Movie Correlations

February 17, 2025

1 Movie Correlations

1.0.1 import libraries

[168]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
plt.style.use('ggplot')
from matplotlib.pyplot import figure

%matplotlib inline
matplotlib.rcParams['figure.figsize'] = (8, 6) # resizing the plot
pd.options.display.float_format = '{:.2f}'.format # limit outputs to 2 decimal␣
↪places
```

[183]:
```
df = pd.read_csv(r"D:\Analyst materials\projects\files\movies.csv")
df.head()
```

[183]:
```
                                             name rating      genre  year  \
0                                     The Shining      R      Drama  1980
1                                 The Blue Lagoon      R  Adventure  1980
2  Star Wars: Episode V - The Empire Strikes Back     PG     Action  1980
3                                       Airplane!     PG     Comedy  1980
4                                      Caddyshack      R     Comedy  1980

                        released  score      votes         director  \
0  June 13, 1980 (United States)   8.40  927000.00  Stanley Kubrick
1   July 2, 1980 (United States)   5.80   65000.00   Randal Kleiser
2  June 20, 1980 (United States)   8.70 1200000.00   Irvin Kershner
3   July 2, 1980 (United States)   7.70  221000.00     Jim Abrahams
4  July 25, 1980 (United States)   7.30  108000.00     Harold Ramis

                    writer            star         country      budget  \
0             Stephen King  Jack Nicholson  United Kingdom 19000000.00
1  Henry De Vere Stacpoole  Brooke Shields   United States  4500000.00
2           Leigh Brackett     Mark Hamill   United States 18000000.00
```

```
3             Jim Abrahams     Robert Hays   United States  3500000.00
4       Brian Doyle-Murray     Chevy Chase   United States  6000000.00

          gross             company  runtime
0   46998772.00        Warner Bros.   146.00
1   58853106.00   Columbia Pictures   104.00
2  538375067.00           Lucasfilm   124.00
3   83453539.00 Paramount Pictures     88.00
4   39846344.00      Orion Pictures    98.00
```

[184]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7668 entries, 0 to 7667
Data columns (total 15 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   name      7668 non-null   object
 1   rating    7591 non-null   object
 2   genre     7668 non-null   object
 3   year      7668 non-null   int64
 4   released  7666 non-null   object
 5   score     7665 non-null   float64
 6   votes     7665 non-null   float64
 7   director  7668 non-null   object
 8   writer    7665 non-null   object
 9   star      7667 non-null   object
 10  country   7665 non-null   object
 11  budget    5497 non-null   float64
 12  gross     7479 non-null   float64
 13  company   7651 non-null   object
 14  runtime   7664 non-null   float64
dtypes: float64(5), int64(1), object(9)
memory usage: 898.7+ KB
```

1.1 First we do some Data Cleaning

1.1.1 Check for null values in each column

[190]:
```
df.isnull().sum()
```

[190]:
```
name         0
rating      12
genre        0
year         0
released     0
score        0
votes        0
```

```
director     0
writer       0
star         0
country      1
budget       0
gross        0
company      2
runtime      1
dtype: int64
```

1.1.2 Drop rows where budget or gross is null because much of the analysis will be reliant on those values

[186]:
```python
df =  df.dropna(subset=['budget', 'gross'])
```

1.1.3 Changing data type of budget and gross from float to int

[192]:
```python
df['budget'] = df['budget'].astype(int)
df['gross'] = df['gross'].astype(int)
```

1.1.4 Sort movies by highest grossing

[194]:
```python
df.sort_values(by=['gross'], inplace=True, ascending=False)
```

1.1.5 Drop any duplicates

[196]:
```python
df.drop_duplicates().head()
```

[196]:
```
                                       name rating   genre  year  \
5445                                 Avatar  PG-13  Action  2009
7445                      Avengers: Endgame  PG-13  Action  2019
3045                                Titanic  PG-13   Drama  1997
6663  Star Wars: Episode VII - The Force Awakens  PG-13  Action  2015
7244                 Avengers: Infinity War  PG-13  Action  2018

                           released  score       votes        director  \
5445  December 18, 2009 (United States)   7.80 1100000.00   James Cameron
7445     April 26, 2019 (United States)   8.40  903000.00  Anthony Russo
3045  December 19, 1997 (United States)   7.80 1100000.00   James Cameron
6663  December 18, 2015 (United States)   7.80  876000.00     J.J. Abrams
7244     April 27, 2018 (United States)   8.40  897000.00  Anthony Russo

                  writer               star        country     budget  \
5445       James Cameron    Sam Worthington  United States  237000000
7445  Christopher Markus  Robert Downey Jr.  United States  356000000
3045       James Cameron  Leonardo DiCaprio  United States  200000000
6663     Lawrence Kasdan       Daisy Ridley  United States  245000000
```

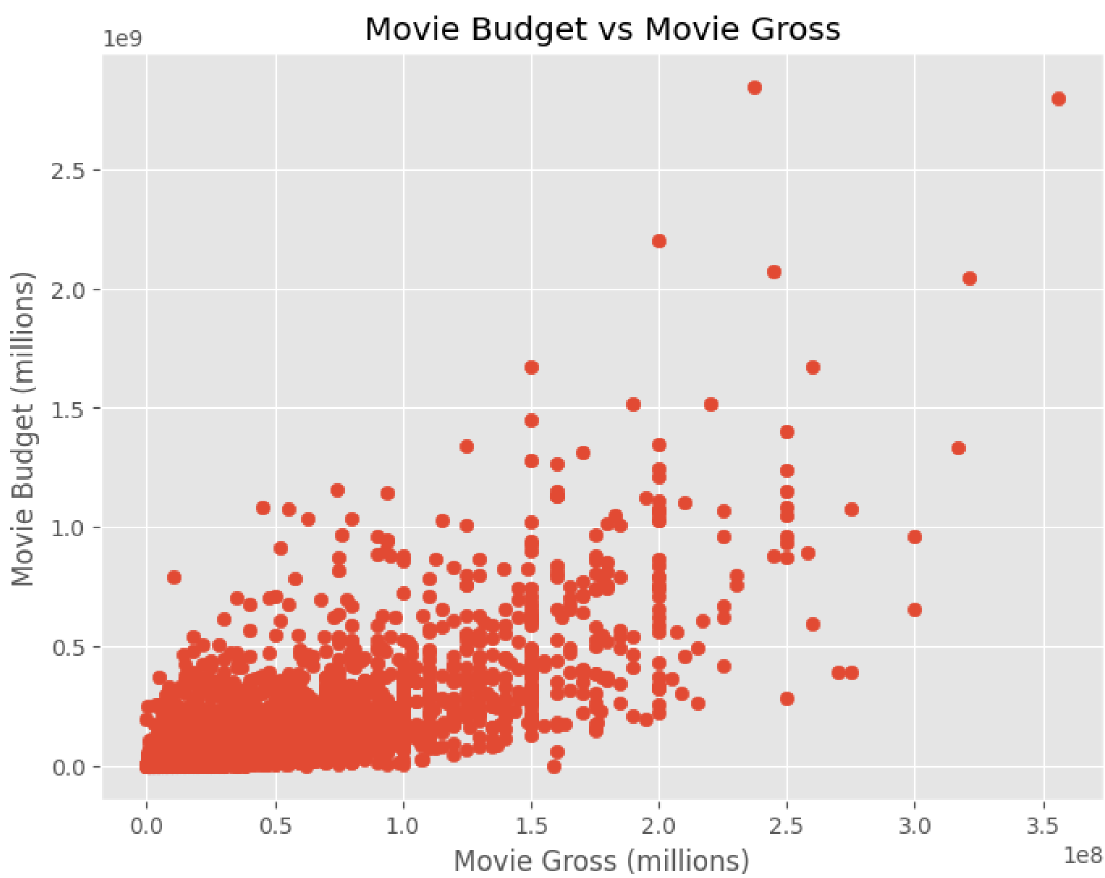
```
7244  Christopher Markus  Robert Downey Jr.  United States  321000000

           gross                company  runtime
5445  2847246203  Twentieth Century Fox   162.00
7445  2797501328         Marvel Studios   181.00
3045  2201647264  Twentieth Century Fox   194.00
6663  2069521700              Lucasfilm   138.00
7244  2048359754         Marvel Studios   149.00
```

1.2 Now we create some visualizations

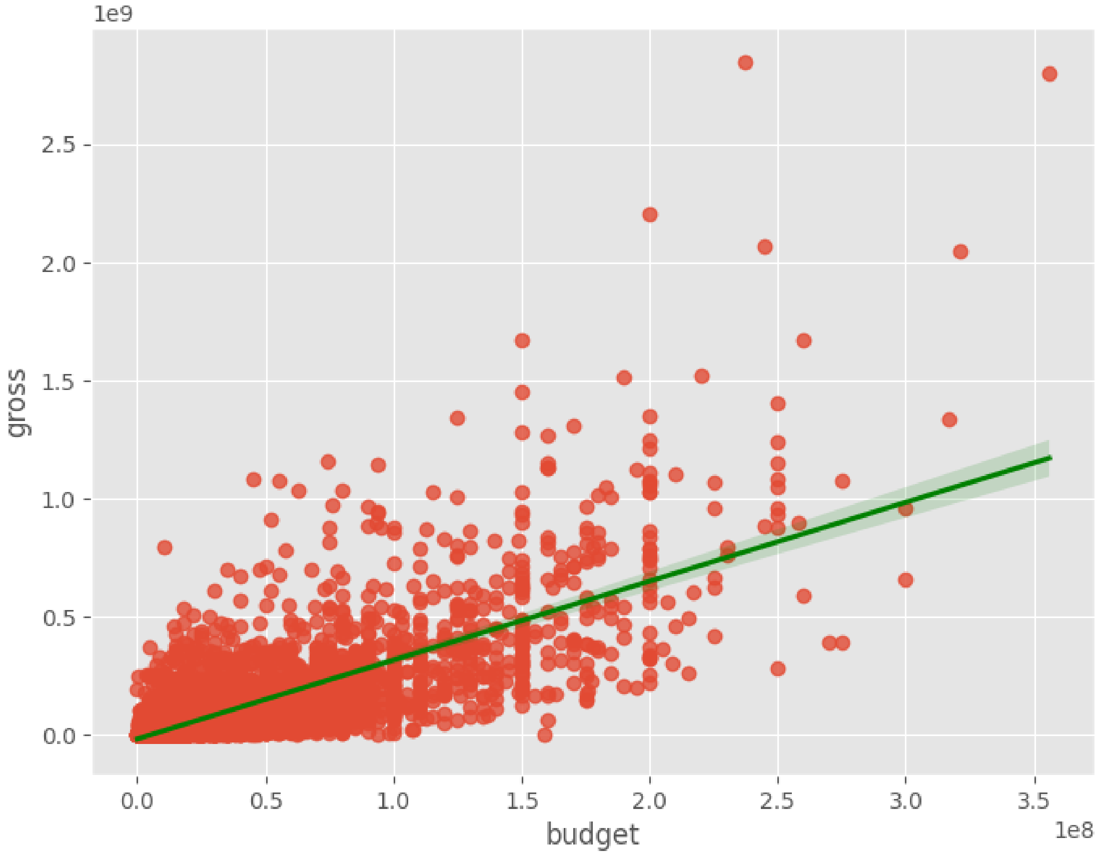
1.2.1 Scatterplot with budget vs gross

[176]:
```python
plt.scatter(x=df['budget'], y=df['gross'])
plt.title('Movie Budget vs Movie Gross')
plt.xlabel('Movie Gross (millions)')
plt.ylabel('Movie Budget (millions)')
plt.show()
```



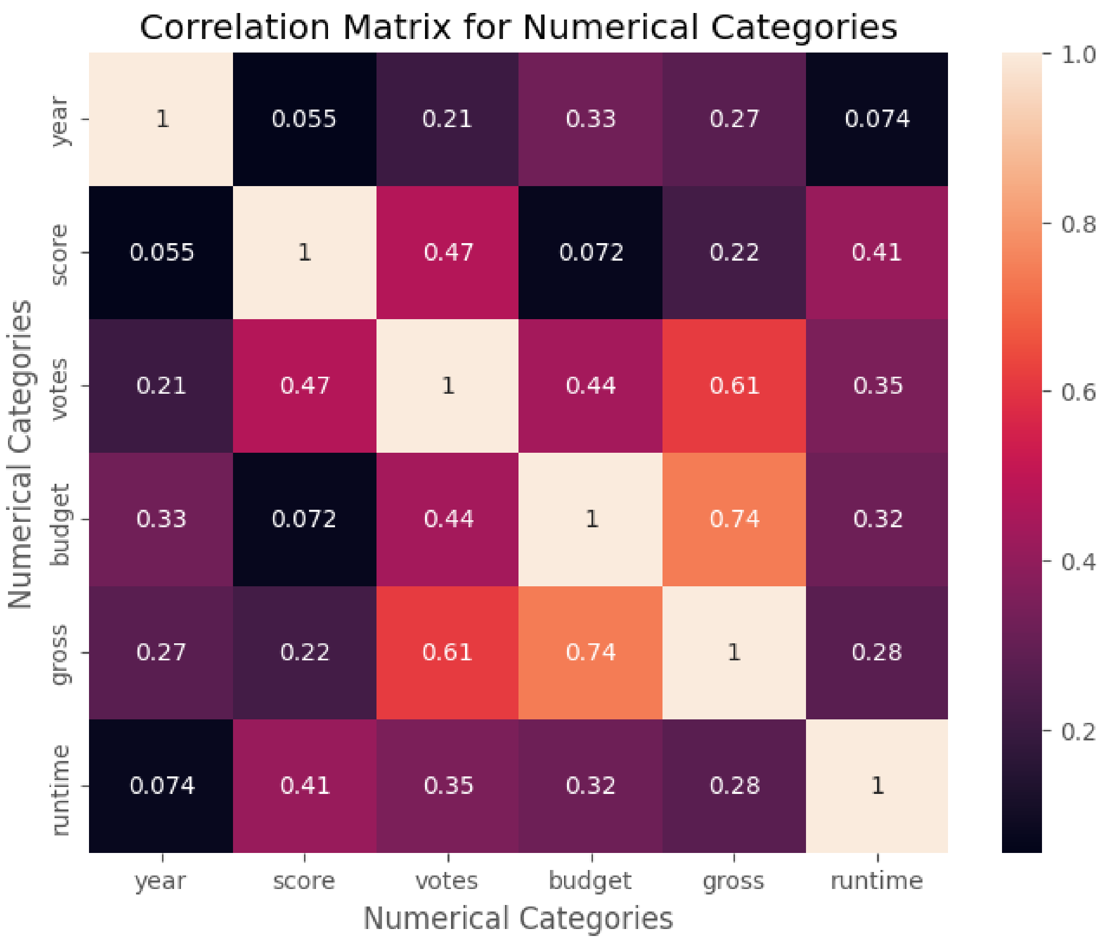
[177]:
```python
sns.regplot(x='budget', y='gross', data=df, line_kws={"color": "green"})
```

[177]: <Axes: xlabel='budget', ylabel='gross'>



1.2.2 Check Correlations

[178]:
```python
corr_matrix = df.corr(numeric_only=True)
sns.heatmap(corr_matrix, annot=True)
plt.title('Correlation Matrix for Numerical Categories')
plt.xlabel('Numerical Categories')
plt.ylabel('Numerical Categories')
plt.show()
```



1.2.3 *We can see a high correlation between* `budget` *and* `gross`*, and a moderately-high correlation between* `votes` *and* `gross`

1.2.4 Want to incorporate non-numeric values into the correlation analysis

[197]:
```python
# assign numerical values to non-numeric fields using category codes
df_num = df
for col in df_num.columns:
    if(df_num[col].dtype == 'object'):
        df_num[col] = df_num[col].astype('category')
        df_num[col] = df_num[col].cat.codes
df_num.head()
```

[197]:

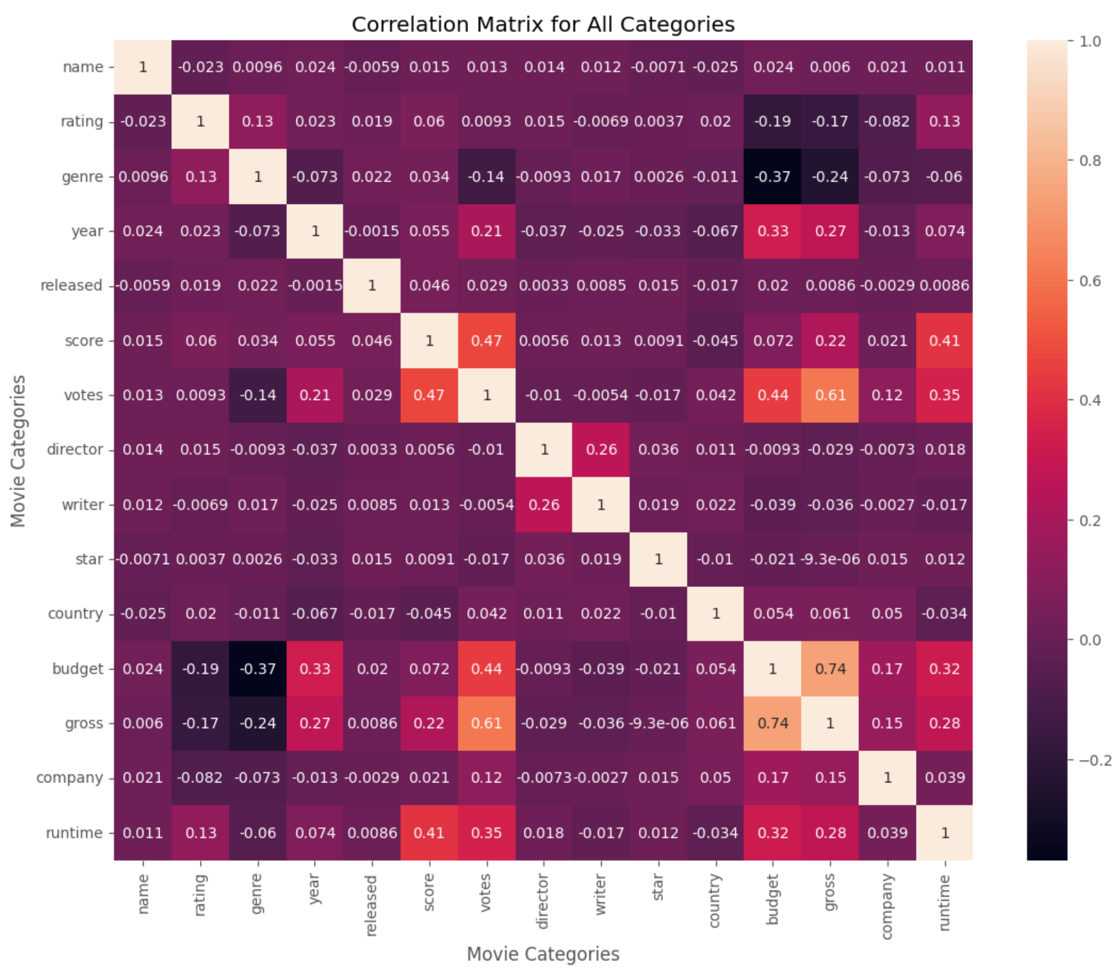
	name	rating	genre	year	released	score	votes	director	writer	\
5445	387	5	0	2009	528	7.80	1100000.00	787	1265	
7445	389	5	0	2019	138	8.40	903000.00	106	515	
3045	4923	5	6	1997	535	7.80	1100000.00	787	1265	
6663	3656	5	0	2015	530	7.80	876000.00	770	1810	

```
7244    390         5        0   2018         146    8.40   897000.00         106       515

       star  country       budget         gross  company  runtime
5445   1538       47   237000000   2847246203     1388   162.00
7445   1474       47   356000000   2797501328      987   181.00
3045   1076       47   200000000   2201647264     1388   194.00
6663    357       47   245000000   2069521700      949   138.00
7244   1474       47   321000000   2048359754      987   149.00
```

[180]:

```python
corr_matrix = df_num.corr(numeric_only=True)
plt.figure(figsize=(13, 10))
sns.heatmap(corr_matrix, annot=True)
plt.title('Correlation Matrix for All Categories')
plt.xlabel('Movie Categories')
plt.ylabel('Movie Categories')
plt.show()
```



[181]:
```
unstacked_corr = df_num.corr().unstack()
unstacked_corr
```

[181]:
```
name      name        1.00
          rating     -0.02
          genre       0.01
          year        0.02
          released   -0.01
                      …
runtime   country    -0.03
          budget      0.32
          gross       0.28
          company     0.04
          runtime     1.00
Length: 225, dtype: float64
```

[182]:
```
# check for all highly correlated variables
high_corr = unstacked_corr[unstacked_corr > 0.5]
high_corr
```

[182]:
```
name      name        1.00
rating    rating      1.00
genre     genre       1.00
year      year        1.00
released  released    1.00
score     score       1.00
votes     votes       1.00
          gross       0.61
director  director    1.00
writer    writer      1.00
star      star        1.00
country   country     1.00
budget    budget      1.00
          gross       0.74
gross     votes       0.61
          budget      0.74
          gross       1.00
company   company     1.00
runtime   runtime     1.00
dtype: float64
```

1.2.5 *Seems like the only highly correlated variables are what we learned from our insight earlier, which is that only `budget` and `gross` and `votes` and `gross` are highly correlated!*