Data-Analysis2

October 7, 2024

1 Data Analysis Example

2 Superhero Movies

- Highest BO Gross
- Best opening weekend attendance.
- Engineering the columns we need to answer the questions!

[1]:
```
import pandas as pd

sh = pd.read_csv("https://raw.githubusercontent.com/mafudge/datasets/master/
 ↪superhero/superhero-movie-dataset-1978-2012-header.csv")
sh = sh.dropna()
sh.head(10)
```

[1]:

	Year	Title	Comic	IMDB Score	RT Score	\
0	1978	Superman	DC	7.3	95	
1	1980	Superman II	DC	6.7	88	
3	1983	Superman III	DC	4.9	24	
4	1984	Supergirl	DC	4.2	8	
5	1986	Howard the Duck	Marvel	4.3	16	
6	1987	Superman IV: The Quest for Peace	DC	3.6	10	
7	1989	Batman	DC	7.6	71	
10	1992	Batman Returns	DC	7.0	78	
11	1995	Batman Forever	DC	5.4	42	
12	1997	Batman & Robin	DC	3.6	12	

	Composite Score	Opening Weekend	Box Office	Avg Ticket Price	\
0	84.0		7465343.0	2.34	
1	77.5		14100523.0	2.69	
3	36.5		13352357.0	3.15	
4	25.0		5738249.0	3.36	
5	29.5		5070136.0	3.71	
6	23.0		5683122.0	3.91	
7	73.5		40489746.0	3.97	
10	74.0		45687711.0	4.15	
11	48.0		52784433.0	4.35	
12	24.0		42872605.0	4.59	

```
    Opening Weekend Attendance  US Population That Year
0                 3.190318e+06                222584545
1                 5.241830e+06                227224681
3                 4.238843e+06                233791994
4                 1.707812e+06                235824902
5                 1.366613e+06                240132887
6                 1.453484e+06                242288918
7                 1.019893e+07                246819230
10                1.100909e+07                255029699
11                1.213435e+07                262803276
12                9.340437e+06                267783607
```

Let us see how many people went to the movie on opening weekend. We create a new column 'pct_of_pop' in the DataFrame sh. This column represents the percentage of the U.S. population (for a given year) that attended the movie on its opening weekend. The calculation divides the 'Opening Weekend Attendance' by the 'US Population That Year'. We then select specific columns ('Year', 'Title', 'Comic', 'Opening Weekend Attendance', 'US Population That Year', and 'pct_of_pop') and displays the first five rows with the .head() method.

[9]:
```
# feature engineering: percentage of population seeing the movie.
sh['pct_of_pop'] = sh['Opening Weekend Attendance'] /sh['US Population That␣
 ↪Year']
sh[['Year','Title','Comic','Opening Weekend Attendance', 'US Population That␣
 ↪Year', "pct_of_pop"]].head()
```

[9]:
```
   Year            Title   Comic  Opening Weekend Attendance  \
0  1978         Superman      DC                  3190317.521
1  1980      Superman II      DC                  5241830.112
3  1983     Superman III      DC                  4238843.492
4  1984        Supergirl      DC                  1707812.202
5  1986  Howard the Duck  Marvel                  1366613.477

   US Population That Year  pct_of_pop
0                222584545    0.014333
1                227224681    0.023069
3                233791994    0.018131
4                235824902    0.007242
5                240132887    0.005691
```

Let us sort the DataFrame based on the 'pct_of_pop' column in ascending order using .sort_values("pct_of_pop"). We get the top 5 attendance using the .tail() method

[12]:
```
# 5 most popular, when normalized by attendance.
sh[['Year','Title','Comic','Opening Weekend Attendance', 'US Population That␣
 ↪Year', "pct_of_pop"]]\
```

```
    .sort_values("pct_of_pop").tail(5)
```

[12]:

	Year	Title	Comic	Opening Weekend Attendance	US Population That Year	pct_of_pop
46	2012	The Dark Knight Rises	DC	20314052.40	314055984	0.064683
17	2002	Spider-Man	Marvel	19766629.26	287803914	0.068681
33	2008	The Dark Knight	DC	22062880.64	304374846	0.072486
32	2007	Spider-Man 3	Marvel	21964609.88	301579895	0.072832
45	2012	Marvel's The Avengers	Marvel	26191756.06	314055984	0.083398

sh['box_off_estimate'] = sh['Opening Weekend Box Office'] * sh['Avg Ticket Price'] calculates an estimated box office revenue for each entry in the sh DataFrame. Here's a breakdown:

- A new column 'box_off_estimate' is created.
- we multiply the 'Opening Weekend Box Office' by the 'Avg Ticket Price', to estimate total box office revenue for the opening weekend. #### This would give us an estimate of total revenue, assuming all ticket sales follow the average ticket price.

[29]:
```
sh['box_off_estimate'] = sh['Opening Weekend  Box Office'] * sh['Avg Ticket␣
 ↪Price']
```

[30]:
```
#Titles with highest opening_weeked_box office estimate (without inflation)
sh.sort_values("box_off_estimate").tail()
```

[30]:

	Year	Title	Comic	IMDB Score	RT Score	Composite Score	Opening Weekend Box Office	Avg Ticket Price	Opening Weekend Attendance	US Population That Year	pct_of_pop
39	2010	Iron Man 2	Marvel	7.1	74	72.5	128122480.0	7.89	16238590.62	308745538	0.052595
32	2007	Spider-Man 3	Marvel	6.3	63	63.0	151116516.0	6.88			
33	2008	The Dark Knight	DC	8.9	94	91.5	158411483.0	7.18			
46	2012	The Dark Knight Rises	DC	9.1	86	88.5	160887295.0	7.92			
45	2012	Marvel's The Avengers	Marvel	8.7	92	89.5	207438708.0	7.92			

```
32                21964609.88                301579895    0.072832
33                22062880.64                304374846    0.072486
46                20314052.40                314055984    0.064683
45                26191756.06                314055984    0.083398

    box_off_estimate
39      1.010886e+09
32      1.039682e+09
33      1.137394e+09
46      1.274227e+09
45      1.642915e+09
```

The code sh.to_csv("superhero2.csv", header=True, index=False) performs the following :

- It saves the DataFrame sh to a CSV file named superhero2.csv.
- The header=True argument ensures that the column names are written as the first row in the CSV file.
- The index=False argument prevents pandas from writing row indices to the file, so the resulting CSV won't have an extra index column. #### This will create a clean CSV file with just the data and column headers.

[32]:
```
# we've done some work. Let's save the dataset again so we don't have to␣
↪rengineer things
sh.to_csv("superhero2.csv", header=True, index=False)
```

[ ]:
```

```