Investigate_a_Dataset[TMDb movie Database]

August 18, 2022

> **Tip**: Welcome to the Investigate a Dataset project! You will find tips in quoted sections like this to help organize your approach to your investigation. Once you complete this project, remove these **Tip** sections from your report before submission. First things first, you might want to double-click this Markdown cell and change the title so that it reflects your dataset and investigation.

1 Project: Investigate a Dataset - [[TMDb movie Database]

1.1 Table of Contents

## Introduction

1.1.1 Dataset Description

This data set contains information about 10,000 movies collected from The Movie Database (TMDb)

1.1.2 Question(s) for Analysis

Below are the questions addressed in this analysis:

1. How the profit is affected by the revenue, budget, popularity, runtime.
2. Movie with the highest/lowest profit/budget/revenue.
3. Find the average budget/revenue/profit/runtime/popularity of all movies

```
In [1]: #Import the neccesary packages
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        % matplotlib inline
```

```
In [2]: # Upgrade pandas to use dataframe.explode() function.
        !pip install --upgrade pandas==0.25.0
```

```
Collecting pandas==0.25.0
  Downloading https://files.pythonhosted.org/packages/1d/9a/7eb9952f4b4d73fbd75ad1d5d6112f407e69
    100% || 10.5MB 3.8MB/s eta 0:00:01  6% |                              | 645kB 20.5MB/s eta 0
Requirement already satisfied, skipping upgrade: python-dateutil>=2.6.1 in /opt/conda/lib/python
Collecting numpy>=1.13.3 (from pandas==0.25.0)
  Downloading https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d63048c7696804a0d9473
    100% || 13.4MB 2.6MB/s eta 0:00:01  7% |                             | 1.1MB 23.7MB/s eta 0:
Requirement already satisfied, skipping upgrade: pytz>=2017.2 in /opt/conda/lib/python3.6/site-p
Requirement already satisfied, skipping upgrade: six>=1.5 in /opt/conda/lib/python3.6/site-packa
tensorflow 1.3.0 requires tensorflow-tensorboard<0.2.0,>=0.1.0, which is not installed.
Installing collected packages: numpy, pandas
  Found existing installation: numpy 1.12.1
    Uninstalling numpy-1.12.1:
      Successfully uninstalled numpy-1.12.1
  Found existing installation: pandas 0.23.3
    Uninstalling pandas-0.23.3:
      Successfully uninstalled pandas-0.23.3
Successfully installed numpy-1.19.5 pandas-0.25.0
```

## Data Wrangling

> **Tip**: In this section of the report, you will load in the data, check for cleanliness, and then trim and clean your dataset for analysis. Make sure that you **document your data cleaning steps in mark-down cells precisely and justify your cleaning decisions.**

1.1.3 General Properties

> **Tip**: You should *not* perform too many operations in each cell. Create cells freely to explore your data. One option that you can take with this project is to do a lot of explorations in an initial notebook. These don't have to be organized, but make sure you use enough comments to understand the purpose of each code cell. Then, after you're done with your analysis, create a duplicate notebook where you will trim the excess and organize your steps so that you have a flowing, cohesive report.

```
In [2]: # Load your data and print out a few lines. Perform operations to inspect data
        #   types and look for instances of missing or possibly errant data.
        df = pd.read_csv('tmdb-movies.csv')
        df.head()
```

```
Out[2]:        id    imdb_id  popularity     budget     revenue  \
        0  135397  tt0369610   32.985763  150000000  1513528810
        1   76341  tt1392190   28.419936  150000000   378436354
        2  262500  tt2908446   13.112507  110000000   295238201
        3  140607  tt2488496   11.173104  200000000  2068178225
        4  168259  tt2820852    9.335014  190000000  1506249360

                         original_title  \
        0                Jurassic World
```

```
1           Mad Max: Fury Road
2                    Insurgent
3  Star Wars: The Force Awakens
4                    Furious 7

                                                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...
3  Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...
4  Vin Diesel|Paul Walker|Jason Statham|Michelle ...

                                            homepage          director  \
0                      http://www.jurassicworld.com/   Colin Trevorrow
1                        http://www.madmaxmovie.com/     George Miller
2     http://www.thedivergentseries.movie/#insurgent  Robert Schwentke
3  http://www.starwars.com/films/star-wars-episod...       J.J. Abrams
4                           http://www.furious7.com/         James Wan

                         tagline      ...       \
0              The park is open.      ...
1             What a Lovely Day.      ...
2     One Choice Can Destroy You      ...
3  Every generation has a story.      ...
4            Vengeance Hits Home      ...

                                            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
3  Thirty years after defeating the Galactic Empi...     136
4  Deckard Shaw seeks revenge against Dominic Tor...     137

                                      genres  \
0  Action|Adventure|Science Fiction|Thriller
1  Action|Adventure|Science Fiction|Thriller
2         Adventure|Science Fiction|Thriller
3   Action|Adventure|Science Fiction|Fantasy
4                      Action|Crime|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
3          Lucasfilm|Truenorth Productions|Bad Robot     12/15/15       5292
4  Universal Pictures|Original Film|Media Rights ...       4/1/15       2947

   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           7.5          2015  1.839999e+08  1.902723e+09
        4           7.3          2015  1.747999e+08  1.385749e+09

        [5 rows x 21 columns]
```

```
In [3]: #Find out the dimension of the dataframe-the numbers of rows and columns in the datafram
        df.shape
```

```
Out[3]: (10866, 21)
```

```
In [4]: #Print the information of the dataframe
        df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 21 columns):
id                      10866 non-null int64
imdb_id                 10856 non-null object
popularity              10866 non-null float64
budget                  10866 non-null int64
revenue                 10866 non-null int64
original_title          10866 non-null object
cast                    10790 non-null object
homepage                2936 non-null object
director                10822 non-null object
tagline                 8042 non-null object
keywords                9373 non-null object
overview                10862 non-null object
runtime                 10866 non-null int64
genres                  10843 non-null object
production_companies    9836 non-null object
release_date            10866 non-null object
vote_count              10866 non-null int64
vote_average            10866 non-null float64
release_year            10866 non-null int64
budget_adj              10866 non-null float64
revenue_adj             10866 non-null float64
dtypes: float64(4), int64(6), object(11)
memory usage: 1.7+ MB
```

```
In [5]: #Get the statistics summary of the dataframe
        df.describe()
```

```
Out[5]:                   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
```

```
In [6]: #Select relevant columns to use
        df = df[['popularity', 'id', 'budget', 'revenue', 'runtime', 'original_title', 'release_
        df.head()
```

```
Out[6]:    popularity      id     budget     revenue  runtime  \
        0   32.985763  135397  150000000  1513528810      124
        1   28.419936   76341  150000000   378436354      120
        2   13.112507  262500  110000000   295238201      119
        3   11.173104  140607  200000000  2068178225      136
        4    9.335014  168259  190000000  1506249360      137

                        original_title  release_year    budget_adj   revenue_adj  \
        0               Jurassic World          2015  1.379999e+08  1.392446e+09
        1           Mad Max: Fury Road          2015  1.379999e+08  3.481613e+08
        2                    Insurgent          2015  1.012000e+08  2.716190e+08
        3  Star Wars: The Force Awakens          2015  1.839999e+08  1.902723e+09
        4                    Furious 7          2015  1.747999e+08  1.385749e+09

           vote_count  vote_average
        0        5562           6.5
        1        6185           7.1
        2        2480           6.3
        3        5292           7.5
        4        2947           7.3
```

```
In [7]: #Print the information of the dataframe
        df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 11 columns):
```

```
popularity          10866 non-null float64
id                  10866 non-null int64
budget              10866 non-null int64
revenue             10866 non-null int64
runtime             10866 non-null int64
original_title      10866 non-null object
release_year        10866 non-null int64
budget_adj          10866 non-null float64
revenue_adj         10866 non-null float64
vote_count          10866 non-null int64
vote_average        10866 non-null float64
dtypes: float64(4), int64(6), object(1)
memory usage: 933.9+ KB
```

1.1.4 Data Cleaning

> **Tip**: Make sure that you keep your reader informed on the steps that you are taking in your investigation. Follow every code cell, or every set of related code cells, with a markdown cell to describe to the reader what was found in the preceding cell(s). Try to make it so that the reader can then understand what they will be seeing in the following cell(s).

```
In [8]: # After discussing the structure of the data and any problems that need to be
        #   cleaned, perform those cleaning steps in the second part of this section.
        #check for null values
        df.isnull().sum()
```

```
Out[8]: popularity        0
        id                0
        budget            0
        revenue           0
        runtime           0
        original_title    0
        release_year      0
        budget_adj        0
        revenue_adj       0
        vote_count        0
        vote_average      0
        dtype: int64
```

```
In [9]: #Create a new column, profit
        df['profit'] = df['revenue'] - df['budget']
        df.head()
```

```
Out[9]:    popularity      id     budget     revenue  runtime  \
        0   32.985763  135397  150000000  1513528810      124
        1   28.419936   76341  150000000   378436354      120
        2   13.112507  262500  110000000   295238201      119
```

```
3   11.173104  140607  200000000  2068178225      136
4    9.335014  168259  190000000  1506249360      137

                 original_title  release_year    budget_adj   revenue_adj  \
0                Jurassic World          2015  1.379999e+08  1.392446e+09
1            Mad Max: Fury Road          2015  1.379999e+08  3.481613e+08
2                     Insurgent          2015  1.012000e+08  2.716190e+08
3  Star Wars: The Force Awakens          2015  1.839999e+08  1.902723e+09
4                     Furious 7          2015  1.747999e+08  1.385749e+09

   vote_count  vote_average      profit
0        5562           6.5  1363528810
1        6185           7.1   228436354
2        2480           6.3   185238201
3        5292           7.5  1868178225
4        2947           7.3  1316249360
```

```
In [10]: #Drop any available duplicates in the dataset
         df.drop_duplicates()
```

```
Out[10]:        popularity      id     budget     revenue  runtime  \
          0       32.985763  135397  150000000  1513528810      124
          1       28.419936   76341  150000000   378436354      120
          2       13.112507  262500  110000000   295238201      119
          3       11.173104  140607  200000000  2068178225      136
          4        9.335014  168259  190000000  1506249360      137
          5        9.110700  281957  135000000   532950503      156
          6        8.654359   87101  155000000   440603537      125
          7        7.667400  286217  108000000   595380321      141
          8        7.404165  211672   74000000  1156730962       91
          9        6.326804  150540  175000000   853708609       94
          10       6.200282  206647  245000000   880674609      148
          11       6.189369   76757  176000003   183987723      124
          12       6.118847  264660   15000000    36869414      108
          13       5.984995  257344   88000000   243637091      105
          14       5.944927   99861  280000000  1405035767      141
          15       5.898400  273248   44000000   155760117      167
          16       5.749758  260346   48000000   325771424      109
          17       5.573184  102899  130000000   518602163      115
          18       5.556818  150689   95000000   542351353      112
          19       5.476958  131634  160000000   650523427      136
          20       5.462138  158852  190000000   209035668      130
          21       5.337064  307081   30000000    91709827      123
          22       4.907832  254128  110000000   470490832      114
          23       4.710402  216015   40000000   569651467      125
          24       4.648046  318846   28000000   133346506      130
          25       4.566713  177677  150000000   682330139      131
          26       4.564549  214756   68000000   215863606      115
```

```
27       4.503789  207703   81000000   403802136      130
28       4.062293  314365   20000000    88346473      128
29       3.968891  294254   61000000   311256926      132
...           ...     ...        ...         ...      ...
10836    0.239435   38720          0           0      114
10837    0.291704   19728          0           0      156
10838    0.151845   22383          0           0      117
10839    0.276133   13353          0           0       25
10840    0.102530   34388          0           0      102
10841    0.264925   42701      75000           0       82
10842    0.253437   36540          0           0       25
10843    0.252399   29710          0           0      134
10844    0.236098   23728          0           0      108
10845    0.230873    5065          0           0       93
10846    0.212716   17102          0           0       90
10847    0.034555   28763          0           0       89
10848    0.207257    2161    5115000    12000000      100
10849    0.206537   28270          0           0      109
10850    0.202473   26268          0           0      121
10851    0.342791   15347          0           0       95
10852    0.227220   37301          0           0       95
10853    0.163592   15598          0           0      114
10854    0.146402   31602          0           0      135
10855    0.141026   13343     700000           0       90
10856    0.140934   20277          0           0       93
10857    0.131378    5921          0           0      128
10858    0.317824   31918          0           0      126
10859    0.089072   20620          0           0      100
10860    0.087034    5060          0           0       87
10861    0.080598      21          0           0       95
10862    0.065543   20379          0           0      176
10863    0.065141   39768          0           0       94
10864    0.064317   21449          0           0       80
10865    0.035919   22293      19000           0       74

                                         original_title  release_year  \
0                                        Jurassic World          2015
1                                    Mad Max: Fury Road          2015
2                                             Insurgent          2015
3                          Star Wars: The Force Awakens          2015
4                                             Furious 7          2015
5                                           The Revenant          2015
6                                    Terminator Genisys          2015
7                                            The Martian          2015
8                                               Minions          2015
9                                            Inside Out          2015
10                                              Spectre          2015
11                                     Jupiter Ascending          2015
```

```
12                                               Ex Machina          2015
13                                                   Pixels          2015
14                                  Avengers: Age of Ultron          2015
15                                        The Hateful Eight          2015
16                                                  Taken 3          2015
17                                                  Ant-Man          2015
18                                               Cinderella          2015
19                    The Hunger Games: Mockingjay - Part 2          2015
20                                             Tomorrowland          2015
21                                                 Southpaw          2015
22                                              San Andreas          2015
23                                     Fifty Shades of Grey          2015
24                                            The Big Short          2015
25                       Mission: Impossible - Rogue Nation          2015
26                                                    Ted 2          2015
27                             Kingsman: The Secret Service          2015
28                                                Spotlight          2015
29                           Maze Runner: The Scorch Trials          2015
...                                                     ...           ...
10836                                    Walk Don't Run          1966
10837                                      The Blue Max          1966
10838                                 The Professionals          1966
10839              It's the Great Pumpkin, Charlie Brown          1966
10840                                 Funeral in Berlin          1966
10841                                      The Shooting          1966
10842                 Winnie the Pooh and the Honey Tree          1966
10843                                          Khartoum          1966
10844                                     Our Man Flint          1966
10845                                   Carry On Cowboy          1966
10846                       Dracula: Prince of Darkness          1966
10847                                  Island of Terror          1966
10848                                  Fantastic Voyage          1966
10849                                            Gambit          1966
10850                                            Harper          1966
10851                                         Born Free          1966
10852                    A Big Hand for the Little Lady          1966
10853                                             Alfie          1966
10854                                         The Chase          1966
10855                           The Ghost & Mr. Chicken          1966
10856                                The Ugly Dachshund          1966
10857                                      Nevada Smith          1966
10858  The Russians Are Coming, The Russians Are Coming          1966
10859                                           Seconds          1966
10860                               Carry On Screaming!          1966
10861                                The Endless Summer          1966
10862                                        Grand Prix          1966
10863                               Beregis Avtomobilya          1966
10864                            What's Up, Tiger Lily?          1966
```

```
10865                       Manos: The Hands of Fate         1966

        budget_adj   revenue_adj  vote_count  vote_average      profit
0     1.379999e+08  1.392446e+09        5562           6.5  1363528810
1     1.379999e+08  3.481613e+08        6185           7.1   228436354
2     1.012000e+08  2.716190e+08        2480           6.3   185238201
3     1.839999e+08  1.902723e+09        5292           7.5  1868178225
4     1.747999e+08  1.385749e+09        2947           7.3  1316249360
5     1.241999e+08  4.903142e+08        3929           7.2   397950503
6     1.425999e+08  4.053551e+08        2598           5.8   285603537
7     9.935996e+07  5.477497e+08        4572           7.6   487380321
8     6.807997e+07  1.064192e+09        2893           6.5  1082730962
9     1.609999e+08  7.854116e+08        3935           8.0   678708609
10    2.253999e+08  8.102203e+08        3254           6.2   635674609
11    1.619199e+08  1.692686e+08        1937           5.2     7987720
12    1.379999e+07  3.391985e+07        2854           7.6    21869414
13    8.095996e+07  2.241460e+08        1575           5.8   155637091
14    2.575999e+08  1.292632e+09        4304           7.4  1125035767
15    4.047998e+07  1.432992e+08        2389           7.4   111760117
16    4.415998e+07  2.997096e+08        1578           6.1   277771424
17    1.195999e+08  4.771138e+08        3779           7.0   388602163
18    8.739996e+07  4.989630e+08        1495           6.8   447351353
19    1.471999e+08  5.984813e+08        2380           6.5   490523427
20    1.747999e+08  1.923127e+08        1899           6.2    19035668
21    2.759999e+07  8.437300e+07        1386           7.3    61709827
22    1.012000e+08  4.328514e+08        2060           6.1   360490832
23    3.679998e+07  5.240791e+08        1865           5.3   529651467
24    2.575999e+07  1.226787e+08        1545           7.3   105346506
25    1.379999e+08  6.277435e+08        2349           7.1   532330139
26    6.255997e+07  1.985944e+08        1666           6.3   147863606
27    7.451997e+07  3.714978e+08        3833           7.6   322802136
28    1.839999e+07  8.127872e+07        1559           7.8    68346473
29    5.611998e+07  2.863562e+08        1849           6.4   250256926
...            ...           ...         ...           ...         ...
10836 0.000000e+00  0.000000e+00          11           5.8           0
10837 0.000000e+00  0.000000e+00          12           5.5           0
10838 0.000000e+00  0.000000e+00          21           6.0           0
10839 0.000000e+00  0.000000e+00          49           7.2           0
10840 0.000000e+00  0.000000e+00          13           5.7           0
10841 5.038511e+05  0.000000e+00          12           5.5      -75000
10842 0.000000e+00  0.000000e+00          12           7.9           0
10843 0.000000e+00  0.000000e+00          12           5.8           0
10844 0.000000e+00  0.000000e+00          13           5.6           0
10845 0.000000e+00  0.000000e+00          15           5.9           0
10846 0.000000e+00  0.000000e+00          16           5.7           0
10847 0.000000e+00  0.000000e+00          13           5.3           0
10848 3.436265e+07  8.061618e+07          42           6.7     6885000
10849 0.000000e+00  0.000000e+00          14           6.1           0
```

```
10850  0.000000e+00  0.000000e+00            14            6.0            0
10851  0.000000e+00  0.000000e+00            15            6.6            0
10852  0.000000e+00  0.000000e+00            11            6.0            0
10853  0.000000e+00  0.000000e+00            26            6.2            0
10854  0.000000e+00  0.000000e+00            17            6.0            0
10855  4.702610e+06  0.000000e+00            14            6.1      -700000
10856  0.000000e+00  0.000000e+00            14            5.7            0
10857  0.000000e+00  0.000000e+00            10            5.9            0
10858  0.000000e+00  0.000000e+00            11            5.5            0
10859  0.000000e+00  0.000000e+00            22            6.6            0
10860  0.000000e+00  0.000000e+00            13            7.0            0
10861  0.000000e+00  0.000000e+00            11            7.4            0
10862  0.000000e+00  0.000000e+00            20            5.7            0
10863  0.000000e+00  0.000000e+00            11            6.5            0
10864  0.000000e+00  0.000000e+00            22            5.4            0
10865  1.276423e+05  0.000000e+00            15            1.5       -19000

[10865 rows x 12 columns]
```

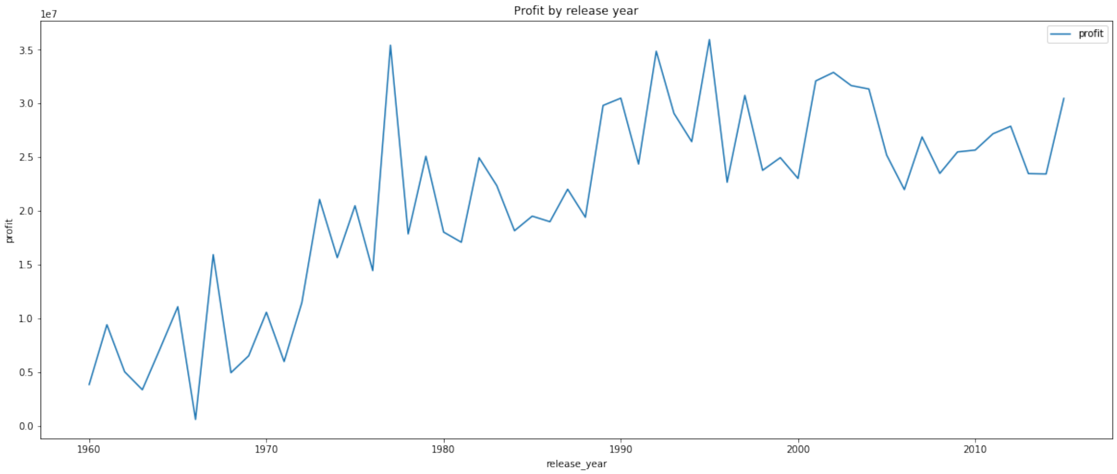
## Exploratory Data Analysis

Tip: Now that you've trimmed and cleaned your data, you're ready to move on to exploration. **Compute statistics** and **create visualizations** with the goal of addressing the research questions that you posed in the Introduction section. You should compute the relevant statistics throughout the analysis when an inference is made about the data. Note that at least two or more kinds of plots should be created as part of the exploration, and you must compare and show trends in the varied visualizations.

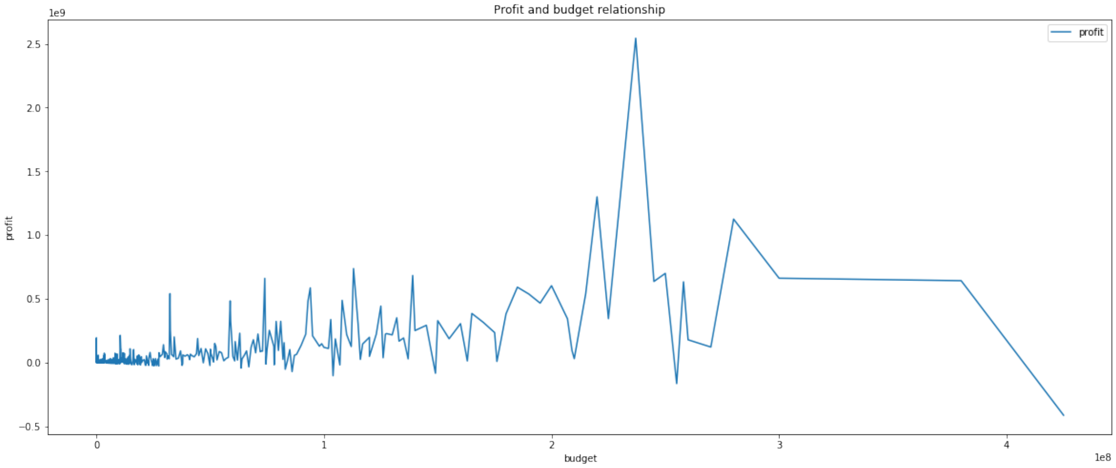
Tip: - Investigate the stated question(s) from multiple angles. It is recommended that you be systematic with your approach. Look at one variable at a time, and then follow it up by looking at relationships between variables. You should explore at least three variables in relation to the primary question. This can be an exploratory relationship between three variables of interest, or looking at how two independent variables relate to a single dependent variable of interest. Lastly, you should perform both single-variable (1d) and multiple-variable (2d) explorations.

1.1.5 Research Question 1 (How the profit is affected by the revenue, budget, popularity, runtime)

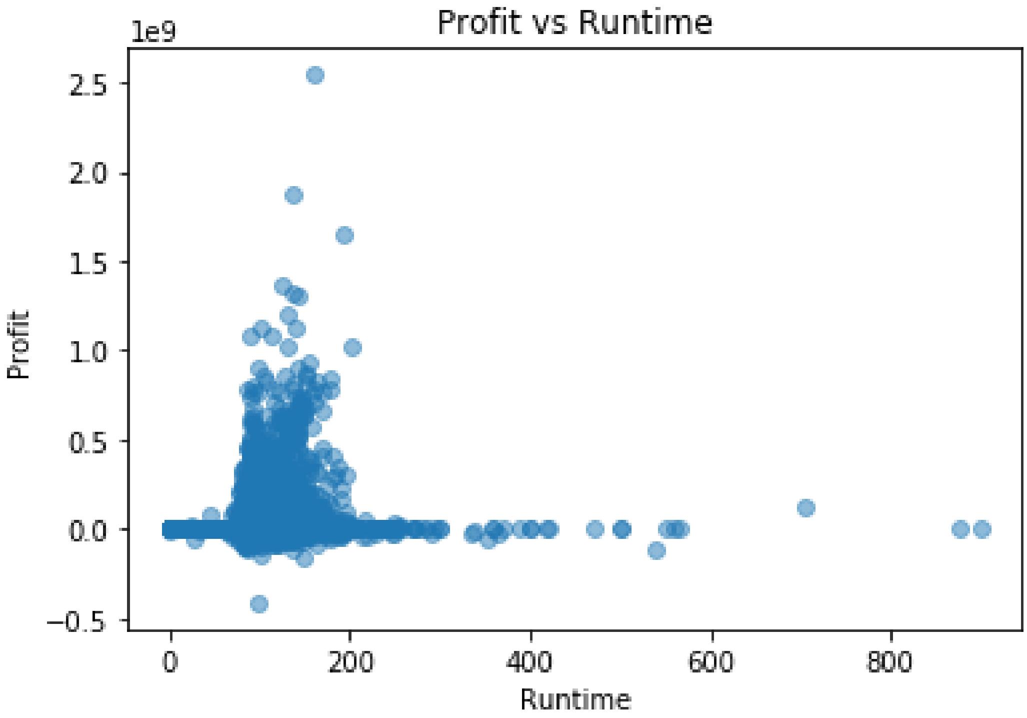
```
In [11]: #A line plot of the profit and release year
         release_yr_rev = (df[df['profit'].notnull()][['release_year', 'profit']].groupby('relea
         release_yr_rev.plot(figsize=(20,8))
         plt.xlabel('release_year')
         plt.ylabel('profit')
         plt.title('Profit by release year')
         plt.show()
```



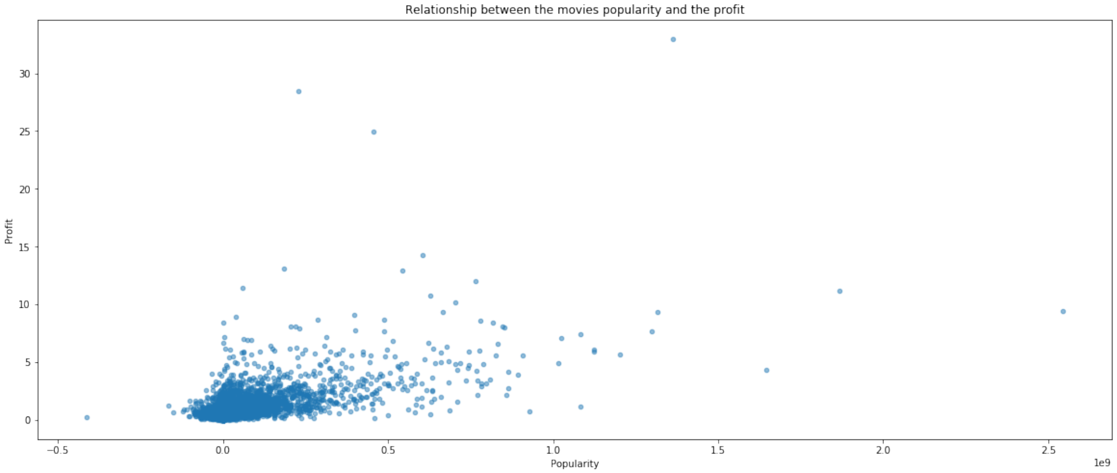
```
In [12]: #A line plot of the profit and budget
         budget_rev = (df[df['profit'].notnull()][['budget', 'profit']].groupby('budget').mean()
         budget_rev.plot(figsize=(20,8))
         plt.xlabel('budget')
         plt.ylabel('profit')
         plt.title('Profit and budget relationship')
         plt.show()
```



```
In [13]: #Relationship between Profit and Runtime
         plt.xlabel('Runtime')
         plt.ylabel('Profit')
         plt.title('Profit vs Runtime')
         plt.scatter(df['runtime'], df['profit'], alpha=0.5)
         plt.show()
```

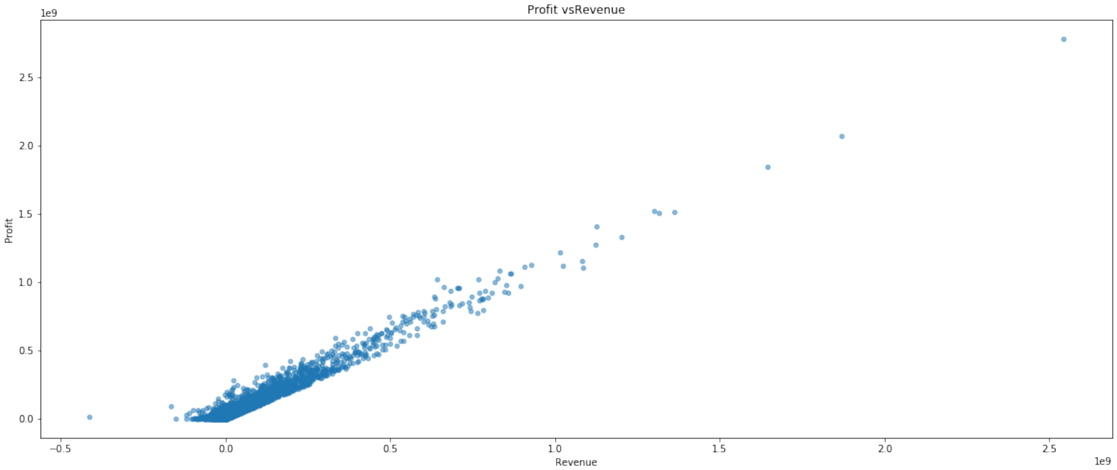


```
In [14]: #Investigate the relationship between the movies popularity and the profit.
         df.plot(x='profit', y='popularity', kind='scatter', figsize=(20,8), alpha=0.5)
         plt.xlabel('Popularity')
         plt.ylabel('Profit')
         plt.title('Relationship between the movies popularity and the profit')
         plt.show()
```



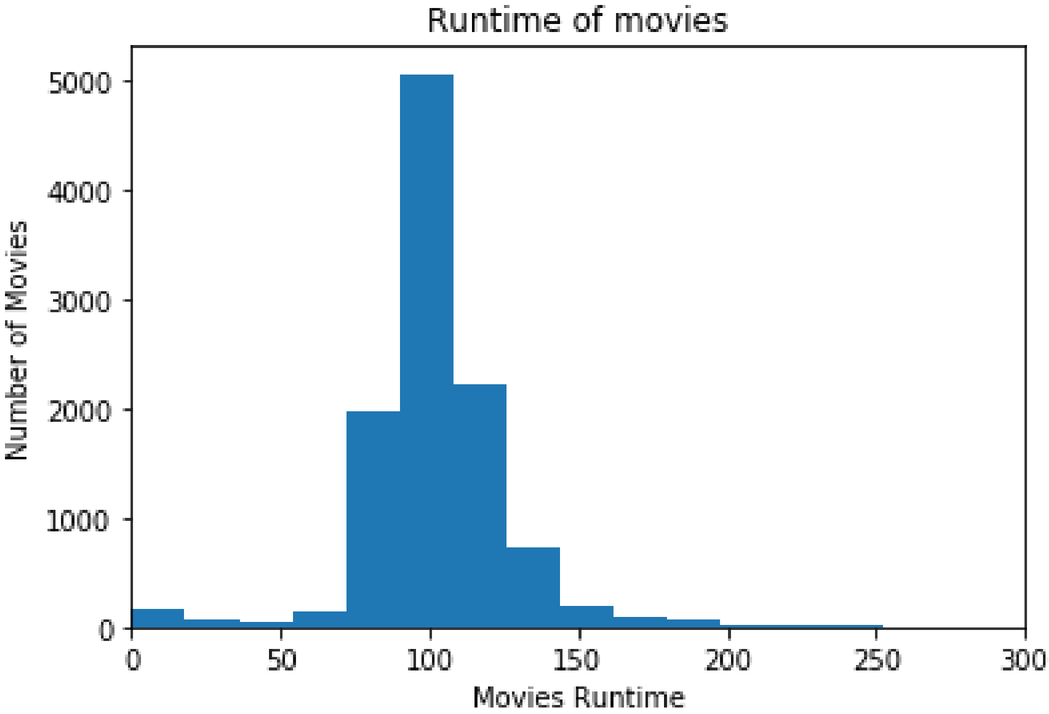
In [15]:

```
#Relationship between Profit and Revenue
df.plot(x='profit', y='revenue', kind='scatter', figsize=(20,8), alpha=0.5)
plt.xlabel('Revenue')
plt.ylabel('Profit')
plt.title('Profit vsRevenue')
plt.show()
```



In [16]:

```
#Runtime of movies
plt.xlabel('Movies Runtime')
plt.ylabel('Number of Movies')
plt.title('Runtime of movies')
plt.hist(df['runtime'], bins = 50);
plt.xlim(0, 300);
```



1.1.6 Research Question 2 (Movie with the highest/lowest profit/budget/revenue!)

```
In [17]: # Function to find maximum value of a column
         def max(column):
             return df.loc[df[column].idxmax()]
```

```
In [18]: #Movie with highest profit
         max('profit')
```

```
Out[18]: popularity            9.43277
         id                      19995
         budget              237000000
         revenue            2781505847
         runtime                   162
         original_title         Avatar
         release_year             2009
         budget_adj        2.40887e+08
         revenue_adj       2.82712e+09
         vote_count               8458
         vote_average              7.1
         profit             2544505847
         Name: 1386, dtype: object
```

```
In [19]: #Movie with highest revenue
         max('revenue')
```

```
Out[19]: popularity            9.43277
         id                      19995
         budget              237000000
         revenue            2781505847
         runtime                   162
         original_title         Avatar
         release_year             2009
         budget_adj        2.40887e+08
         revenue_adj       2.82712e+09
         vote_count               8458
         vote_average              7.1
         profit             2544505847
         Name: 1386, dtype: object
```

```
In [20]: #Movie with highest budget
         max('budget')
```

```
Out[20]: popularity                  0.25054
         id                            46528
         budget                    425000000
         revenue                    11087569
         runtime                         100
         original_title    The Warrior's Way
         release_year                   2010
         budget_adj                 4.25e+08
         revenue_adj             1.10876e+07
         vote_count                       74
         vote_average                    6.4
         profit                   -413912431
         Name: 2244, dtype: object
```

```
In [21]: # Function to find minimum value of a column
         def min(column):
             return df.loc[df[column].idxmin()]
```

```
In [22]: #Movie with lowest profit
         min('profit')
```

```
Out[22]: popularity                  0.25054
         id                            46528
         budget                    425000000
         revenue                    11087569
         runtime                         100
         original_title    The Warrior's Way
         release_year                   2010
         budget_adj                 4.25e+08
         revenue_adj             1.10876e+07
         vote_count                       74
         vote_average                    6.4
```

```
        profit                   -413912431
        Name: 2244, dtype: object
```

```
In [23]: #Movie with lowest revenue
         min('revenue')
```

```
Out[23]: popularity          2.93234
         id                   265208
         budget             30000000
         revenue                   0
         runtime                  92
         original_title    Wild Card
         release_year           2015
         budget_adj         2.76e+07
         revenue_adj               0
         vote_count              481
         vote_average            5.3
         profit            -30000000
         Name: 48, dtype: object
```

```
In [24]: #Movie with lowest budget
         min('budget')
```

```
Out[24]: popularity            3.92733
         id                     280996
         budget                      0
         revenue              29355203
         runtime                   103
         original_title     Mr. Holmes
         release_year             2015
         budget_adj                  0
         revenue_adj       2.70068e+07
         vote_count                425
         vote_average              6.4
         profit               29355203
         Name: 30, dtype: object
```

1.1.7 Research Question 3 (Find the average budget/revenue/profit/runtime/popularity of all movies)

```
In [25]: # Function to find average of a column
         def average(column):
             return df[column].mean()
```

```
In [26]: #Find the average budget of all movies
         average('budget')
```

```
Out[26]: 14625701.094146879
```

```
In [27]: #Find the average revenue of all movies
         average('revenue')

Out[27]: 39823319.793392234

In [28]: #Find the average profit of all movies
         average('profit')

Out[28]: 25197618.699245352

In [29]: #Find the average runtime of all movies
         average('runtime')

Out[29]: 102.07086324314375

In [30]: #Find the average popularity of all movies
         average('popularity')

Out[30]: 0.64644095196024287
```

## Conclusions 1. Movies around 200 minutes are more profitable. 2. Most movies have a runtime of 100 minutes. 3. There is a steady increase in profit over the years. 4. There is a strong relationship between profit and revenue. 5. Avatar has the highest profit and revenue. 6. The warriors way has the highest budget and the lowest profit. 7. Wild Card has the lowest revenue. 8. Mr Holmes has the lowest budget. 9. Average budget of all movies is 14,625,701. 10. Average revenue ofall moviesis 39,823,320. 11. Average profit of all movies is $25,197,619. 12. Average runtime of all movies is 102 minutes. 13. Average popularity of all movies is 0.65

Limitation: I left out some columns that could have an effect on the conclusion during the analysis.

> **Tip**: If you haven't done any statistical tests, do not imply any statistical conclusions. And make sure you avoid implying causation from correlation!

> **Tip**: Once you are satisfied with your work here, check over your report to make sure that it is satisfies all the areas of the rubric (found on the project submission page at the end of the lesson). You should also probably remove all of the "Tips" like this one so that the presentation is as polished as possible.

1.2 Submitting your Project

> **Tip**: Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

> **Tip**: Alternatively, you can download this report as .html via the **File** > **Download as** submenu, and then manually upload it into the workspace directory by clicking on the orange Jupyter icon in the upper left, then using the Upload button.

Tip: Once you've done this, you can submit your project by clicking on the "Submit Project" button in the lower right here. This will create and submit a zip file with this .ipynb doc and the .html or .pdf version you created. Congratulations!

```
In [31]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])

Out[31]: 255

In [ ]:
```