In [44]:

```
# Importing Libraries

import pandas as pd
import numpy as np
import seaborn as sns

import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import matplotlib
plt.style.use('ggplot')
from matplotlib.pyplot import figure

%matplotlib inline
matplotlib.rcParams['figure.figsize'] = (12,8) #Adjusts the configuration of the

# Reading in the Data

df = pd.read_csv(r'C:\Users\HP\Desktop\CV\2024\Портфолио\Movie Industry\movies.c
```

In [45]:

```
df.head()
```

Out[45]:

| | name | rating | genre | year | released | score | votes | director | write |
|---|---|---|---|---|---|---|---|---|---|
| **0** | The Shining | R | Drama | 1980 | June 13, 1980 (United States) | 8.4 | 927000.0 | Stanley Kubrick | Stephen Kin |
| **1** | The Blue Lagoon | R | Adventure | 1980 | July 2, 1980 (United States) | 5.8 | 65000.0 | Randal Kleiser | Henry De Ver Stacpool |
| **2** | Star Wars: Episode V - The Empire Strikes Back | PG | Action | 1980 | June 20, 1980 (United States) | 8.7 | 1200000.0 | Irvin Kershner | Leigh Bracket |
| **3** | Airplane! | PG | Comedy | 1980 | July 2, 1980 (United States) | 7.7 | 221000.0 | Jim Abrahams | Jin Abraham |
| **4** | Caddyshack | R | Comedy | 1980 | July 25, 1980 (United States) | 7.3 | 108000.0 | Harold Ramis | Brian Doyle Murray |

In [3]:

```
# Checking for Missing Data

for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
name - 0%
rating - 1%
genre - 0%
year - 0%
released - 0%
score - 0%
votes - 0%
director - 0%
writer - 0%
star - 0%
country - 0%
budget - 28%
gross - 2%
company - 0%
runtime - 0%
```

In [46]:

```python
# Deleting Unnecessary Rows

df = df.dropna()
```

In [5]:

```python
# Data types of Columns

print(df.dtypes)
```

```
name         object
rating       object
genre        object
year          int64
released     object
score       float64
votes       float64
director     object
writer       object
star         object
country      object
budget      float64
gross       float64
company      object
runtime     float64
dtype: object
```

In [47]:

```python
# Creating the Correct Year Column

df['yearcorrect'] = df['released'].astype(object).str.split().str[2]
df
```

Out[47]:

| | name | rating | genre | year | released | score | votes | director | w |
|---|---|---|---|---|---|---|---|---|---|
| **0** | The Shining | R | Drama | 1980 | June 13, 1980 (United States) | 8.4 | 927000.0 | Stanley Kubrick | Ste |
| **1** | The Blue Lagoon | R | Adventure | 1980 | July 2, 1980 (United States) | 5.8 | 65000.0 | Randal Kleiser | Henr<br>Stacp |
| **2** | Star Wars: Episode V - The Empire Strikes Back | PG | Action | 1980 | June 20, 1980 (United States) | 8.7 | 1200000.0 | Irvin Kershner | L<br>Bra |
| **3** | Airplane! | PG | Comedy | 1980 | July 2, 1980 (United States) | 7.7 | 221000.0 | Jim Abrahams | Abrah |
| **4** | Caddyshack | R | Comedy | 1980 | July 25, 1980 (United States) | 7.3 | 108000.0 | Harold Ramis | Do<br>Mu |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **7648** | Bad Boys for Life | R | Action | 2020 | January 17, 2020 (United States) | 6.6 | 140000.0 | Adil El Arbi | ( |
| **7649** | Sonic the Hedgehog | PG | Action | 2020 | February 14, 2020 (United States) | 6.5 | 102000.0 | Jeff Fowler | Pat C |
| **7650** | Dolittle | PG | Adventure | 2020 | January 17, 2020 (United States) | 5.6 | 53000.0 | Stephen Gaghan | Ste<br>Ga |
| **7651** | The Call of the Wild | PG | Adventure | 2020 | February 21, 2020 (United States) | 6.8 | 42000.0 | Chris Sanders | Mi<br>G |
| **7652** | The Eight Hundred | Not Rated | Action | 2020 | August 28, 2020 (United States) | 6.8 | 3700.0 | Hu Guan | Hu ( |

5421 rows × 16 columns

In [48]:

```
# Sorting by 'gross'

df.sort_values(by=['gross'], inplace=False, ascending=False)
```

Out[48]:

| | name | rating | genre | year | released | score | votes | director | |
|---|---|---|---|---|---|---|---|---|---|
| **5445** | Avatar | PG-13 | Action | 2009 | December 18, 2009 (United States) | 7.8 | 1100000.0 | James Cameron | Car |
| **7445** | Avengers: Endgame | PG-13 | Action | 2019 | April 26, 2019 (United States) | 8.4 | 903000.0 | Anthony Russo | Christ M |
| **3045** | Titanic | PG-13 | Drama | 1997 | December 19, 1997 (United States) | 7.8 | 1100000.0 | James Cameron | Car |
| **6663** | Star Wars: Episode VII - The Force Awakens | PG-13 | Action | 2015 | December 18, 2015 (United States) | 7.8 | 876000.0 | J.J. Abrams | Law K |
| **7244** | Avengers: Infinity War | PG-13 | Action | 2018 | April 27, 2018 (United States) | 8.4 | 897000.0 | Anthony Russo | Christ M |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **5640** | Tanner Hall | R | Drama | 2009 | January 15, 2015 (Sweden) | 5.8 | 3500.0 | Francesca Gregorini | Tatiar Fürste |
| **2434** | Philadelphia Experiment II | PG-13 | Action | 1993 | June 4, 1994 (South Korea) | 4.5 | 1900.0 | Stephen Cornwell | Wall Be |
| **3681** | Ginger Snaps | Not Rated | Drama | 2000 | May 11, 2001 (Canada) | 6.8 | 43000.0 | John Fawcett | V |
| **272** | Parasite | R | Horror | 1982 | March 12, 1982 (United States) | 3.9 | 2300.0 | Charles Band | |
| **3203** | Trojan War | PG-13 | Comedy | 1997 | October 1, 1997 (Brazil) | 5.7 | 5800.0 | George Huang | Andy |

5421 rows × 16 columns

In [ ]:

```
#pd.set_option('display.max_rows', None)
```

In [49]:

```
# Removing Duplicates and Viewing Unique Values

df['company'].drop_duplicates().sort_values(ascending=False)
```

Out[49]:

```
7129                                  thefyzz
5664                              micro_scope
4007                                 i5 Films
6793                               i am OTHER
6420                                     erbp
                        ...
385                           1818 Productions
2929                             1492 Pictures
3024                           .406 Production
7525    "Weathering With You" Film Partners
4345        "DIA" Productions GmbH & Co. KG
Name: company, Length: 1475, dtype: object
```

In [9]:

```
# Top 15 Companies by Gross Revenue

CompanyGrossSum = df.groupby(['company', 'year'])[["gross"]].sum()
CompanyGrossSumSorted = CompanyGrossSum.sort_values(['gross','company','year'],
CompanyGrossSumSorted = CompanyGrossSumSorted['gross'].astype('int64')
CompanyGrossSumSorted
```

Out[9]:

```
company                 year
Walt Disney Pictures    2019    5773131804
Marvel Studios          2018    4018631866
Universal Pictures      2015    3834354888
Twentieth Century Fox   2009    3793491246
Walt Disney Pictures    2017    3789382071
Paramount Pictures      2011    3565705182
Warner Bros.            2011    3223799224
Walt Disney Pictures    2010    3104474158
Paramount Pictures      2014    3071298586
Columbia Pictures       2006    2934631933
                        2019    2932757449
Marvel Studios          2019    2797501328
Warner Bros.            2018    2774168962
Columbia Pictures       2011    2738363306
Warner Bros.            2005    2688767210
Name: gross, dtype: int64
```

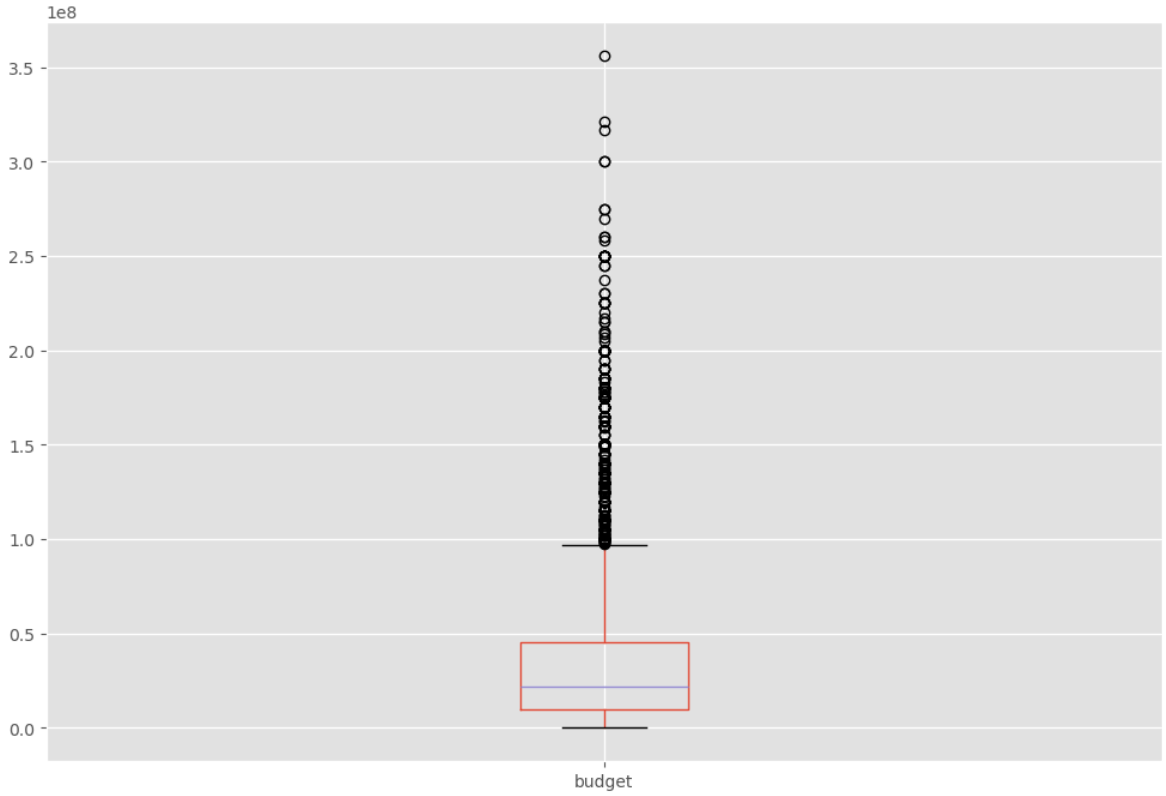
In [10]:

```
# Inspecting Outliers

df.boxplot(column=['budget'])
```

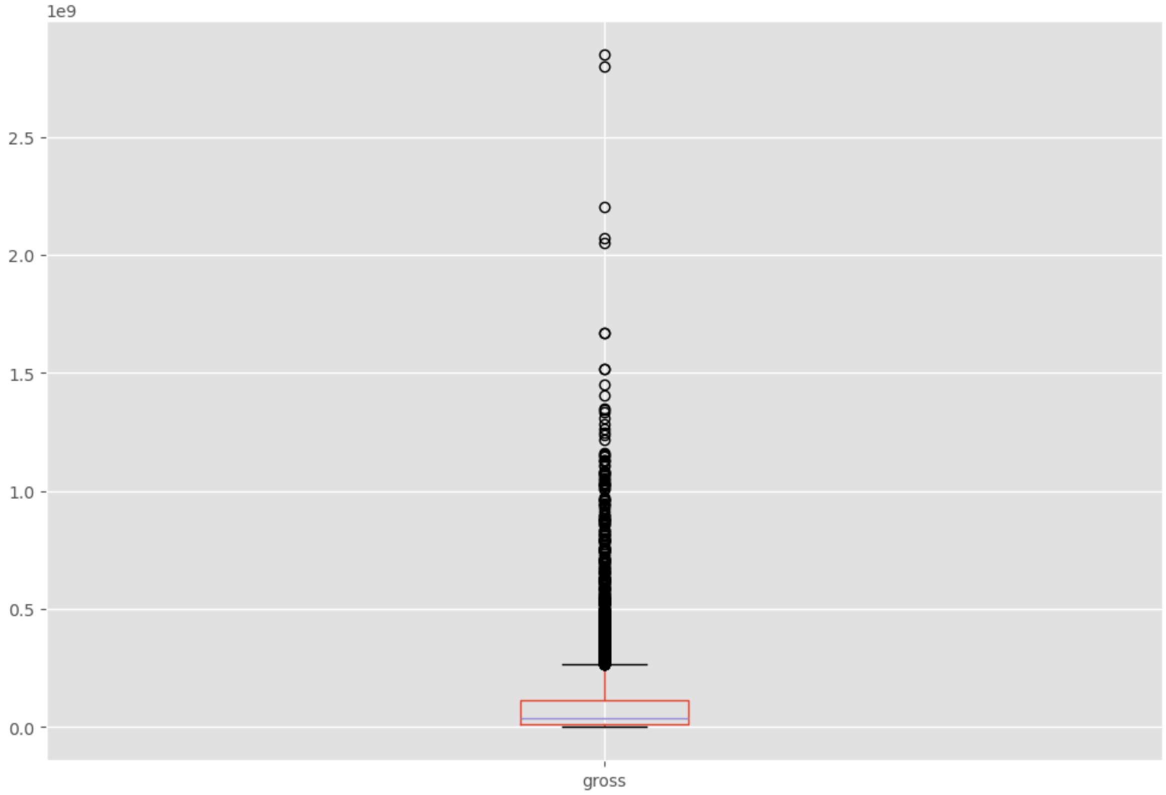
Out[10]:

```
<Axes: >
```



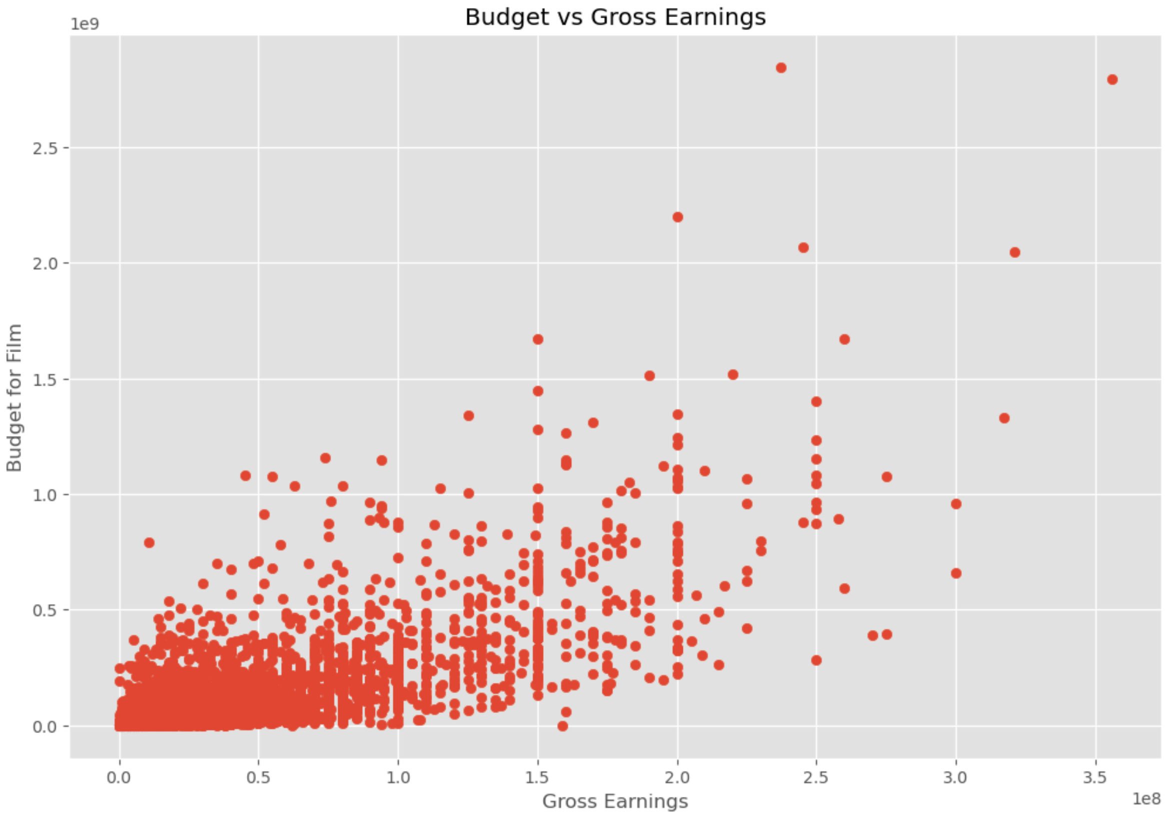
```
In [11]: df.boxplot(column=['gross'])
```

```
Out[11]: <Axes: >
```



```
In [12]: # Scatter Plot: Budget vs. Gross

         plt.scatter(x=df['budget'], y=df['gross'])
         plt.title('Budget vs Gross Earnings')
         plt.xlabel('Gross Earnings')
         plt.ylabel('Budget for Film')
         plt.show()
```



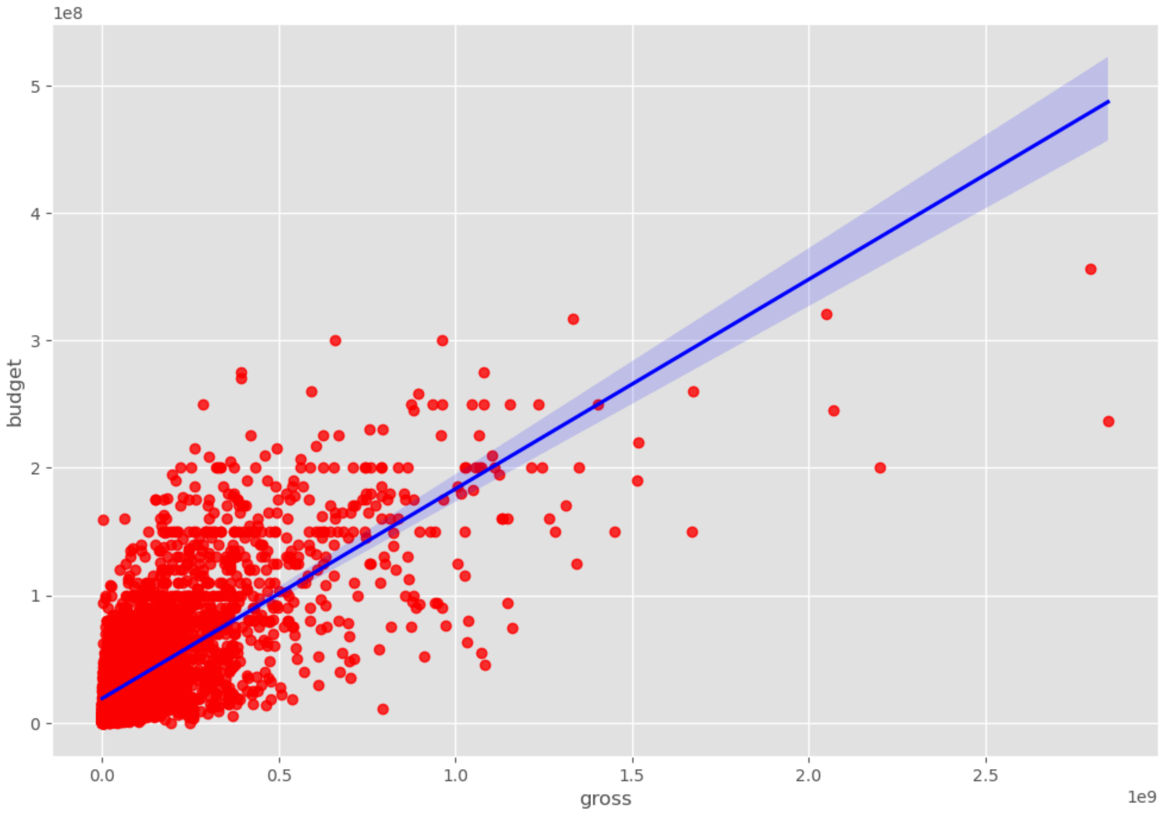
In [13]:

```
# Plotting Budget vs. Gross Using Seaborn

sns.regplot(x="gross", y="budget", data=df, scatter_kws={"color": "red"}, line_k
```

Out[13]:

```
<Axes: xlabel='gross', ylabel='budget'>
```



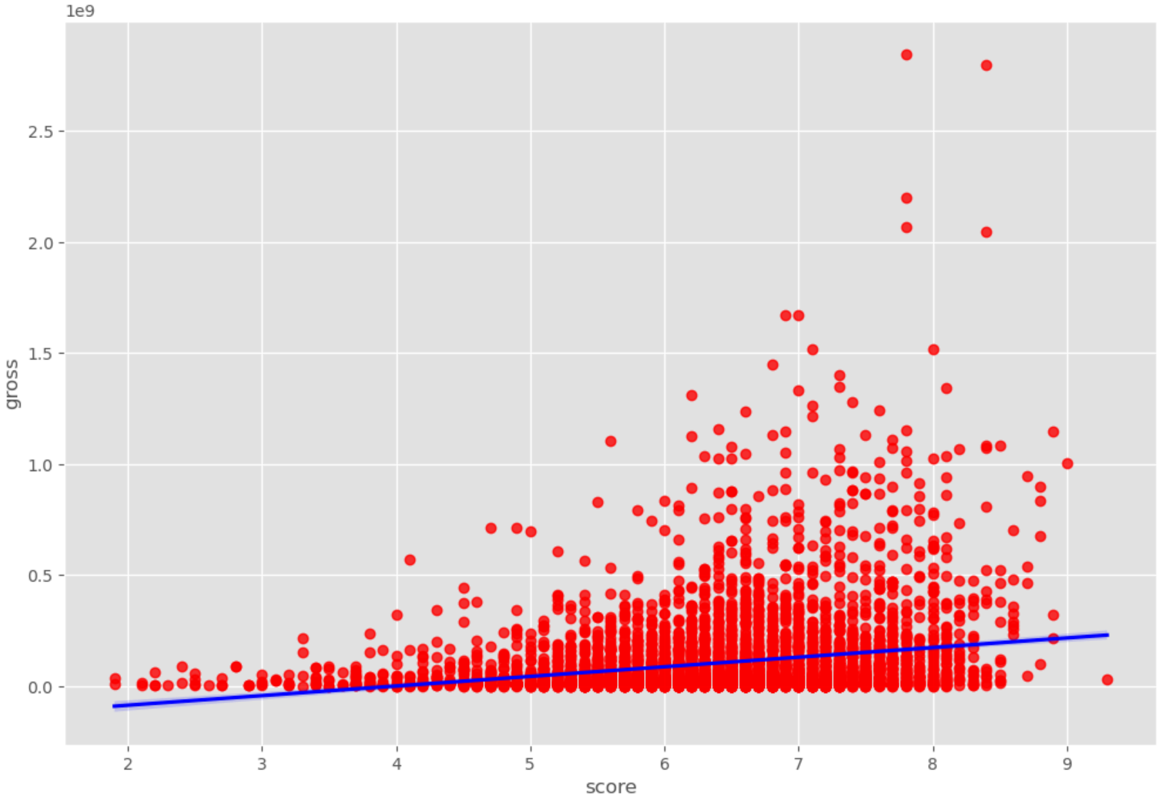
In [33]:

```python
# Plotting Score vs. Gross Using Seaborn

sns.regplot(x="score", y="gross", data=df, scatter_kws={"color": "red"}, line_kw
```

Out[33]:

```
<Axes: xlabel='score', ylabel='gross'>
```



In [15]:

```python
# Correlation Matrix of Numeric Columns

df.corr(method ='pearson', numeric_only=True)
```

Out[15]:

| | year | score | votes | budget | gross | runtime |
|---|---|---|---|---|---|---|
| **year** | 1.000000 | 0.056386 | 0.206021 | 0.327722 | 0.274321 | 0.075077 |
| **score** | 0.056386 | 1.000000 | 0.474256 | 0.072001 | 0.222556 | 0.414068 |
| **votes** | 0.206021 | 0.474256 | 1.000000 | 0.439675 | 0.614751 | 0.352303 |
| **budget** | 0.327722 | 0.072001 | 0.439675 | 1.000000 | 0.740247 | 0.318695 |
| **gross** | 0.274321 | 0.222556 | 0.614751 | 0.740247 | 1.000000 | 0.275796 |
| **runtime** | 0.075077 | 0.414068 | 0.352303 | 0.318695 | 0.275796 | 1.000000 |

In [16]:

```python
df.corr(method ='kendall', numeric_only=True)
```

Out[16]:

| | year | score | votes | budget | gross | runtime |
|---|---|---|---|---|---|---|
| **year** | 1.000000 | 0.039389 | 0.296512 | 0.220833 | 0.239539 | 0.064824 |
| **score** | 0.039389 | 1.000000 | 0.350185 | -0.006406 | 0.124943 | 0.292254 |
| **votes** | 0.296512 | 0.350185 | 1.000000 | 0.346274 | 0.553625 | 0.205344 |
| **budget** | 0.220833 | -0.006406 | 0.346274 | 1.000000 | 0.512057 | 0.231278 |
| **gross** | 0.239539 | 0.124943 | 0.553625 | 0.512057 | 1.000000 | 0.176979 |
| **runtime** | 0.064824 | 0.292254 | 0.205344 | 0.231278 | 0.176979 | 1.000000 |

In [17]:

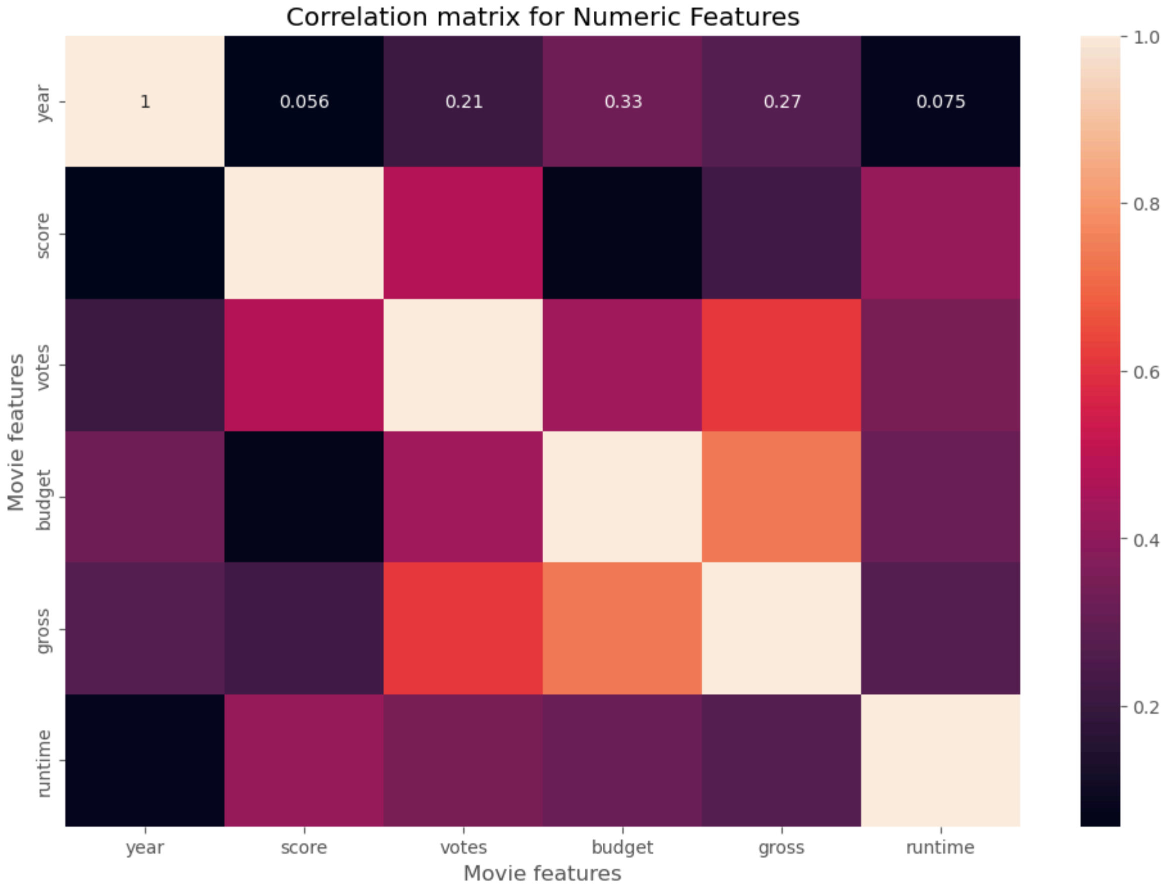
```
df.corr(method ='spearman', numeric_only=True)
```

Out[17]:

| | year | score | votes | budget | gross | runtime |
|---|---|---|---|---|---|---|
| **year** | 1.000000 | 0.057741 | 0.427623 | 0.312886 | 0.351045 | 0.095444 |
| **score** | 0.057741 | 1.000000 | 0.495409 | -0.009971 | 0.183192 | 0.412155 |
| **votes** | 0.427623 | 0.495409 | 1.000000 | 0.493461 | 0.745793 | 0.300621 |
| **budget** | 0.312886 | -0.009971 | 0.493461 | 1.000000 | 0.692958 | 0.330794 |
| **gross** | 0.351045 | 0.183192 | 0.745793 | 0.692958 | 1.000000 | 0.257400 |
| **runtime** | 0.095444 | 0.412155 | 0.300621 | 0.330794 | 0.257400 | 1.000000 |

In [50]:

```
correlation_matrix = df.corr(method ='pearson', numeric_only=True)
sns.heatmap(correlation_matrix, annot=True)
plt.title("Correlation matrix for Numeric Features")
plt.xlabel("Movie features")
plt.ylabel("Movie features")
plt.show()
```



In [41]:

```
# Numerical Representation of Data Frame

df_numerized = df

for col_name in df_numerized.columns:
    if(df_numerized[col_name].dtype == 'object'):
        df_numerized[col_name]= df_numerized[col_name].astype('category')
        df_numerized[col_name] = df_numerized[col_name].cat.codes

df_numerized
```

Out[41]:

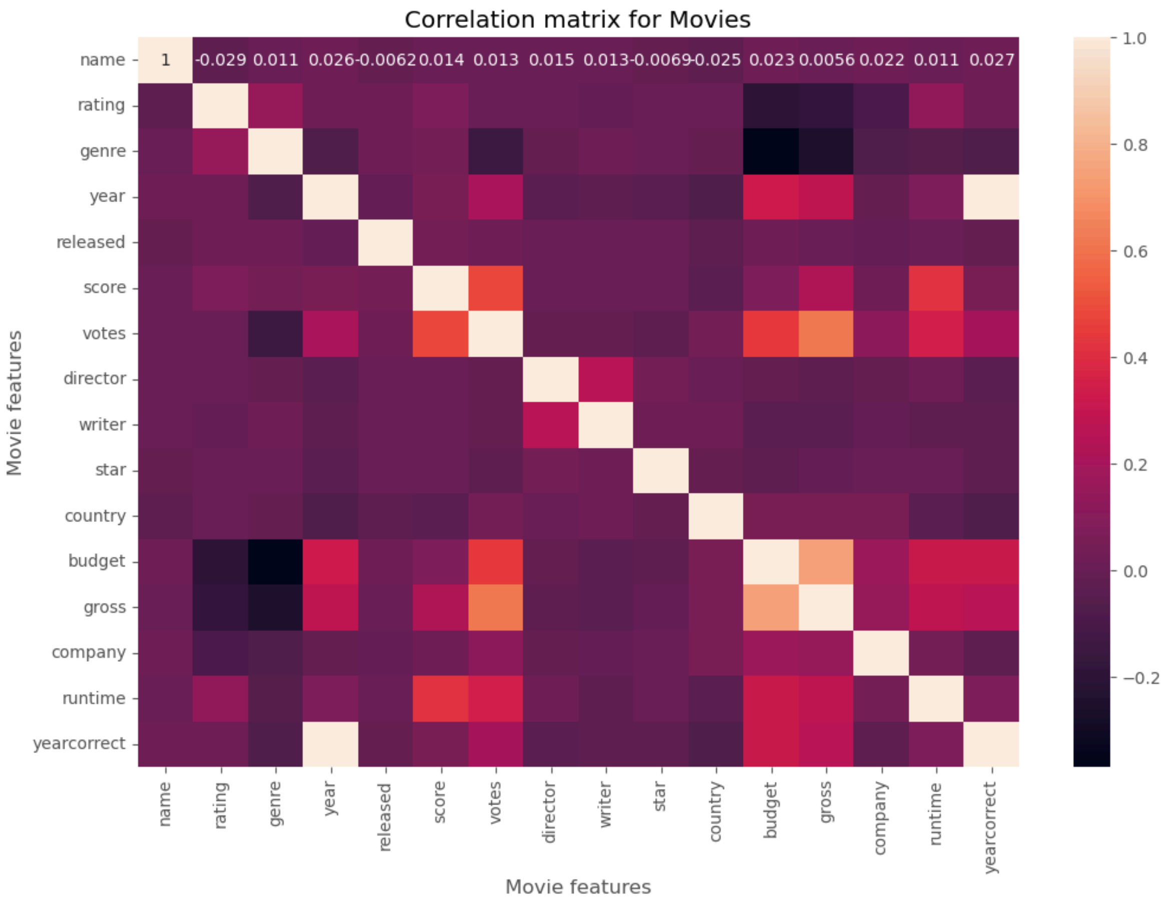
| | name | rating | genre | year | released | score | votes | director | writer | star | c |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 4692 | 6 | 6 | 1980 | 1304 | 8.4 | 927000.0 | 1795 | 2832 | 699 | |
| **1** | 3929 | 6 | 1 | 1980 | 1127 | 5.8 | 65000.0 | 1578 | 1158 | 214 | |
| **2** | 3641 | 4 | 0 | 1980 | 1359 | 8.7 | 1200000.0 | 757 | 1818 | 1157 | |
| **3** | 204 | 4 | 4 | 1980 | 1127 | 7.7 | 221000.0 | 889 | 1413 | 1474 | |
| **4** | 732 | 6 | 4 | 1980 | 1170 | 7.3 | 108000.0 | 719 | 351 | 271 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **7648** | 415 | 6 | 0 | 2020 | 904 | 6.6 | 140000.0 | 16 | 2390 | 1812 | |
| **7649** | 3556 | 4 | 0 | 2020 | 713 | 6.5 | 102000.0 | 852 | 2309 | 147 | |
| **7650** | 1153 | 4 | 1 | 2020 | 904 | 5.6 | 53000.0 | 1809 | 2827 | 1470 | |
| **7651** | 3978 | 4 | 1 | 2020 | 758 | 6.8 | 42000.0 | 294 | 2091 | 640 | |
| **7652** | 4090 | 3 | 0 | 2020 | 370 | 6.8 | 3700.0 | 746 | 1184 | 1839 | |

5421 rows × 16 columns

In [42]:

```
# Correlation Matrix for All Columns

correlation_matrix = df_numerized.corr(method='pearson', numeric_only=True)
sns.heatmap(correlation_matrix, annot = True)
plt.title("Correlation matrix for Movies")
plt.xlabel("Movie features")
plt.ylabel("Movie features")
plt.show()
```


Correlation matrix for Movies
name
rating
genre
year
released
score
votes
director
writer
star
country
budget
gross
company
runtime
yearcorrect
1
-0.029
0.011
0.026
-0.0062
0.014
0.013
0.015
0.013
-0.0069
-0.025
0.023
0.0056
0.022
0.011
0.027
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
Movie features
Movie features

```
In [ ]:
```