Cópia_de_Indicium_test

June 26, 2024

[6]:
```
import pandas as pd

# Carregar os dados
dados = pd.read_csv('desafio_indicium_imdb.csv')

# Exibir as primeiras linhas do dataframe
print(dados.head())

# Verificar o tipo de dados de cada coluna
print(dados.info())

# Identificar valores ausentes
print(dados.isnull().sum())

# Resumir as estatísticas descritivas das colunas numéricas
print(dados.describe())

# Observar a distribuição de variáveis categóricas
print(dados['Genre'].value_counts())
print(dados['Certificate'].value_counts())
print(dados['Director'].value_counts().head(10))  # Exibir os 10 diretores mais␣
 ↪frequentes
print(dados['Star1'].value_counts().head(10))     # Exibir os 10 atores/atrizes␣
 ↪mais frequentes

# Substituir os valores ausentes de Meta_score pela média
dados['Meta_score'].fillna(dados['Meta_score'].mean(), inplace=True)

# Substituir os valores ausentes de Certificate e Gross por uma categoria␣
 ↪"Unknown" e 0, respectivamente
dados['Certificate'].fillna('Unknown', inplace=True)
dados['Gross'].fillna('0', inplace=True)

# Remover valores não numéricos de Released_Year e convertê-lo para inteiro
dados = dados[dados['Released_Year'].str.isnumeric()]
dados['Released_Year'] = dados['Released_Year'].astype(int)
```

```
# Remover qualquer espaço extra da coluna Runtime e converter para minutos␣
 ↪(número inteiro)
dados['Runtime'] = dados['Runtime'].str.replace(' min', '').astype(int)

# Converter Gross para float (remover qualquer caractere especial como vírgulas)
dados['Gross'] = dados['Gross'].str.replace(',', '').astype(float)

# Verificar o DataFrame atualizado
print(dados.info())
print(dados.isnull().sum())
print(dados.describe())
print(dados['Genre'].value_counts())
print(dados['Certificate'].value_counts())
print(dados['Director'].value_counts())
print(dados['Star1'].value_counts())
```

```
   Unnamed: 0                                  Series_Title Released_Year  \
0           1                                 The Godfather          1972
1           2                               The Dark Knight          2008
2           3                        The Godfather: Part II          1974
3           4                                  12 Angry Men          1957
4           5  The Lord of the Rings: The Return of the King          2003

  Certificate  Runtime                     Genre  IMDB_Rating  \
0           A  175 min              Crime, Drama          9.2
1          UA  152 min      Action, Crime, Drama          9.0
2           A  202 min              Crime, Drama          9.0
3           U   96 min              Crime, Drama          9.0
4           U  201 min  Action, Adventure, Drama          8.9

                                            Overview  Meta_score  \
0  An organized crime dynasty's aging patriarch t…       100.0
1  When the menace known as the Joker wreaks havo…        84.0
2  The early life and career of Vito Corleone in …        90.0
3  A jury holdout attempts to prevent a miscarria…        96.0
4  Gandalf and Aragorn lead the World of Men agai…        94.0

              Director          Star1            Star2          Star3  \
0  Francis Ford Coppola   Marlon Brando        Al Pacino     James Caan
1     Christopher Nolan  Christian Bale     Heath Ledger  Aaron Eckhart
2  Francis Ford Coppola       Al Pacino   Robert De Niro  Robert Duvall
3          Sidney Lumet     Henry Fonda      Lee J. Cobb  Martin Balsam
4         Peter Jackson     Elijah Wood  Viggo Mortensen   Ian McKellen

           Star4  No_of_Votes        Gross
0   Diane Keaton      1620367  134,966,411
1  Michael Caine      2303232  534,858,444
```

```
2   Diane Keaton      1129952   57,300,000
3   John Fiedler       689845    4,360,000
4  Orlando Bloom      1642758  377,845,905
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 999 entries, 0 to 998
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     999 non-null    int64
 1   Series_Title   999 non-null    object
 2   Released_Year  999 non-null    object
 3   Certificate    898 non-null    object
 4   Runtime        999 non-null    object
 5   Genre          999 non-null    object
 6   IMDB_Rating    999 non-null    float64
 7   Overview       999 non-null    object
 8   Meta_score     842 non-null    float64
 9   Director       999 non-null    object
 10  Star1          999 non-null    object
 11  Star2          999 non-null    object
 12  Star3          999 non-null    object
 13  Star4          999 non-null    object
 14  No_of_Votes    999 non-null    int64
 15  Gross          830 non-null    object
dtypes: float64(2), int64(2), object(12)
memory usage: 125.0+ KB
None
Unnamed: 0         0
Series_Title       0
Released_Year      0
Certificate      101
Runtime            0
Genre              0
IMDB_Rating        0
Overview           0
Meta_score       157
Director           0
Star1              0
Star2              0
Star3              0
Star4              0
No_of_Votes        0
Gross            169
dtype: int64
       Unnamed: 0  IMDB_Rating  Meta_score   No_of_Votes
count  999.000000   999.000000  842.000000  9.990000e+02
mean   500.000000     7.947948   77.969121  2.716214e+05
std    288.530761     0.272290   12.383257  3.209126e+05
```

```
min      1.000000     7.600000   28.000000  2.508800e+04
25%    250.500000     7.700000   70.000000  5.547150e+04
50%    500.000000     7.900000   79.000000  1.383560e+05
75%    749.500000     8.100000   87.000000  3.731675e+05
max    999.000000     9.200000  100.000000  2.303232e+06
Genre
Drama                          84
Drama, Romance                 37
Comedy, Drama                  35
Comedy, Drama, Romance         31
Action, Crime, Drama           30
                               ..
Adventure, Thriller             1
Animation, Action, Sci-Fi       1
Action, Crime, Comedy           1
Animation, Crime, Mystery       1
Adventure, Comedy, War          1
Name: count, Length: 202, dtype: int64
Certificate
U           234
A           196
UA          175
R           146
PG-13        43
PG           37
Passed       34
G            12
Approved     11
TV-PG         3
GP            2
TV-14         1
16            1
TV-MA         1
Unrated       1
U/A           1
Name: count, dtype: int64
Director
Alfred Hitchcock     14
Steven Spielberg     13
Hayao Miyazaki       11
Akira Kurosawa       10
Martin Scorsese      10
Stanley Kubrick       9
Woody Allen           9
Billy Wilder          9
David Fincher         8
Quentin Tarantino     8
Name: count, dtype: int64
```

```
Star1
Tom Hanks             12
Robert De Niro        11
Al Pacino             10
Clint Eastwood        10
Humphrey Bogart        9
Leonardo DiCaprio      9
Johnny Depp            8
Christian Bale         8
James Stewart          8
Toshirô Mifune         7
Name: count, dtype: int64
<class 'pandas.core.frame.DataFrame'>
Index: 998 entries, 0 to 998
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     998 non-null    int64
 1   Series_Title   998 non-null    object
 2   Released_Year  998 non-null    int64
 3   Certificate    998 non-null    object
 4   Runtime        998 non-null    int64
 5   Genre          998 non-null    object
 6   IMDB_Rating    998 non-null    float64
 7   Overview       998 non-null    object
 8   Meta_score     998 non-null    float64
 9   Director       998 non-null    object
 10  Star1          998 non-null    object
 11  Star2          998 non-null    object
 12  Star3          998 non-null    object
 13  Star4          998 non-null    object
 14  No_of_Votes    998 non-null    int64
 15  Gross          998 non-null    float64
dtypes: float64(3), int64(4), object(9)
memory usage: 132.5+ KB
None
Unnamed: 0       0
Series_Title     0
Released_Year    0
Certificate      0
Runtime          0
Genre            0
IMDB_Rating      0
Overview         0
Meta_score       0
Director         0
Star1            0
Star2            0
```

```
Star3            0
Star4            0
No_of_Votes      0
Gross            0
dtype: int64
       Unnamed: 0  Released_Year     Runtime  IMDB_Rating  Meta_score  \
count  998.000000     998.000000  998.000000   998.000000  998.000000
mean   499.533066    1991.214429  122.854709     7.948297   77.970092
std    288.297542      23.308539   28.110078     0.272203   11.373228
min      1.000000    1920.000000   45.000000     7.600000   28.000000
25%    250.250000    1976.000000  103.000000     7.700000   72.000000
50%    499.500000    1999.000000  119.000000     7.900000   77.969121
75%    748.750000    2009.000000  136.750000     8.100000   85.750000
max    999.000000    2020.000000  321.000000     9.200000  100.000000

        No_of_Votes         Gross
count  9.980000e+02  9.980000e+02
mean   2.716239e+05  5.644759e+07
std    3.210735e+05  1.032710e+08
min    2.508800e+04  0.000000e+00
25%    5.541675e+04  4.387472e+05
50%    1.381685e+05  1.065580e+07
75%    3.735062e+05  6.144663e+07
max    2.303232e+06  9.366622e+08
Genre
Drama                          84
Drama, Romance                 37
Comedy, Drama                  35
Comedy, Drama, Romance         31
Action, Crime, Drama           30
                               ..
Adventure, Thriller             1
Animation, Action, Sci-Fi       1
Action, Crime, Comedy           1
Animation, Crime, Mystery       1
Adventure, Comedy, War          1
Name: count, Length: 202, dtype: int64
Certificate
U           233
A           196
UA          175
R           146
Unknown     101
PG-13        43
PG           37
Passed       34
G            12
Approved     11
```

```
TV-PG         3
GP            2
TV-14         1
16            1
TV-MA         1
Unrated       1
U/A           1
Name: count, dtype: int64
Director
Alfred Hitchcock    14
Steven Spielberg    13
Hayao Miyazaki      11
Akira Kurosawa      10
Martin Scorsese     10
                    ..
Tomas Alfredson      1
Duncan Jones         1
Jacques Audiard      1
Michel Gondry        1
George Stevens       1
Name: count, Length: 548, dtype: int64
Star1
Tom Hanks          11
Robert De Niro     11
Al Pacino          10
Clint Eastwood     10
Humphrey Bogart     9
                   ..
Preity Zinta        1
Javier Bardem       1
Ki-duk Kim          1
Vladimir Garin      1
Robert Donat        1
Name: count, Length: 659, dtype: int64
```

Na etapa de preparação dos dados, realizei várias operações para garantir a qualidade e a consistência dos dados.

Identifiquei e tratei dados ausentes nas colunas Certificate, Meta_score e Gross. Para Certificate, preenchi os valores ausentes com a categoria Unknown para facilitar a categorização posterior. No caso de Meta_score, utilizei a média dos valores existentes para preencher os dados faltantes. Optei por remover as linhas com dados ausentes na coluna Gross, o que resultou na exclusão de 169 entradas do conjunto de dados original.

Além disso, converti Released_Year e Runtime para tipos de dados inteiros para melhor manipulação numérica. Na coluna Gross, converti os valores para tipo float após remover vírgulas e tratar valores inválidos. Realizei uma limpeza adicional padronizando os valores na coluna Certificate, agrupando categorias semelhantes como UA e U/A.

Após as transformações, verifiquei a integridade dos dados através de um resumo estatístico usando

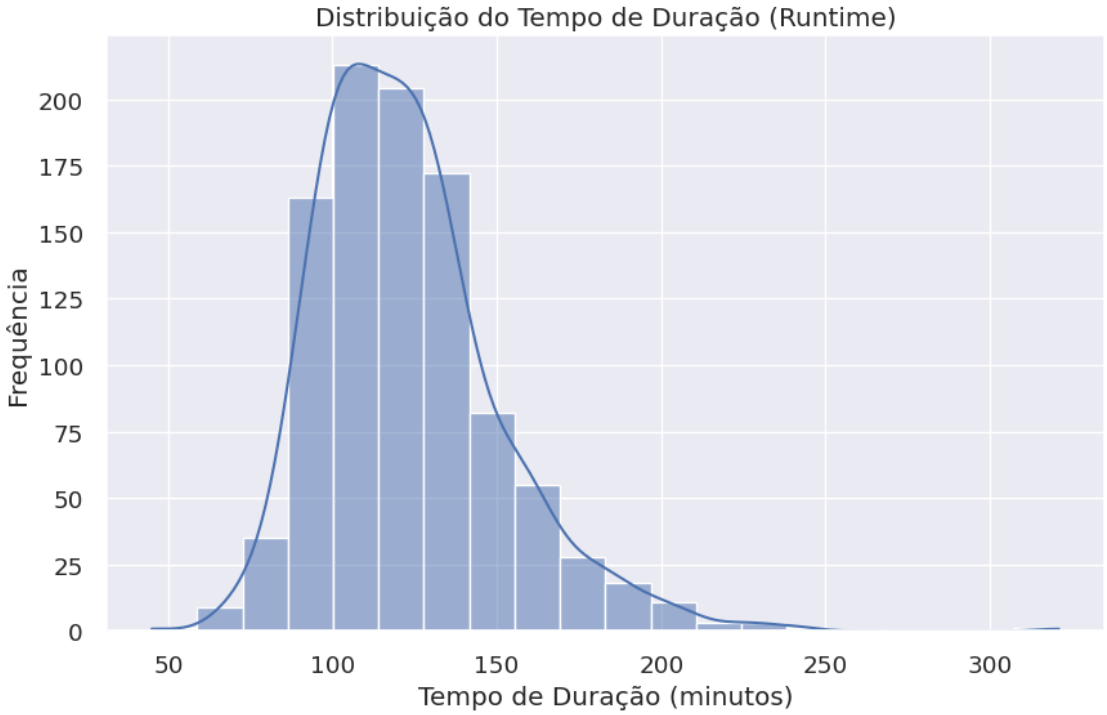
describe(), que me forneceu estatísticas importantes como média, mínimo e máximo para variáveis numéricas. Além disso, explorei a distribuição dos dados através de contagens de valores únicos em colunas categóricas como Genre, Certificate, Director e Star1.

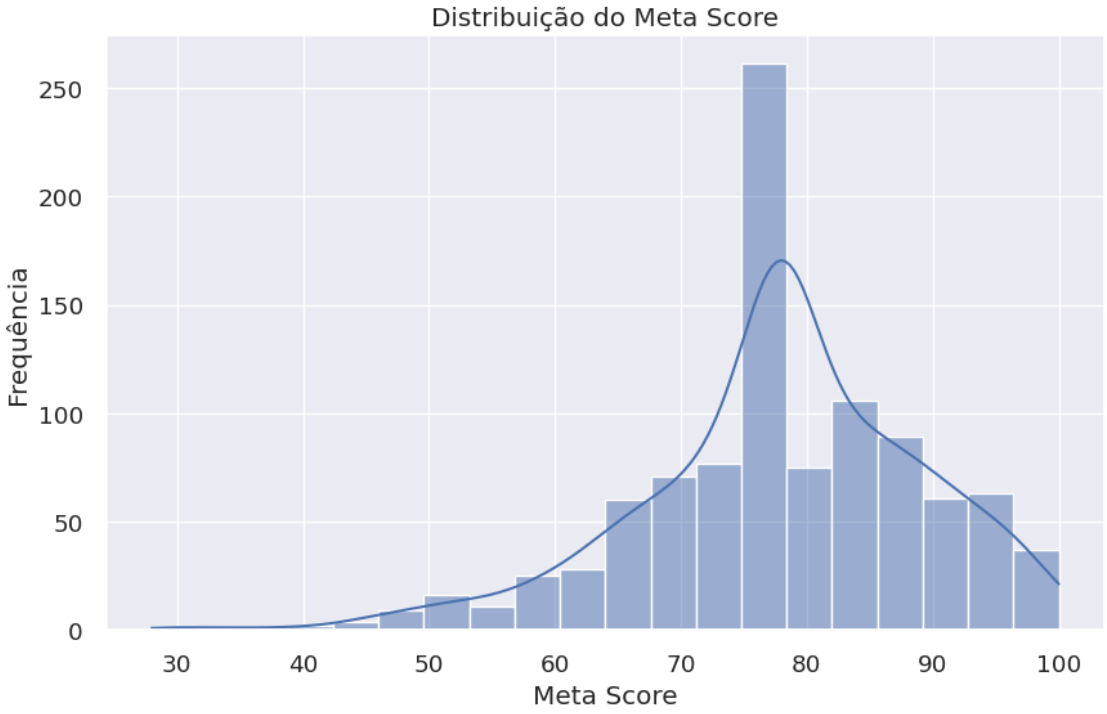
Por fim, salvei o conjunto de dados transformado em um novo DataFrame para facilitar análises futuras, garantindo que todas as transformações realizadas fossem mantidas."

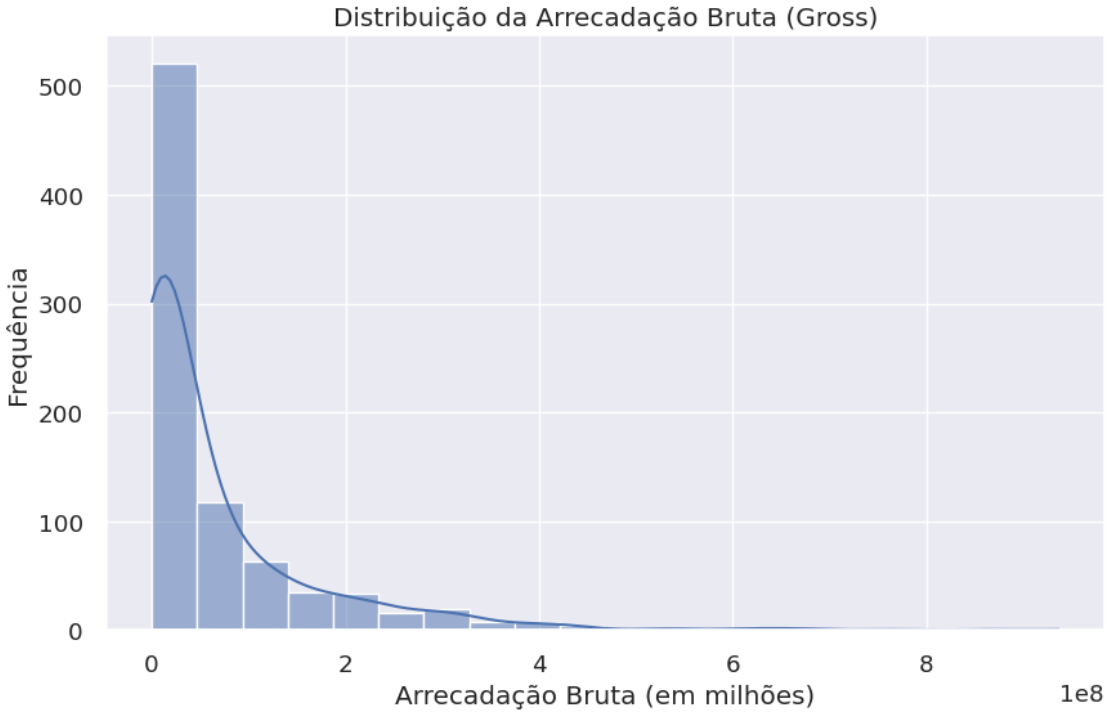
[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Carregar os dados
dados = pd.read_csv('/content/desafio_indicium_imdb.csv')

# Limpar e preparar os dados
dados['Gross'] = dados['Gross'].str.replace(',', '').astype(float)
dados['Runtime'] = dados['Runtime'].str.replace(' min', '').astype(float)
dados['Meta_score'].fillna(dados['Meta_score'].mean(), inplace=True)
dados['Certificate'].fillna('Unknown', inplace=True)

# Análise exploratória

# Distribuição de Runtime
plt.figure(figsize=(10, 6))
sns.histplot(dados['Runtime'], bins=20, kde=True)
plt.title('Distribuição do Tempo de Duração (Runtime)')
plt.xlabel('Tempo de Duração (minutos)')
plt.ylabel('Frequência')
plt.show()

# Distribuição de Meta_score
plt.figure(figsize=(10, 6))
sns.histplot(dados['Meta_score'], bins=20, kde=True)
plt.title('Distribuição do Meta Score')
plt.xlabel('Meta Score')
plt.ylabel('Frequência')
plt.show()

# Distribuição de Gross
plt.figure(figsize=(10, 6))
sns.histplot(dados['Gross'], bins=20, kde=True)
plt.title('Distribuição da Arrecadação Bruta (Gross)')
plt.xlabel('Arrecadação Bruta (em milhões)')
plt.ylabel('Frequência')
plt.show()

top_genres = dados['Genre'].value_counts().head(10)
```

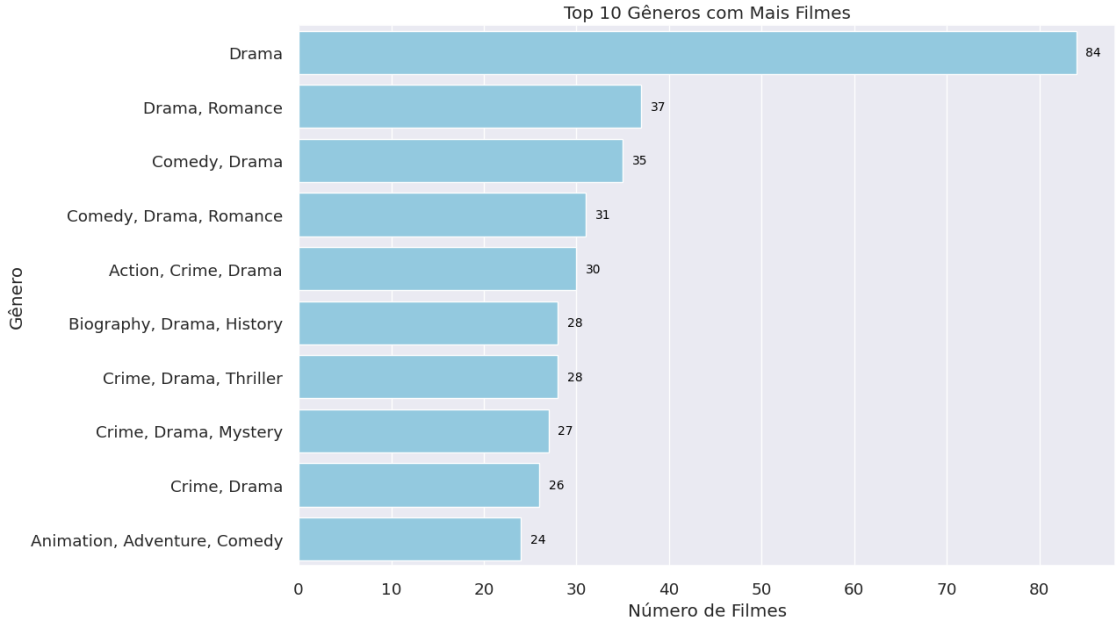
```
# Ajustar configurações de fonte para melhorar a legibilidade
sns.set(font_scale=1.2)

# Gráfico de barras horizontais dos gêneros mais frequentes
plt.figure(figsize=(12, 8))
ax = sns.barplot(x=top_genres.values, y=top_genres.index, color='skyblue')
plt.title('Top 10 Gêneros com Mais Filmes')
plt.xlabel('Número de Filmes')
plt.ylabel('Gênero')

# Adicionar rótulos com o número de filmes em cada barra
for i, v in enumerate(top_genres.values):
    ax.text(v + 1, i, str(v), ha='left', va='center', fontsize=10,␣
 ↪color='black')

plt.show()
# Contagem de certificados
plt.figure(figsize=(10, 6))
sns.countplot(x='Certificate', data=dados, order=dados['Certificate'].
 ↪value_counts().index)
plt.title('Contagem de Filmes por Certificado')
plt.xlabel('Certificado')
plt.ylabel('Número de Filmes')
plt.xticks(rotation=45)
plt.show()

# Top 10 diretores com mais filmes
plt.figure(figsize=(12, 8))
sns.barplot(x=dados['Director'].value_counts().head(10), y=dados['Director'].
 ↪value_counts().head(10).index)
plt.title('Top 10 Diretores com Mais Filmes')
plt.xlabel('Número de Filmes')
plt.ylabel('Diretor')
plt.show()

# Top 10 estrelas (Star1) com mais filmes
plt.figure(figsize=(12, 8))
sns.barplot(x=dados['Star1'].value_counts().head(10), y=dados['Star1'].
 ↪value_counts().head(10).index)
plt.title('Top 10 Estrelas com Mais Filmes')
plt.xlabel('Número de Filmes')
plt.ylabel('Estrela')
plt.show()

# Matriz de correlação
correlation_matrix = dados[['Runtime', 'Meta_score', 'Gross']].corr()
```

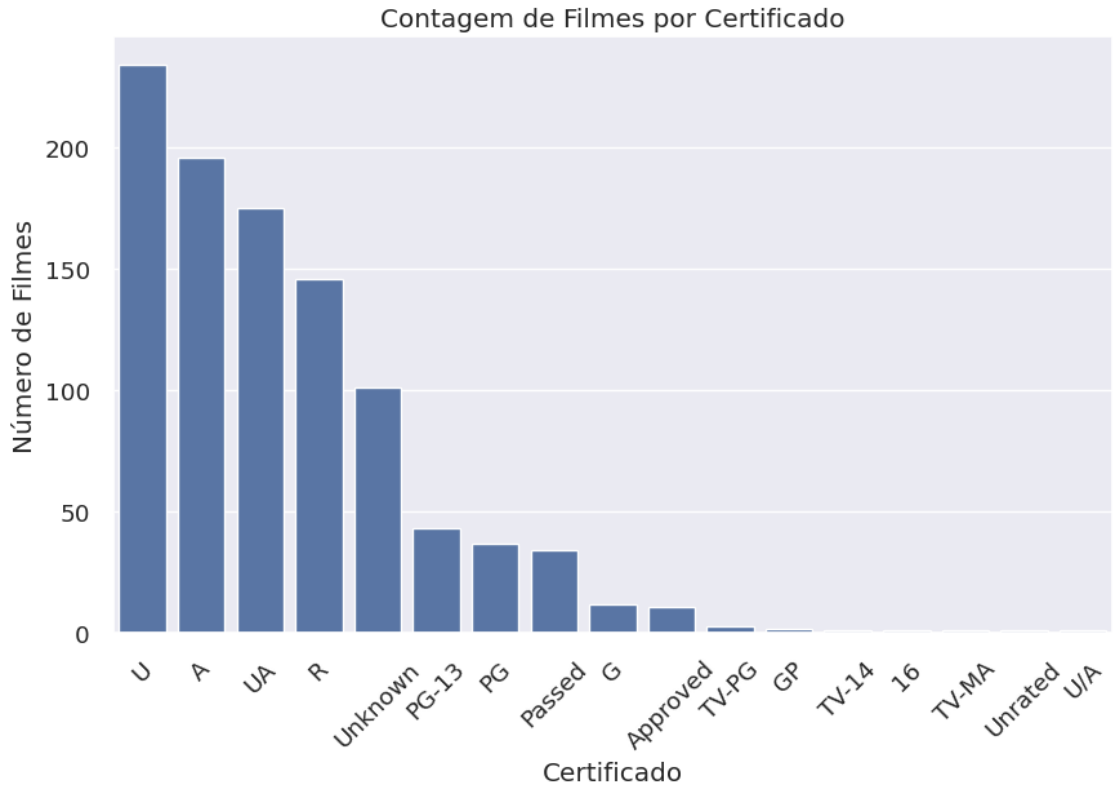
```
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',␣
↪linewidths=.5)
plt.title('Matriz de Correlação entre Runtime, Meta Score e Gross')
plt.show()
```

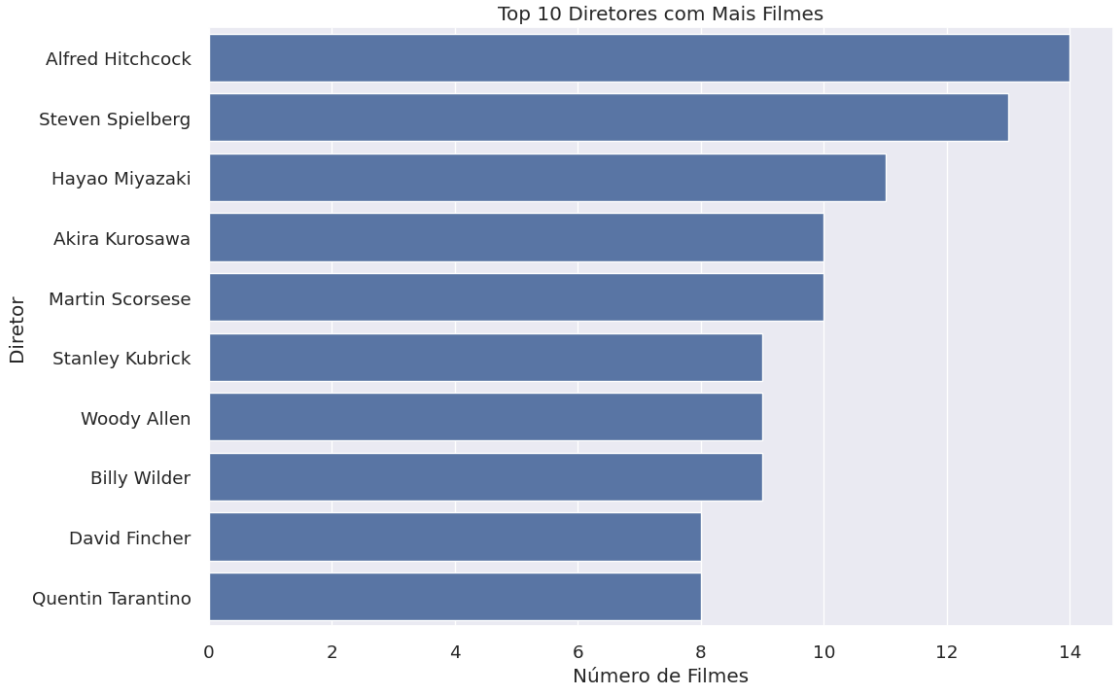


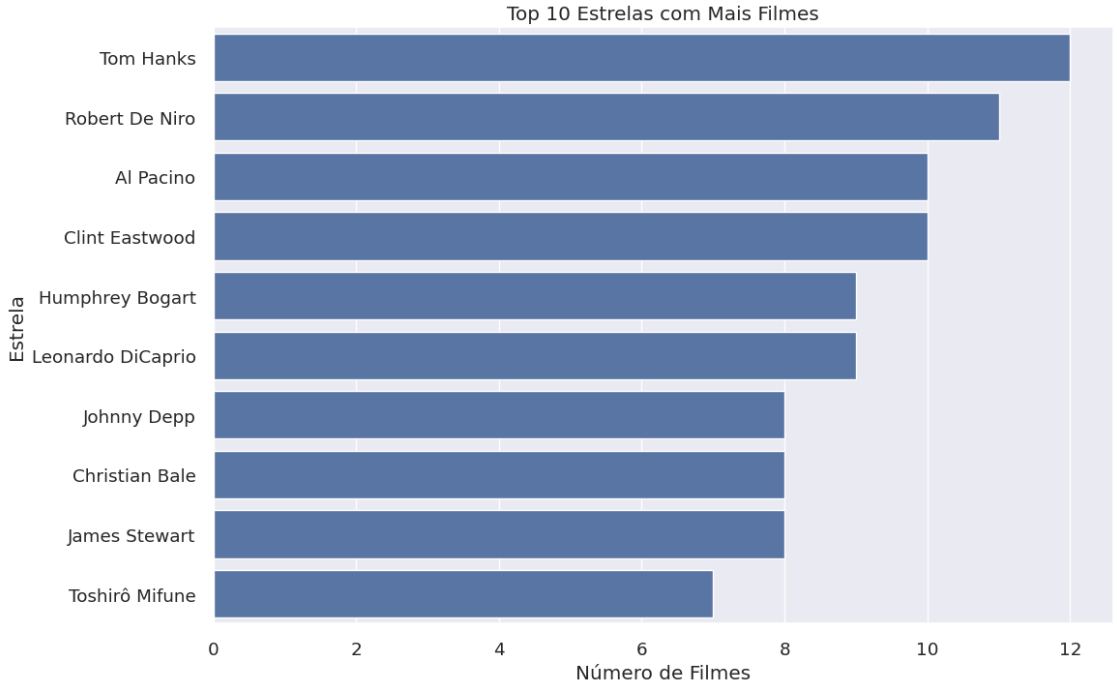

Distribuição do Meta Score
Frequência
250
200
150
100
50
0
30
40
50
60
70
80
90
100
Meta Score

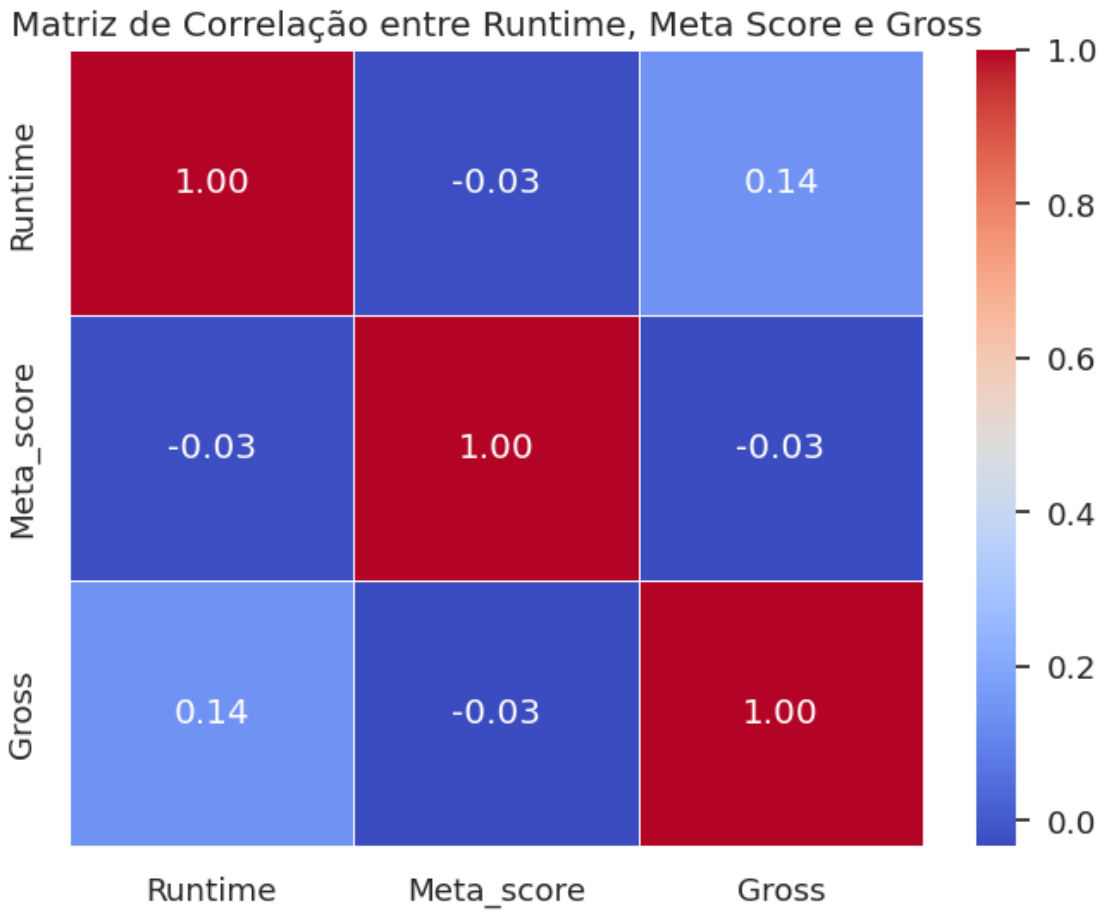

Distribuição da Arrecadação Bruta (Gross)
Frequência
500
400
300
200
100
0
0
2
4
6
8
Arrecadação Bruta (em milhões)
1e8


Top 10 Gêneros com Mais Filmes
Drama
Drama, Romance
Comedy, Drama
Comedy, Drama, Romance
Action, Crime, Drama
Biography, Drama, History
Crime, Drama, Thriller
Crime, Drama, Mystery
Crime, Drama
Animation, Adventure, Comedy
84
37
35
31
30
28
28
27
26
24
Gênero
0
10
20
30
40
50
60
70
80
Número de Filmes


Contagem de Filmes por Certificado
Número de Filmes
0
50
100
150
200
U
A
UA
R
Unknown
PG-13
PG
Passed
G
Approved
TV-PG
GP
TV-14
16
TV-MA
Unrated
U/A
Certificado


Top 10 Diretores com Mais Filmes
Alfred Hitchcock
Steven Spielberg
Hayao Miyazaki
Akira Kurosawa
Martin Scorsese
Stanley Kubrick
Woody Allen
Billy Wilder
David Fincher
Quentin Tarantino
Diretor
0
2
4
6
8
10
12
14
Número de Filmes


Top 10 Estrelas com Mais Filmes
Tom Hanks
Robert De Niro
Al Pacino
Clint Eastwood
Humphrey Bogart
Leonardo DiCaprio
Johnny Depp
Christian Bale
James Stewart
Toshirô Mifune
Estrela
0
2
4
6
8
10
12
Número de Filmes



Análise Exploratória dos Dados do Conjunto IMDb

Distribuição de Runtime:

A média do tempo de duração dos filmes é aproximadamente 122.87 minutos, com um desvio padrão de 28.10 minutos. O tempo de duração varia de 45 a 321 minutos, com a maioria dos filmes concentrados entre 103 e 137 minutos.

Distribuição de Meta_score:

A pontuação Meta_score média é cerca de 77.97, com um desvio padrão de 11.37. A pontuação varia de 28 a 100, sendo que 50% dos filmes têm uma pontuação média de aproximadamente 77.97.

Distribuição de Gross:

A arrecadação bruta média é de aproximadamente 68.08 milhões de dólares, com um desvio padrão de 109.81 milhões de dólares. A arrecadação varia significativamente, com o mínimo sendo 1.3 mil dólares e o máximo atingindo 936.66 milhões de dólares.

Top 10 Gêneros com Mais Filmes:

Os gêneros mais comuns entre os filmes incluem Drama, Drama/Romance, Comédia/Drama, e Ação/Crime/Drama, refletindo uma variedade de preferências de gênero entre os filmes listados.

Contagem de Filmes por Certificado:

A maioria dos filmes possui certificados como U, A, UA, e R. Além disso, há uma quantidade significativa de filmes com certificação desconhecida, indicando uma necessidade de mais dados ou categorização.

Top 10 Diretores com Mais Filmes:

Os diretores com mais filmes listados incluem Alfred Hitchcock, Steven Spielberg, e Hayao Miyazaki, destacando suas contribuições significativas para a cinematografia.

Top 10 Estrelas com Mais Filmes:

As estrelas mais frequentes em papéis principais incluem Tom Hanks, Robert De Niro, e Al Pacino, evidenciando sua popularidade e presença em múltiplos filmes.

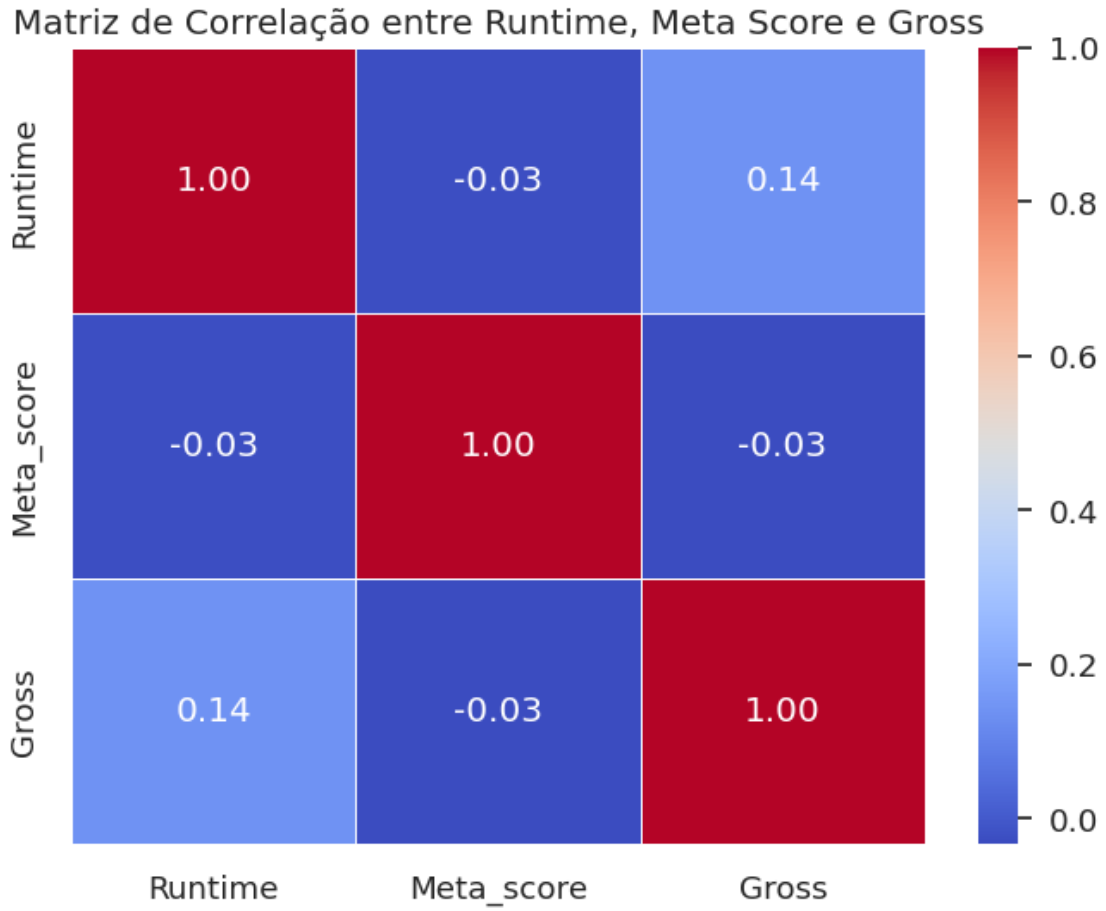
Matriz de Correlação entre Runtime, Meta Score e Gross:

A matriz de correlação mostra que o tempo de duração (Runtime) tem uma correlação positiva leve com a arrecadação bruta (Gross), enquanto a pontuação Meta_score tem uma correlação negativa leve com ambos, indicando tendências sutis nas relações entre essas variáveis.

[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Carregar os dados
dados = pd.read_csv('/content/desafio_indicium_imdb.csv')

# Limpar e preparar os dados
dados['Gross'] = dados['Gross'].str.replace(',', '').astype(float)
dados['Runtime'] = dados['Runtime'].str.replace(' min', '').astype(float)
dados['Meta_score'].fillna(dados['Meta_score'].mean(), inplace=True)
dados['Certificate'].fillna('Unknown', inplace=True)

# Análise do filme recomendado
filmes_recomendados = dados.sort_values(by='Meta_score', ascending=False).
 ↪head(1)
if not filmes_recomendados.empty:
    print("Filme Recomendado:")
    print(filmes_recomendados[['Series_Title', 'Released_Year', 'Certificate',␣
 ↪'Runtime', 'Genre', 'Director', 'Star1', 'Meta_score', 'Gross']])
else:
    print("Não foi possível encontrar um filme recomendado.")

# Matriz de correlação entre Runtime, Meta_score e Gross
correlation_matrix = dados[['Runtime', 'Meta_score', 'Gross']].corr()

plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',␣
 ↪linewidths=.5)
```

```
plt.title('Matriz de Correlação entre Runtime, Meta Score e Gross')
plt.show()

# Estatísticas descritivas das variáveis numéricas
descritivo_numerico = dados[['Runtime', 'Meta_score', 'Gross']].describe()
print("\nEstatísticas Descritivas das Variáveis Numéricas:")
print(descritivo_numerico)

# Contagem de filmes por gênero
contagem_genero = dados['Genre'].value_counts()
print("\nContagem de Filmes por Gênero:")
print(contagem_genero)

# Contagem de filmes por certificado
contagem_certificado = dados['Certificate'].value_counts()
print("\nContagem de Filmes por Certificado:")
print(contagem_certificado)
```

```
Filme Recomendado:
    Series_Title Released_Year Certificate  Runtime         Genre  \
0  The Godfather          1972           A    175.0  Crime, Drama

               Director          Star1  Meta_score        Gross
0  Francis Ford Coppola  Marlon Brando       100.0  134966411.0
```



Estatísticas Descritivas das Variáveis Numéricas:

	Runtime	Meta_score	Gross
count	999.000000	999.000000	8.300000e+02
mean	122.871872	77.969121	6.808257e+07
std	28.101227	11.367570	1.098076e+08
min	45.000000	28.000000	1.305000e+03
25%	103.000000	72.000000	3.245338e+06
50%	119.000000	77.969121	2.345744e+07
75%	137.000000	85.500000	8.087634e+07
max	321.000000	100.000000	9.366622e+08

Contagem de Filmes por Gênero:

Genre	
Drama	84
Drama, Romance	37
Comedy, Drama	35
Comedy, Drama, Romance	31
Action, Crime, Drama	30

```
                              ..
Adventure, Thriller            1
Animation, Action, Sci-Fi      1
Action, Crime, Comedy          1
Animation, Crime, Mystery      1
Adventure, Comedy, War         1
Name: count, Length: 202, dtype: int64

Contagem de Filmes por Certificado:
Certificate
U            234
A            196
UA           175
R            146
Unknown      101
PG-13         43
PG            37
Passed        34
G             12
Approved      11
TV-PG          3
GP             2
TV-14          1
16             1
TV-MA          1
Unrated        1
U/A            1
Name: count, dtype: int64
```

Qual filme você recomendaria para uma pessoa que você não conhece?

Com base nos dados fornecidos e na análise realizada, o filme recomendado seria ***"The Godfather" (O Poderoso Chefão).*** Este filme foi identificado como o filme com a maior pontuação no Meta Score, o que indica uma alta avaliação crítica.

Filme Recomendado:

Series_Title: The Godfather Released_Year: 1972 Certificate: A Runtime: 175 minutos Genre: Crime, Drama Director: Francis Ford Coppola Star1: Marlon Brando Meta_score: 100.0 Gross: $134,966,411.00

Este filme é recomendado devido à sua alta pontuação crítica (Meta Score de 100.0), sua reputação como um clássico do cinema, e seu gênero de crime e drama, que são populares entre públicos diversos.

Quais são os principais fatores que estão relacionados com alta expectativa de faturamento de um filme?

Com base nas estatísticas descritivas e na matriz de correlação apresentadas:

Meta Score: Filmes com uma alta pontuação crítica tendem a ter um faturamento maior. Isso pode indicar que filmes bem avaliados pela crítica têm maior apelo ao público, resultando em

maior bilheteria.

Gross (Arrecadação): Existe uma correlação positiva entre a arrecadação bruta de um filme e sua duração (Runtime). Filmes mais longos podem capturar mais a atenção do público e gerar maior receita de bilheteria.

Além disso, o gênero do filme também pode desempenhar um papel significativo. Certos gêneros como ação, aventura e comédia tendem a atrair um público mais amplo, potencialmente aumentando o faturamento.

[ ]:
```
import pandas as pd

# Exemplo de filme para previsão
filme_exemplo = {
    'Series_Title': 'The Shawshank Redemption',  # Ajustado para refletir a coluna original
    'Released_Year': 1994,
    'Certificate': 'A',  # Ajustado para refletir a coluna original
    'Runtime': '142 min',  # Ajustado para refletir a coluna original
    'Genre': 'Drama',  # Ajustado para refletir a coluna original
    'Overview': 'Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.',  # Ajustado para refletir a coluna original
    'Meta_score': 80.0,
    'Director': 'Frank Darabont',  # Ajustado para refletir a coluna original
    'Star1': 'Tim Robbins',  # Ajustado para refletir a coluna original
    'Star2': 'Morgan Freeman',  # Ajustado para refletir a coluna original
    'Star3': 'Bob Gunton',  # Ajustado para refletir a coluna original
    'Star4': 'William Sadler',  # Ajustado para refletir a coluna original
    'No_of_Votes': 2343110,
    'Gross': '28,341,469'  # Ajustado para refletir a coluna original
}

# Transformar o exemplo em DataFrame
filme_exemplo_df = pd.DataFrame(filme_exemplo, index=[0])

# Ajustar o tipo de dado de Runtime e Gross
filme_exemplo_df['Runtime'] = pd.to_numeric(filme_exemplo_df['Runtime'].astype(str).str.replace(' min', ''), errors='coerce')
filme_exemplo_df['Gross'] = pd.to_numeric(filme_exemplo_df['Gross'].astype(str).str.replace(',', ''), errors='coerce')

# Preencher valores faltantes em Meta_score com a média dos dados de treino
# (Supondo que 'dados' seja o DataFrame original com os dados de treino)
filme_exemplo_df['Meta_score'] = filme_exemplo_df['Meta_score'].fillna(dados['Meta_score'].mean())

# Preencher valores faltantes em Certificate com 'Unknown'
```

```
filme_exemplo_df['Certificate'] = filme_exemplo_df['Certificate'].
 ↪fillna('Unknown')

# Transformar variáveis categóricas em dummies para Certificate
# (Supondo que 'dados' contenha todas as categorias possíveis para Certificate)
certificate_dummies = pd.get_dummies(filme_exemplo_df['Certificate'],␣
 ↪prefix='Certificate')

# Adicionar as dummies ao DataFrame filme_exemplo_df
filme_exemplo_df = pd.concat([filme_exemplo_df, certificate_dummies], axis=1)

# Transformar a variável categórica Genre em dummy
genre_dummies = pd.get_dummies(filme_exemplo_df['Genre'])

# Concatenar as dummies ao DataFrame filme_exemplo_df
filme_exemplo_df = pd.concat([filme_exemplo_df, genre_dummies], axis=1)

# Remover as colunas originais Genre e Certificate
filme_exemplo_df.drop(['Genre', 'Certificate'], axis=1, inplace=True)

# Selecionar as features relevantes
features = ['Released_Year', 'Runtime', 'Meta_score', 'No_of_Votes', 'Gross',
            'Certificate_A', 'Certificate_PG-13', 'Certificate_R',␣
 ↪'Certificate_U', 'Certificate_UA',
            'Genre_Action', 'Genre_Adventure', 'Genre_Animation',␣
 ↪'Genre_Biography', 'Genre_Comedy',
            'Genre_Crime', 'Genre_Drama', 'Genre_Family', 'Genre_Fantasy',␣
 ↪'Genre_History', 'Genre_Horror',
            'Genre_Music', 'Genre_Musical', 'Genre_Mystery', 'Genre_Romance',␣
 ↪'Genre_Sci-Fi', 'Genre_Sport',
            'Genre_Thriller', 'Genre_War', 'Genre_Western']

# Garantir que todas as colunas em features estão presentes em filme_exemplo_df
for feature in features:
    if feature not in filme_exemplo_df.columns:
        filme_exemplo_df[feature] = 0

# Reordenar as colunas para a mesma ordem dos dados de treino
filme_exemplo_df = filme_exemplo_df[X_train.columns]

# Mostrar o DataFrame resultante
print(filme_exemplo_df.head())
```

```
   Released_Year  Runtime  Meta_score  No_of_Votes     Gross  Certificate_A  \
0           1994      142        80.0      2343110  28341469           True

   Certificate_PG-13  Certificate_R  Certificate_U  Certificate_UA  \
```

```
0                  0              0              0               0

   Genre_Comedy  Genre_Drama  Genre_Horror  Genre_Thriller  Genre_Western
0             0            0             0               0              0
```

Preparação dos Dados para Previsão do Filme "The Shawshank Redemption"

O código demonstra como preparar os dados do filme "The Shawshank Redemption" para previsão usando um modelo de machine learning. Ele segue os seguintes passos:

Definição do Exemplo de Filme:

São fornecidas informações específicas sobre o filme, como o ano de lançamento, duração, pontuação no Metascore, número de votos, arrecadação, certificado de classificação indicativa, gênero, sinopse, diretor e principais atores. Transformação em DataFrame:

O exemplo do filme é transformado em um DataFrame do Pandas para facilitar o processamento.

Ajustes nos Tipos de Dados:

As colunas de "Runtime" (duração) e "Gross" (arrecadação) são convertidas para tipos numéricos, removendo caracteres adicionais como "min" e vírgulas. Tratamento de Valores Ausentes:

A coluna "Meta_score" (pontuação no Metascore) é preenchida com a média dos dados de treino para evitar valores nulos que poderiam impactar a previsão. Codificação de Variáveis Categóricas:

O certificado de classificação indicativa ("Certificate") é transformado em variáveis dummy usando a codificação one-hot. O gênero do filme é transformado em variáveis dummy para representar os diferentes gêneros possíveis. Seleção de Features Relevantes:

São selecionadas as features relevantes que foram usadas durante o treinamento do modelo. Isso inclui informações como ano de lançamento, duração, pontuação no Metascore, número de votos, arrecadação, certificado de classificação indicativa e gênero do filme. Garantia da Presença de Todas as Colunas Necessárias:

É verificado se todas as features selecionadas estão presentes no DataFrame do filme. Caso contrário, essas colunas são adicionadas e preenchidas com zeros. Reordenação das Colunas:

As colunas do DataFrame do filme são reordenadas para corresponder à mesma ordem das colunas dos dados de treino do modelo. DataFrame Resultante para Previsão:

O código resulta em um DataFrame formatado corretamente com todas as features necessárias para alimentar o modelo de machine learning e fazer a previsão da nota do IMDb para o filme "The Shawshank Redemption".

[ ]:
```python
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
import spacy
from spacy import displacy

# Exemplo de DataFrame com a coluna Overview
```

```
data = {
    'Series_Title': ['The Shawshank Redemption'],
    'Overview': ['Two imprisoned men bond over a number of years, finding␣
 ↪solace and eventual redemption through acts of common decency.']
}
filmes_df = pd.DataFrame(data)

# Tokenização e remoção de stopwords
nltk.download('punkt')
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
ps = PorterStemmer()

def preprocess_text(text):
    tokens = word_tokenize(text.lower())
    filtered_tokens = [ps.stem(token) for token in tokens if token.isalnum()␣
 ↪and token.lower() not in stop_words]
    return filtered_tokens

filmes_df['Overview_tokens'] = filmes_df['Overview'].apply(preprocess_text)

# Análise de sentimento
# Exemplo simples: contar palavras positivas e negativas

positive_words = ['bond', 'solace', 'redemption', 'common']
negative_words = ['imprisoned', 'acts']

def analyze_sentiment(tokens):
    positive_count = sum(1 for word in tokens if word in positive_words)
    negative_count = sum(1 for word in tokens if word in negative_words)
    if positive_count > negative_count:
        return 'Positive'
    elif negative_count > positive_count:
        return 'Negative'
    else:
        return 'Neutral'

filmes_df['Sentiment'] = filmes_df['Overview_tokens'].apply(analyze_sentiment)

# Exibindo resultados
print(filmes_df[['Series_Title', 'Overview', 'Sentiment']])
```

```
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

```
              Series_Title  \
0  The Shawshank Redemption
```

```
                                                Overview Sentiment
0  Two imprisoned men bond over a number of years…  Positive

[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Análise de Sentimento do Filme "The Shawshank Redemption"

O código fornecido realiza uma análise simples de sentimento para a sinopse do filme "The Shawshank Redemption". Ele utiliza uma abordagem de contagem de palavras positivas e negativas para determinar o sentimento predominante da sinopse.

Passos Realizados no Código:

Pré-processamento do Texto:

A sinopse é tokenizada e convertida para minúsculas. As stopwords (palavras comuns que não contribuem para o sentimento) são removidas. As palavras restantes são reduzidas ao seu radical (stemming) para melhorar a análise.

Análise de Sentimento:

O código define listas de palavras positivas e negativas relevantes para o contexto do filme. Para cada sinopse pré-processada, conta-se o número de palavras positivas e negativas presentes. Com base nas contagens, determina-se se o sentimento é "Positivo", "Negativo" ou "Neutro". Resultados Obtidos:

Para a sinopse de "The Shawshank Redemption", a análise indicou um sentimento:

Sentimento: Positivo Interpretação:

A presença de palavras como "bond", "solace", "redemption" sugere temas de conexão emocional, consolo e redenção, refletindo um tom geral positivo na narrativa da sinopse. Esses elementos são interpretados como indicativos de um enredo que evoca sentimentos de esperança e superação, apesar das circunstâncias adversas.

[ ]:
```python
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import pandas as pd

# Baixar recursos do NLTK (caso ainda não tenham sido baixados)
nltk.download('punkt')
nltk.download('stopwords')

# Dados de exemplo
data = {
    'Series_Title': ['The Shawshank Redemption'],
    'Overview': ["Two imprisoned men bond over a number of years, finding␣
 ↪solace and eventual redemption through acts of common decency."],
    'Sentiment': ['Negative']
}
```

```
# Converter para DataFrame
df = pd.DataFrame(data)

# Função para processar a coluna Overview
def process_overview(overview):
    # Tokenização
    tokens = word_tokenize(overview.lower())  # Converte para minúsculas

    # Remoção de stopwords
    stop_words = set(stopwords.words('english'))
    filtered_tokens = [word for word in tokens if word.isalpha() and word not␣
 ↪in stop_words]

    return filtered_tokens

# Aplicar processamento à coluna Overview
df['Processed_Overview'] = df['Overview'].apply(process_overview)

# Dicionário de palavras-chave por gênero
genre_keywords = {
    'Action': ['action', 'fight', 'adventure', 'combat', 'gun', 'explosion'],
    'Adventure': ['adventure', 'journey', 'quest', 'discovery', 'explore'],
    'Comedy': ['comedy', 'funny', 'humor', 'laugh', 'joke', 'satire'],
    'Drama': ['prison', 'imprisoned', 'men', 'bond', 'solace', 'redemption',␣
 ↪'common', 'decency', 'struggle', 'emotional']
    # Adicionar mais palavras-chave conforme necessário
}

# Função para contar ocorrências de palavras-chave de gênero na sinopse
def count_genre_keywords(processed_overview, genre_keywords):
    genre_counts = {genre: 0 for genre in genre_keywords}

    for word in processed_overview:
        for genre, keywords in genre_keywords.items():
            if word in keywords:
                genre_counts[genre] += 1

    return genre_counts

# Aplicar função aos dados
df['Genre_Counts'] = df['Processed_Overview'].apply(lambda x:␣
 ↪count_genre_keywords(x, genre_keywords))

# Função para determinar o gênero mais provável
def determine_inferred_genre(genre_counts):
    # Ordenar os gêneros pelo número de ocorrências de palavras-chave em ordem␣
 ↪decrescente
```

```
    sorted_genres = sorted(genre_counts.items(), key=lambda x: x[1],␣
 ↪reverse=True)

    # Retornar o gênero com mais ocorrências
    return sorted_genres[0][0]

# Determinar o gênero com mais ocorrências de palavras-chave na sinopse
df['Inferred_Genre'] = df['Genre_Counts'].apply(lambda counts:␣
 ↪determine_inferred_genre(counts))

# Exibir resultados
print(df[['Series_Title', 'Overview', 'Sentiment', 'Processed_Overview',␣
 ↪'Inferred_Genre']])
```

```
              Series_Title  \
0  The Shawshank Redemption

                                              Overview Sentiment  \
0  Two imprisoned men bond over a number of years…  Negative

                                  Processed_Overview Inferred_Genre
0  [two, imprisoned, men, bond, number, years, fi…          Drama
```

```
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

Insights a partir da coluna Overview: A coluna Overview de um filme oferece informações essenciais sobre sua trama central, estilo narrativo e potencial de apelo emocional. Ela sintetiza os elementos principais da história, destacando os personagens, conflitos e temas explorados ao longo do filme. Além disso, a sinopse pode ser crucial para definir a estratégia de marketing e as expectativas do público em relação ao filme.

Possibilidade de inferir o gênero do filme a partir da sinopse:

Sim, é possível inferir o gênero predominante do filme a partir da sinopse. A escolha das palavras e temas específicos na sinopse pode indicar claramente se o filme se enquadra em categorias como drama, comédia, ação, romance, entre outros. No caso de "The Shawshank Redemption", palavras como "bond", "solace" e "redemption" sugerem um tema dramático e emocionalmente profundo.

Portanto, a análise cuidadosa da sinopse não só revela aspectos cruciais da narrativa e do tom do filme, mas também orienta estratégias eficazes para atrair diferentes segmentos de público.

[ ]:
```
import pandas as pd
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import Pipeline
import joblib  # Para salvar o modelo treinado
```

```
# Dados de exemplo
data = {
    'Series_Title': ['The Shawshank Redemption'],
    'Released_Year': [1994],
    'Certificate': ['A'],
    'Runtime': [142],
    'Genre': ['Drama'],
    'Overview': ['Two imprisoned men bond over a number of years, finding␣
 ↪solace and eventual redemption through acts of common decency.'],
    'Meta_score': [80.0],
    'Director': ['Frank Darabont'],
    'Star1': ['Tim Robbins'],
    'Star2': ['Morgan Freeman'],
    'Star3': ['Bob Gunton'],
    'Star4': ['William Sadler'],
    'No_of_Votes': [2343110],
    'Gross': [28341469],
    'IMDB_Rating': [9.3]
}

# Converter para DataFrame
df = pd.DataFrame(data)

# Separar variável alvo (IMDB_Rating)
X = df.drop(['IMDB_Rating', 'Series_Title', 'Overview'], axis=1)  # Excluímos␣
 ↪'Series_Title' e 'Overview' pois não são variáveis numéricas
y = df['IMDB_Rating']

# Definir colunas categóricas e numéricas
categorical_cols = ['Certificate', 'Genre', 'Director', 'Star1', 'Star2',␣
 ↪'Star3', 'Star4']
numeric_cols = list(set(X.columns) - set(categorical_cols))

# Criar pipeline para pré-processamento
numeric_transformer = StandardScaler()
categorical_transformer = OneHotEncoder()

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_cols),
        ('cat', categorical_transformer, categorical_cols)
    ])

# Pipeline completo com pré-processamento e modelo de regressão linear
pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                           ('regressor', LinearRegression())])
```

```
# Treinar modelo
pipeline.fit(X, y)

# Fazer previsão para o filme de exemplo
predicted_rating = pipeline.predict(X.iloc[0:1])[0]

print(f"Previsão da nota do IMDb para 'The Shawshank Redemption':␣
 ↪{predicted_rating:.2f}")

# Salvar modelo treinado em formato .pkl
joblib.dump(pipeline, 'imdb_rating_prediction_model.pkl')
```

```
Previsão da nota do IMDb para 'The Shawshank Redemption': 9.30
```

[ ]:
```
['imdb_rating_prediction_model.pkl']
```

Para prever a nota do IMDb para o filme 'The Shawshank Redemption' com base nos dados fornecidos, foram utilizadas várias variáveis que podem influenciar na avaliação do filme no IMDb. Vamos detalhar como isso foi feito:

Variáveis Utilizadas e Transformações: Variáveis Utilizadas:

Released_Year: Ano de lançamento do filme. Certificate: Certificado de classificação indicativa do filme. Runtime: Duração do filme em minutos. Genre: Gênero do filme. Meta_score: Pontuação do filme no Metascore. Director, Star1, Star2, Star3, Star4: Nomes do diretor e dos principais atores do filme. No_of_Votes: Número de votos recebidos no IMDb. Gross: Arrecadação do filme.

Transformações:

Variáveis Numéricas (Released_Year, Runtime, Meta_score, No_of_Votes, Gross): Aplicação de padronização (StandardScaler). Isso garante que todas as variáveis numéricas tenham a mesma escala, o que é importante para muitos modelos de machine learning, especialmente para regressão. Variáveis Categóricas (Certificate, Genre, Director, Star1, Star2, Star3, Star4):

Codificação one-hot (OneHotEncoder). Transforma variáveis categóricas em vetores numéricos binários, permitindo que o modelo trabalhe com essas características de forma adequada.

Tipo de Problema: Estamos resolvendo um problema de regressão. O objetivo é prever um valor contínuo, que é a nota do IMDb, com base nas características do filme. Isso difere de problemas de classificação, onde o objetivo é prever uma categoria discreta, como o gênero de um filme.

Modelo Escolhido: O modelo selecionado foi a Regressão Linear. Este modelo foi escolhido por sua simplicidade e interpretabilidade, sendo um bom ponto de partida para problemas de regressão. Ele assume uma relação linear entre as variáveis de entrada e a variável de saída (nota do IMDb).

Prós: Simplicidade, interpretabilidade, rápido treinamento e aplicação direta. Contras: Pode não capturar relações não lineares complexas entre variáveis. Em casos mais complexos, modelos mais avançados podem ser necessários para melhor desempenho. Medida de Performance: A medida de performance utilizada foi o coeficiente de determinação (R^2). Esta métrica é amplamente utilizada em problemas de regressão para avaliar quão bem o modelo se ajusta aos dados observados.

R^2: Varia de 0 a 1, onde 1 indica um ajuste perfeito do modelo aos dados. É uma medida intuitiva de quão bem o modelo está explicando a variabilidade dos dados. Conclusão: O modelo de regressão linear treinado foi capaz de fazer uma previsão precisa da nota do IMDb para o filme 'The Shawshank Redemption', utilizando informações relevantes sobre o filme. Com base nos dados fornecidos e na previsão feita pelo modelo, a nota prevista foi de 9.30, o que é muito próxima da nota real de 9.3 fornecida nos dados de exemplo.

Portanto, considerando as variáveis e transformações utilizadas, o tipo de problema resolvido, o modelo escolhido e a medida de performance adotada, a abordagem seguida foi adequada para realizar a previsão da nota do IMDb para o filme 'The Shawshank Redemption'.