Project: Investigate a Movie Dataset (TMDb Movies Dataset)

Introduction

This is a project towards the Udcity Data Analytics nanodegree certification

This data set contains information about 10,000 movies collected from The Movie Database (TMDb), including user ratings and revenue.

- Certain columns, like ‘cast’ and ‘genres’, contain multiple values separated by pipe (|) characters.
- There are some odd characters in the ‘cast’ column. Don’t worry about cleaning them. You can leave them as is.
- The final two columns ending with “_adj” show the budget and revenue of the associated movie in terms of 2010 dollars, accounting for inflation over time.

some questions of inetrest that arise from the movie features are:

- Which genres are most popular?
- How has movie runtime changed over the years?
- Which genres generate more revenue?
- How has overall movie revenue changed over the years?
- which is the most popular and least popular movie?
- What are the top ten most popular movies?
- What genres do most producers produce?
- How has the cost of movie production changed over the years?

However, we are going to answer the questions.

- How has The cost of movie production changed over the years?
- How has revenue from movie sales et'al changed over the years?
- How has profitability changed over the years?
- with respect to the 2010 inflation rate, comment on effect of onflation on overall cost overhead

we answer these questions using the actual revenue, values and the values adjusted for the 2010 value of the Dollar

```
#import the needed pacjkages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pprint as pp
import os
import sys
from code1 import *
```

```
%matplotlib inline
```

```
file = 'tmdb-movies.csv'
movie_df = pd.read_csv(file)
movie_df.head(3)
```

```
       id    imdb_id  popularity     budget     revenue
original_title  \
0  135397  tt0369610   32.985763  150000000  1513528810      Jurassic
World
1   76341  tt1392190   28.419936  150000000   378436354  Mad Max: Fury
Road
2  262500  tt2908446   13.112507  110000000   295238201
Insurgent

                                                cast  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...
2  Shailene Woodley|Theo James|Kate Winslet|Ansel...

                                         homepage          director  \
0                   http://www.jurassicworld.com/   Colin Trevorrow
1                     http://www.madmaxmovie.com/     George Miller
2  http://www.thedivergentseries.movie/#insurgent  Robert Schwentke

                      tagline  ...  \
0           The park is open.  ...
1          What a Lovely Day.  ...
2  One Choice Can Destroy You  ...

                                            overview runtime  \
0  Twenty-two years after the events of Jurassic ...     124
1  An apocalyptic story set in the furthest reach...     120
2  Beatrice Prior must confront her inner demons ...     119

                                      genres  \
0  Action|Adventure|Science Fiction|Thriller
1  Action|Adventure|Science Fiction|Thriller
2         Adventure|Science Fiction|Thriller

                                production_companies release_date
vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...       6/9/15
5562
1  Village Roadshow Pictures|Kennedy Miller Produ...      5/13/15
6185
2  Summit Entertainment|Mandeville Films|Red Wago...      3/18/15
2480
```

```
   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  1.379999e+08  1.392446e+09
1           7.1          2015  1.379999e+08  3.481613e+08
2           6.3          2015  1.012000e+08  2.716190e+08

[3 rows x 21 columns]
```

Observation

The revenue and budget have no currency specified, however, the TMDb from themoviedb, ",All currency values on TMDb is assumed to be USD."

```python
#dimension of dtaframe
samples, features = movie_df.shape
print('The dataset has {} movie samples and {}
features'.format(samples, features))
```

```
The dataset has 10866 movie samples and 21 features
```

```python
#summary statistics
movie_df.describe()
```

```
                  id    popularity        budget       revenue
runtime  \
count   10866.000000  10866.000000  1.086600e+04  1.086600e+04
10866.000000
mean    66064.177434      0.646441  1.462570e+07  3.982332e+07
102.070863
std     92130.136561      1.000185  3.091321e+07  1.170035e+08
31.381405
min         5.000000      0.000065  0.000000e+00  0.000000e+00
0.000000
25%     10596.250000      0.207583  0.000000e+00  0.000000e+00
90.000000
50%     20669.000000      0.383856  0.000000e+00  0.000000e+00
99.000000
75%     75610.000000      0.713817  1.500000e+07  2.400000e+07
111.000000
max    417859.000000     32.985763  4.250000e+08  2.781506e+09
900.000000

         vote_count  vote_average  release_year    budget_adj
revenue_adj
count  10866.000000  10866.000000  10866.000000  1.086600e+04
1.086600e+04
mean     217.389748      5.974922   2001.322658  1.755104e+07
5.136436e+07
std      575.619058      0.935142     12.812941  3.430616e+07
1.446325e+08
min       10.000000      1.500000   1960.000000  0.000000e+00
```

```
0.000000e+00
25%        17.000000      5.400000   1995.000000  0.000000e+00
0.000000e+00
50%        38.000000      6.000000   2006.000000  0.000000e+00
0.000000e+00
75%       145.750000      6.600000   2011.000000  2.085325e+07
3.369710e+07
max      9767.000000      9.200000   2015.000000  4.250000e+08
2.827124e+09
```

some deductions from the summary decription

- Most movies turn out to be profitable, althou some m
- Most movies have an average runtime of 102 mins
- The dataset contains no details of movies released after 2015 ##### Note:

  This is a summary descriptive statistics, and is not a subtitue for the required indepth analysis and conclusion

Data Wrangling

1. Create a new dataframe(df) from a copy of the Movie_df

2. Drop duplicate movie samples

3. Add features relevant to the analysis

we create a new dataframe using the columns relevant to our analysis
['original_title','runtime','release_year','budget','revenue','budget_adj','revenue_adj']

Other features are important but not with respect to our analysis

1. create new dataframe from our old dataframe

```
cols_to_use =
np.array(['original_title','runtime','release_year','budget','revenue'
,'budget_adj','revenue_adj'])
df = movie_df[cols_to_use].copy()
df.head()
```

```
                 original_title  runtime  release_year     budget
revenue  \
0                Jurassic World      124          2015  150000000
1513528810
1            Mad Max: Fury Road      120          2015  150000000
378436354
2                     Insurgent      119          2015  110000000
295238201
3  Star Wars: The Force Awakens      136          2015  200000000
2068178225
```

```
4                     Furious 7      137          2015  190000000
1506249360

     budget_adj   revenue_adj
0  1.379999e+08  1.392446e+09
1  1.379999e+08  3.481613e+08
2  1.012000e+08  2.716190e+08
3  1.839999e+08  1.902723e+09
4  1.747999e+08  1.385749e+09
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   original_title  10866 non-null  object
 1   runtime         10866 non-null  int64
 2   release_year    10866 non-null  int64
 3   budget          10866 non-null  int64
 4   revenue         10866 non-null  int64
 5   budget_adj      10866 non-null  float64
 6   revenue_adj     10866 non-null  float64
dtypes: float64(2), int64(4), object(1)
memory usage: 594.4+ KB
```

clearly there are no empty cells in our new dataframe

2. Drop duplicate movie samples

```
print("The new tmdb dataset has {} duplicated movie
samples".format(df.duplicated().sum()))
```

```
The new tmdb dataset has 1 duplicated movie samples
```

We then drop duplicate rows(samples) keeping the first instance of their occurence.

```
df.drop_duplicates(keep='first', inplace=True)
```

Check for features with null values and since their datatypes are not int/float, we ignore them.

```
print("the runtime feature has {} nan
values".format(df['runtime'].isna().sum()))
```

```
the runtime feature has 0 nan values
```

3. Add feature(s) relevant to our analysis.

```
df.insert(5,'profit',df['revenue']-df['budget'])
df.insert(8,'profit_adj',df['revenue_adj']-df['budget_adj'])
df.head()
```

```
                 original_title  runtime  release_year     budget
revenue  \
```

```
0                 Jurassic World      124          2015  150000000
1513528810
1             Mad Max: Fury Road      120          2015  150000000
378436354
2                      Insurgent      119          2015  110000000
295238201
3  Star Wars: The Force Awakens      136          2015  200000000
2068178225
4                      Furious 7      137          2015  190000000
1506249360

       profit    budget_adj   revenue_adj    profit_adj
0  1363528810  1.379999e+08  1.392446e+09  1.254446e+09
1   228436354  1.379999e+08  3.481613e+08  2.101614e+08
2   185238201  1.012000e+08  2.716190e+08  1.704191e+08
3  1868178225  1.839999e+08  1.902723e+09  1.718723e+09
4  1316249360  1.747999e+08  1.385749e+09  1.210949e+09
```

Exploratory Data Analysis

function to compute data and make our plots

```python
def feature_info_plot(col_name):
    #data_setup
    Title = f"yearly {col_name}"
    temp_data = df.groupby('release_year')[col_name].sum()

    #plot_face
    plt.figure(figsize=(15,5))
    plt.plot(temp_data)
    plt.title(Title)
    plt.xlabel('year')
    plt.ylabel(col_name)
    plt.show()
```

function for max and min values

```python
def high_and_low(col_name):
    #for highest earned profit
    high= df[col_name].idxmax()
    high_details=pd.DataFrame(df.loc[high])

    #for lowest earned profit
    low= df[col_name].idxmin()
    low_details=pd.DataFrame(df.loc[low])

    #collectin data in one place
    info=pd.concat([high_details, low_details], axis=1)

    return info
```

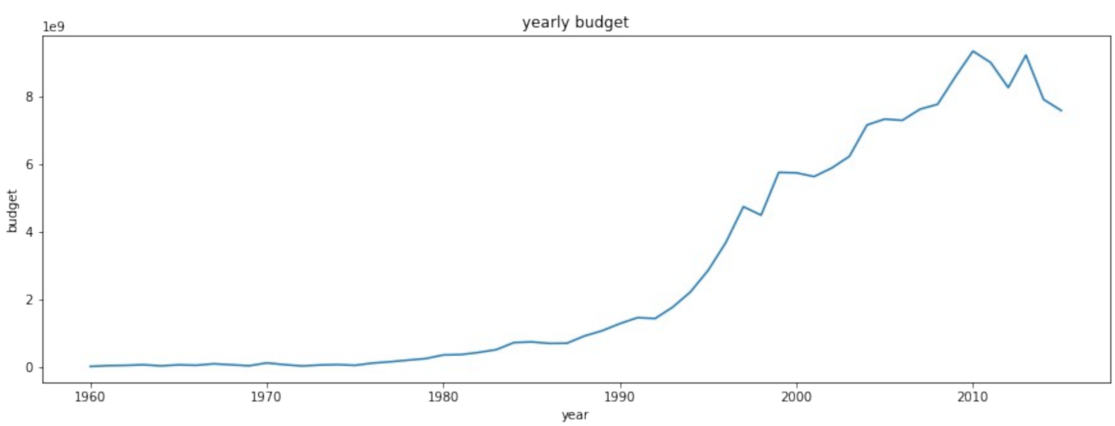
function for line of best fit

```
def plot_best_fit(col_name):
    #data setup
    Title = f"yearly {col_name}"
    temp_data = df.groupby('release_year')[col_name].sum()
    pds = pd.Series(temp_data)
    years = np.array(pds.index.values)
    yearly_values = np.array([pds[i] for i in years])
    x,y = years, yearly_values

    #gradient and intercept
    m,c = np.polyfit(x, y, 1)

    #plot of best fit
    plt.figure(figsize=(15,5))
    plt.plot(x,y,'*')
    plt.plot(x, m*x + c)
    plt.title(Title)
    plt.xlabel('year')
    plt.ylabel(col_name)
    plt.show()
```

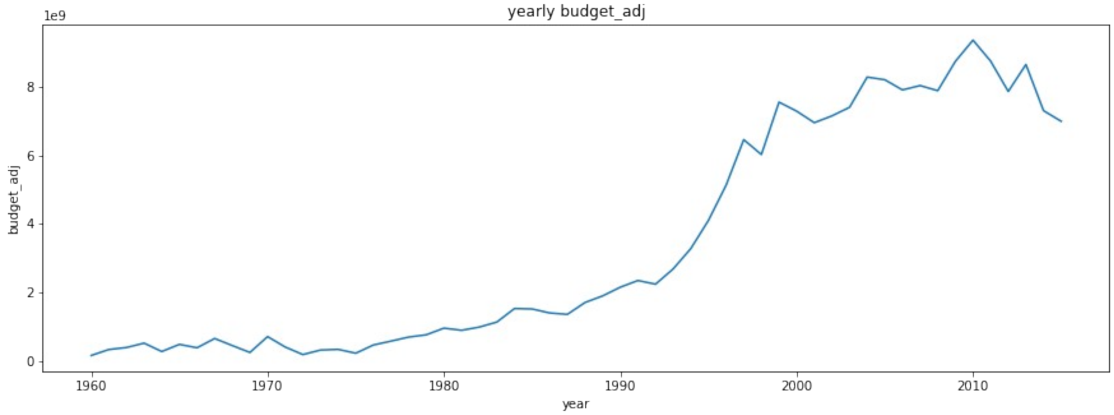
Q1. How has the cost of movie production changed over the years?

Note: The Budget is the overhead cost of movie production.

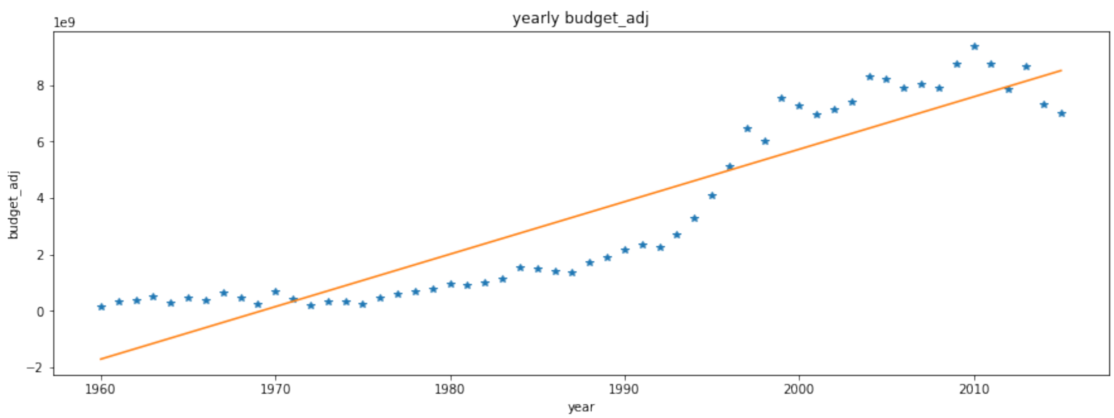
```
#plot for yearly budget
feature_info_plot('budget')
```




```
#plot for yearly budget adjusted to the 2010 Dollar value
feature_info_plot('budget_adj')
```




```
#budget plot of best fit
plot_best_fit('budget_adj')
```




```
pd.concat([high_and_low('budget'), high_and_low('budget_adj')],
axis=1)
```

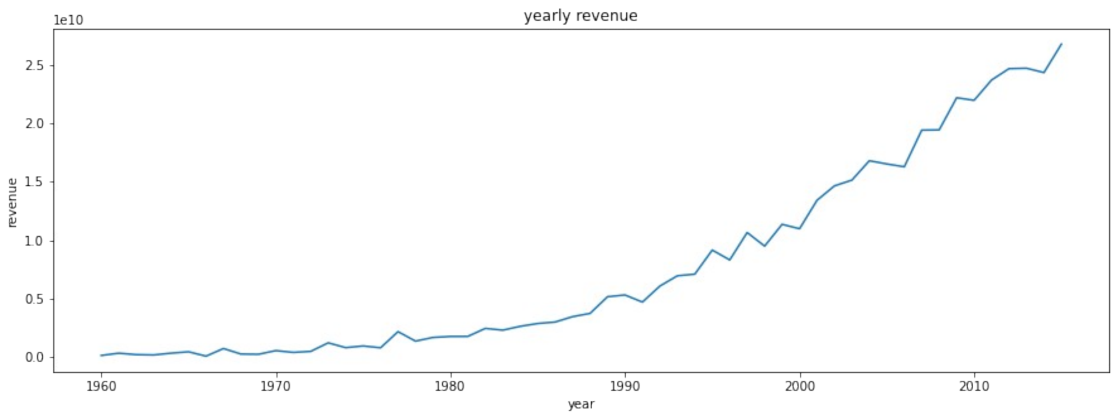
```
                              2244           30               2244  \
original_title  The Warrior's Way    Mr. Holmes  The Warrior's Way
runtime                        100          103                100
release_year                  2010         2015               2010
budget                   425000000            0          425000000
revenue                   11087569     29355203           11087569
profit                  -413912431     29355203         -413912431
budget_adj             425000000.0          0.0        425000000.0
revenue_adj             11087569.0  27006774.877019     11087569.0
```

```
profit_adj           -413912431.0  27006774.877019       -413912431.0
```

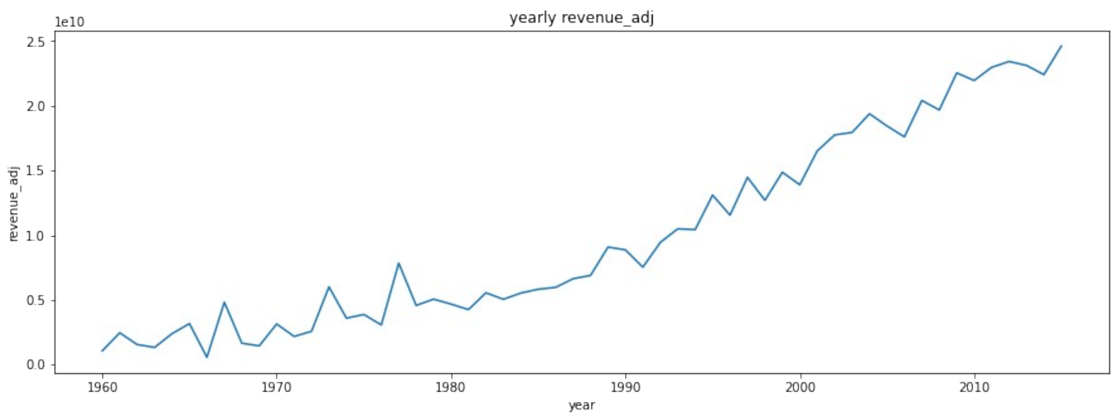
```
                             30
original_title       Mr. Holmes
runtime                     103
release_year               2015
budget                        0
revenue                29355203
profit                 29355203
budget_adj                  0.0
revenue_adj     27006774.877019
profit_adj      27006774.877019
```

Q2. How has revenue from movie sales and publicity changed over the the years

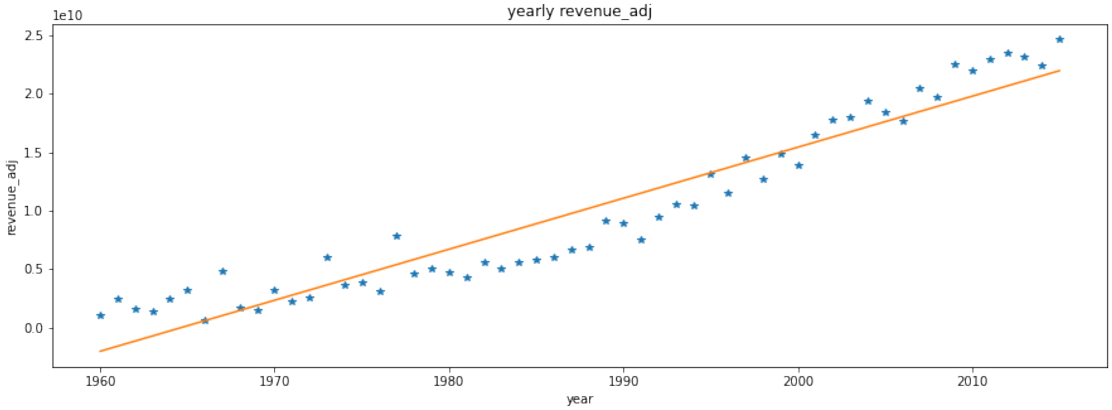
```
#plot for yearly revenue unadjusted
feature_info_plot('revenue')
```




```
#plot for yearly revenue adjusted
feature_info_plot('revenue_adj')
```




```
plot_best_fit('revenue_adj')
```

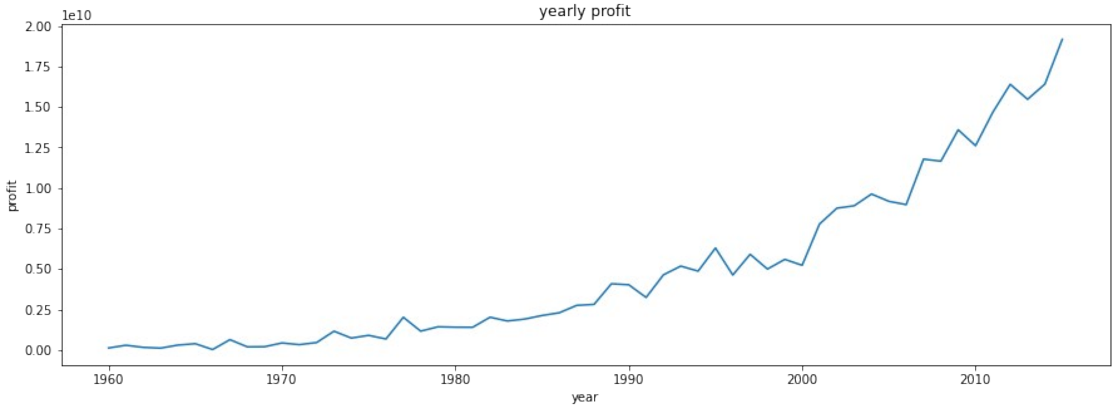



```
pd.concat([high_and_low('revenue'), high_and_low('revenue_adj')],
axis=1)
```

```
                              1386               48                1386
\
original_title              Avatar        Wild Card              Avatar

runtime                        162               92                 162

release_year                  2009             2015                2009

budget                   237000000         30000000           237000000

revenue                 2781505847                0          2781505847

profit                  2544505847        -30000000          2544505847

budget_adj        240886902.887613  27599987.856005    240886902.887613

revenue_adj       2827123750.41189              0.0    2827123750.41189

profit_adj       2586236847.524277 -27599987.856005   2586236847.524277


                            48
original_title       Wild Card
runtime                     92
release_year              2015
budget                30000000
revenue                      0
profit               -30000000
budget_adj     27599987.856005
revenue_adj                0.0
profit_adj    -27599987.856005
```

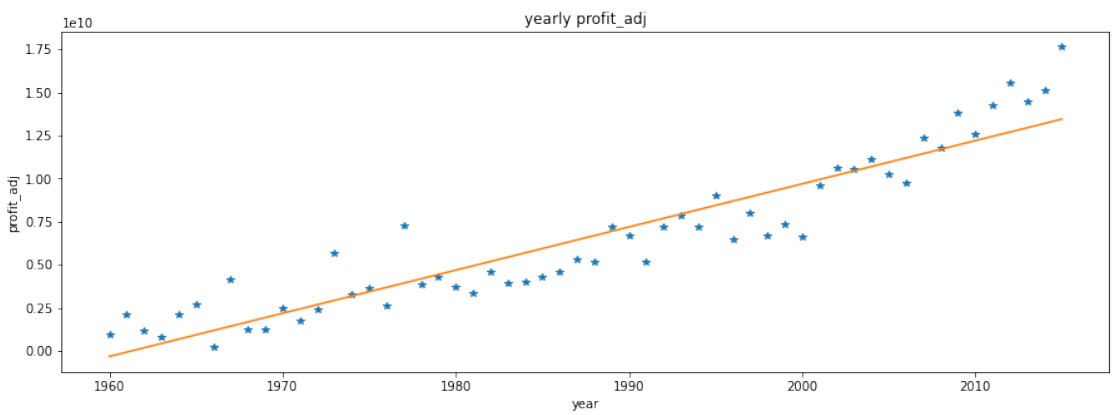
Q3. How Profitable has movie production been over the years.

```
#plot for profit values unadjusted
feature_info_plot('profit')
```




```
#plot for adjusted profit values
feature_info_plot('profit_adj')
```


```
#profit plot of best fit
plot_best_fit('profit_adj')
```




```
pd.concat([high_and_low('profit'), high_and_low('profit_adj')],
axis=1)
```

```
                            1386               2244
1329  \
original_title            Avatar  The Warrior's Way             Star
Wars
runtime                      162                100
121
release_year                2009               2010
1977
budget                 237000000          425000000
11000000
revenue               2781505847           11087569
775398007
profit                2544505847         -413912431
764398007
budget_adj      240886902.887613        425000000.0
39575591.358274
revenue_adj      2827123750.41189         11087569.0
2789712242.27745
profit_adj     2586236847.524277       -413912431.0
2750136650.919176

                            2244
original_title  The Warrior's Way
runtime                      100
release_year                2010
budget                 425000000
revenue                 11087569
profit                -413912431
budget_adj           425000000.0
revenue_adj           11087569.0
profit_adj          -413912431.0
```

Conculusions

1. Budget

- There was a sharp increase in cost of movie production from te early 1990s and peaked in the year 2009-2010 and then begins to decline.
- The rising cost of movie production between the 1990s and the year 2010 could be attributed to the use of state of the art technological equipments in movie oroduction.
- The decline from the year 2010 could be attributed to increased adoption which has cut down the required manpower, thereby reducing the overall budget.

2. Revenue

- Movie revenue has seen an almost steady increase since the birth of movie production
- Inflation has had little effect on Movie Revenue.
- This shows growing interest in viewership.

3. Profit

- The overall profit shows some seasonality but has a summarily profitable trend
- from the high and low dataframes, we can see that a movie budget is not always an indicator high revenue or profit

Financial advice

- For an investor who wants to know if an investments in the movie industry is a worthy in the longterm, my answer is this since there is no significant difference in the adjusted and axctual values of the budget, revenue, and profit, the data at hand that the movie Industry is investment worthy.