidade'] >= 50 ]

# Dimensao
Livros_Avaliados.shape
```

```
(65477, 14)
```

```
# Duplicados
Livros_Avaliados.duplicated().sum()
```

```
0
```

```
# Gerar a Metriz
Matriz = Livros_Avaliados.pivot_table( values='Avaliacao', index='Titulo', columns='Id_Cliente' )

# Retirar os NAN
Matriz.fillna( 0, inplace=True)

# VErificar
Matriz.head()
```

| Id_Cliente | 9 | 16 | 26 | 32 | 42 | 51 | 91 | 97 | 99 | 114 | 125 | 165 | 169 | 183 | 185 | 224 | 226 | 242 | 243 | 244 | 254 | 256 | 272 | 280 | 332 | ... | 278633 | 278641 | 278645 | 278653 | 278663 | 278672 | 278683 | 278698 | 278723 | 278732 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Titulo | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1984 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 9.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 1st to Die: A Novel | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2nd Chance | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 Blondes | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 84 Charing Cross Road | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

5 rows × 24863 columns

```
# Verificando
Tab_Cruzada.loc[ Tab_Cruzada['Titulo'] == '1984' ].head()
```

| | ISBN | Titulo | Autor | Ano_Publicacao | Editora | Image-URL-S | Image-URL-M | Image-URL-L | Id_Cliente |
|---|---|---|---|---|---|---|---|---|---|
| 2713 | 0452262933 | 1984 | George Orwell | 1983.0 | Plume Books | http://images.amazon.com/images/P/0452262933.0... | http://images.amazon.com/images/P/0452262933.0... | http://images.amazon.com/images/P/0452262933.0... | 11676 |
| 33641 | 0451519841 | 1984 | George Orwell | 1980.0 | New Amer Library | http://images.amazon.com/images/P/0451519841.0... | http://images.amazon.com/images/P/0451519841.0... | http://images.amazon.com/images/P/0451519841.0... | 7346 |
| 36405 | 0451524934 | 1984 | George Orwell | 1990.0 | Signet Book | http://images.amazon.com/images/P/0451524934.0... | http://images.amazon.com/images/P/0451524934.0... | http://images.amazon.com/images/P/0451524934.0... | 16795 |
| 106795 | 0451519841 | 1984 | George Orwell | 1980.0 | New Amer Library | http://images.amazon.com/images/P/0451519841.0... | http://images.amazon.com/images/P/0451519841.0... | http://images.amazon.com/images/P/0451519841.0... | 197364 |
| 129342 | 0451524934 | 1984 | George Orwell | 1990.0 | Signet Book | http://images.amazon.com/images/P/0451524934.0... | http://images.amazon.com/images/P/0451524934.0... | http://images.amazon.com/images/P/0451524934.0... | 240567 |

Uma matriz representativa de dados foi criada, que provavelmente representa características ou avaliações dos livros.

Essa matriz foi visualizada por meio de um gráfico de dispersão para entender a distribuição e relação dos dados.

```
# Transformação para vetores
from scipy.sparse import csc_matrix
Matriz_Sparse = csc_matrix( Matriz )

Matriz_Sparse
```

```
<651x24863 sparse matrix of type '<class 'numpy.float64'>'
        with 65081 stored elements in Compressed Sparse Column format>
```

```
# Exemplo da função
csc_matrix(  (4, 4), dtype=np.int8 ).toarray()
```

```
array([[0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0]], dtype=int8)
```

**KNN - NEIGHBORS**

O modelo de vizinhos mais próximos é um tipo de algoritmo de aprendizado não supervisionado usado para tarefas de classificação ou regressão. Ele identifica os 'vizinhos' mais próximos de um ponto de dados em um espaço de características, com base em alguma medida de distância.

n_neighbors=5: Este é o número de vizinhos mais próximos a serem considerados. No contexto de recomendação, significa que para cada ponto de dados, o modelo identificará os 5 vizinhos mais próximos.

algorithm='brute': Especifica o algoritmo usado para computar os vizinhos mais próximos. 'Brute' refere-se ao método de força bruta, que envolve o cálculo da distância entre todos os pares de pontos e é simples, mas pode ser ineficiente em grandes conjuntos de dados.

metric='minkowski': Define a métrica de distância utilizada para o cálculo. Minkowski é uma métrica generalizada que inclui a distância Euclidiana (quando o parâmetro de potência p=2) e a distância de Manhattan (quando p=1).

```
# Criar o Modelo
from sklearn.neighbors import NearestNeighbors

# Parametros
Modelo = NearestNeighbors(
    # Quantidade de recomendações
    n_neighbors=5,
    # Algoritmo
    algorithm='brute',
    # metrica de distancia
    metric='minkowski'
)

# Fitrar o modelo
Modelo.fit( Matriz_Sparse )
```

```
NearestNeighbors(algorithm='brute')
```

```
# Recomendações
# Escolher_Livro

# Descobrir Libros Harry
for Posicao, Titulo in enumerate(Matriz.index):

  # Harry
  if 'Harry' in Titulo:
    print( Posicao, Titulo )
```

```
213 Harry Potter and the Chamber of Secrets (Book 2)
214 Harry Potter and the Goblet of Fire (Book 4)
215 Harry Potter and the Order of the Phoenix (Book 5)
216 Harry Potter and the Prisoner of Azkaban (Book 3)
217 Harry Potter and the Sorcerer's Stone (Book 1)
218 Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))
```

```
# Selecionando o Livro ##### CLIENTE COMPROU !!!!!! ######
Selecionar_Livro = Matriz.iloc[ 213, :].values.reshape( 1, -1 )

# Previsão do Modelo
Distancia, Recomendacao = Modelo.kneighbors( Selecionar_Livro )

## AVALIAÇAO / RENTABILIDADE /  SERIES / NOTICIAS

#### RECOMENDAÇÕES ######
# Ver as sugestões
for Loop in range( len(Recomendacao) ):

  print( Matriz.index[ Recomendacao[Loop] ] )
```

```
Index(['Harry Potter and the Chamber of Secrets (Book 2)',
       'Harry Potter and the Prisoner of Azkaban (Book 3)',
       'Harry Potter and the Goblet of Fire (Book 4)',
       'Harry Potter and the Sorcerer's Stone (Book 1)',
       'Don't Stand Too Close to a Naked Man'],
      dtype='object', name='Titulo')
```

```
Distancia
```

```
array([[  0.        , 150.46714296, 150.8951991 , 158.7714115 ,
        165.39481384]])
```

**Treinamento do Modelo**

O modelo foi treinado a partir da utilização do método Euclidiano.

Aqui, um objeto NearestNeighbors é criado com dois parâmetros principais:

n_neighbors = 2: Isso especifica que o modelo deve considerar os 2 vizinhos mais próximos de um ponto de dados ao fazer previsões. Em outras palavras, para cada ponto de dados, o modelo identificará os dois pontos mais próximos no conjunto de dados.

metric='euclidean': Define a métrica de distância a ser utilizada pelo modelo. 'Euclidean' refere-se à distância euclidiana, uma das métricas mais comuns para medir a distância em um espaço multidimensional. A distância euclidiana é a "distância em linha reta" entre dois pontos.

**Framework – Simulação**

Modelo_Exemplo.fit(Dados)

O método fit é usado para treinar o modelo Modelo_Exemplo com os dados fornecidos. Dados deve ser um conjunto de dados numéricos, como um array do NumPy ou um DataFrame do Pandas, que representa as características dos itens (por exemplo, livros, filmes etc.) ou dos usuários.

Durante o treinamento, o modelo aprende a estrutura dos dados e fica pronto para fazer previsões. Neste contexto, "fazer previsões" significa identificar os vizinhos mais próximos de um determinado ponto.

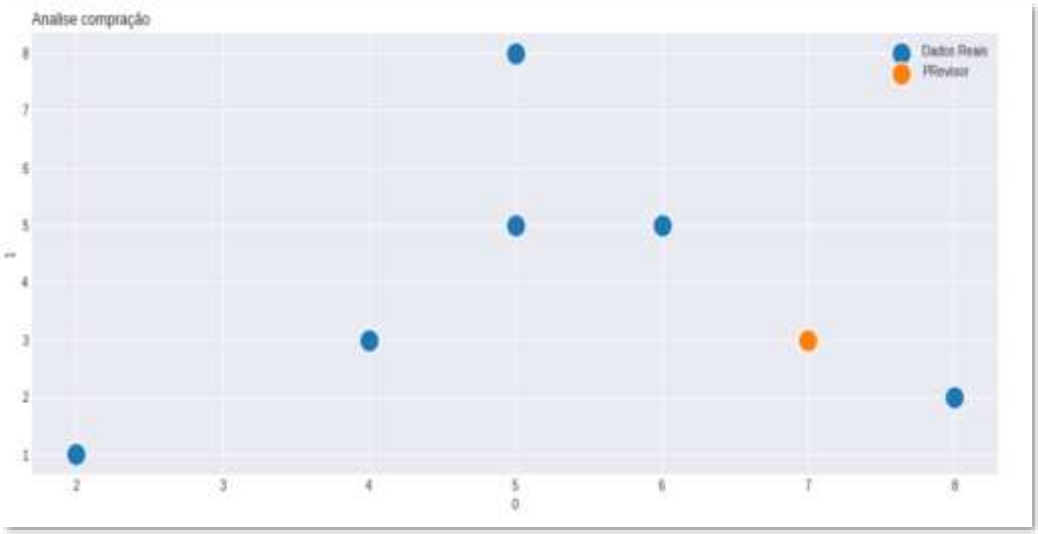
```
1 # Treinar o modelo
2 Modelo_Exemplo = NearestNeighbors( n_neighbors=2, metric='euclidean')
3 Modelo_Exemplo.fit( Dados )
```

```
NearestNeighbors(metric='euclidean', n_neighbors=2)
```

```
# Fazendo a recomendação
Distancias, Indices = Modelo_Exemplo.kneighbors( [[7, 3]] )

print( Distancias )
print( Indices )
```

```
[[1.41421356 2.23606798]]
[[2 4]]
```

```
# Plot
plt.title('Analise compração', loc='left')
sns.scatterplot( data=Tabela_Exemplo, x=0, y=1, s=300)

Previsor = pd.DataFrame( [[7, 3]] )
sns.scatterplot( data=Previsor, x=0, y=1, s=300)
plt.legend(['Dados Reais', 'PRevisor']);
```




```
# Frameworks
# Plotar a imagem
import PIL
import urllib
import requests
import matplotlib.image as mpimg
```

```
# Filtrando o link da capa do Harry poter
Link = Tab_Cruzada.loc[ Tab_Cruzada['Titulo'] == 'Harry Potter and the Chamber of Secrets (Book 2)' ].head(1)['Image-URL-L'].values[0]

# Buscar as informações
Imagem = PIL.Image.open( urllib.request.urlopen( Link ) )

Imagem
```

```
# atribuindo as imagens
Imagem_01 = PIL.Image.open( urllib.request.urlopen( Link_Recomendao_01 ) )
Imagem_02 = PIL.Image.open( urllib.request.urlopen( Link_Recomendao_02 ) )
Imagem_03 = PIL.Image.open( urllib.request.urlopen( Link_Recomendao_03 ) )
Imagem_04 = PIL.Image.open( urllib.request.urlopen( Link_Recomendao_04 ) )
```

```
# Construir relatorio
import plotly.graph_objects as Go
from plotly.subplots import make_subplots
```

```
Titulos = ['Seleção', 'Recomendação 1', 'Recomendação 2', 'Recomendação 3', 'Recomendação 4']

# Criando a Figura
Figura = make_subplots(
    rows=1,
    cols=5,
    subplot_titles=Titulos
)

# Ajustando o layout
Figura.update_layout(
    height=500,
    width=1200,
    title_text='Sistema de recomendação',
    showlegend=False
)

# Imagem da Seleção
Figura.add_trace(
    Go.Image(
        z=Imagem,
    ),
    row=1, col=1
)
```

## Conclusão

Esse projeto acadêmico foi um grande passo para o desenvolvimento de um modelo de recomendação de livros usando análise de dados e aprendizado de máquina. A preparação e o enriquecimento cuidadosos dos dados, além da filtragem criteriosa, criaram uma base sólida para o modelo. A implementação do modelo com a classe NearestNeighbors do Scikit-Learn e a manipulação de estruturas de dados complexas mostraram a aplicabilidade das técnicas de aprendizado de máquina em sistemas de recomendação.

No entanto, é importante reconhecer as limitações desse trabalho. A filtragem de livros com base em um número mínimo de avaliações, embora útil para aumentar a confiabilidade das recomendações, pode ter excluído títulos menos conhecidos ou novos, o que pode limitar a diversidade das recomendações. Além disso, o modelo depende fortemente da qualidade e da integridade dos dados disponíveis, o que significa que quaisquer lacunas ou vieses nos dados podem afetar as recomendações finais.

O projeto pode ser expandido e melhorado de várias maneiras. Uma área de interesse é a incorporação de algoritmos mais sofisticados que possam lidar com a chamada "maldição da dimensionalidade", comum em sistemas de recomendação com grandes conjuntos de dados. Além disso, explorar métodos para integrar avaliações qualitativas, além das quantitativas, poderia enriquecer ainda mais as recomendações. Por fim, seria valioso investigar abordagens para reduzir o viés inerente aos dados de avaliação, garantindo assim que o modelo possa oferecer recomendações equitativas e abrangentes.

Apesar das limitações, o projeto oferece insights valiosos e uma base sólida para futuras pesquisas no campo dos sistemas de recomendação. Ele destaca a importância da análise cuidadosa de dados e da aplicação de técnicas de aprendizado de máquina, ao mesmo tempo em que aponta para desafios e oportunidades de crescimento nessa área em constante evolução.

**Links**

https://github.com/gabigermana/projeto_aplicado_III

https://github.com/Samuelregis/Projeto_Aplicado_3

https://colab.research.google.com/drive/1kb9yZR6q-FTH4fSum1DUXjKFuD-wdIna#scrollTo=-F4_xCwPBoWz

https://www.youtube.com/watch?v=1V1Jpp_TKfQ

## Bibliografia

Book Recommendation Dataset | Kaggle;

https://github.com/Samuelregis/Projeto_Aplicado_3/tree/40e189b869705d3c614591fc8691a3b688aca566;

GOLDSCHMIDT, Ronaldo. Data Mining. [Digite o Local da Editora]: Grupo GEN, 2015. E-book. ISBN 9788595156395. Disponível em: https://app.minhabiblioteca.com.br/#/books/9788595156395/. Acesso em: 24 out. 2023;

SILVA, Leandro Augusto da; PERES, Sarajane M.; BOSCARIOLI, Clodis. Introdução à Mineração de Dados - Com Aplicações em R. [Digite o Local da Editora]: Grupo GEN, 2016. E-book. ISBN 9788595155473. Disponível em: https://app.minhabiblioteca.com.br/#/books/9788595155473/. Acesso em: 24 out. 2023;

https://scikit-learn.org/stable/modules/model_evaluation.html;

https://scikit-learn.org/stable/auto_examples/model_selection/plot_grid_search_digits.html#sphx-glr-auto-examples-model-selection-plot-grid-search-digits-py;

https://medium.com/analytics-vidhya/understanding-k-nearest-neighbour-algorithm-in-detail-fc9649c1d196;