BANA212 FINAL GROUP REPORT

Web Scraping & Avenging:

# Predicting the Upcoming Avengers Endgame Movie Details

Ahnni Wang • Daria Bedareva • Javier Orraca • Joel Dayrit • Xiaoting Sun

## Table of Contents:

# I. Introduction

Avengers Endgame is the culmination of a little over a decade of movies for Marvel Studios' Marvel Cinematic Universe. It represents the resolution for the primary characters of the franchise thus far. With gross revenues totaling upwards of 18 billion in the global box office, 2 billion of which can be attributed to the preceding Avengers movie 'Infinity War,' expectations for 'Endgame' are at an all-time high. Scheduled for release this April, and with a two MCU films directly preceding it (Antman & The Wasp and Captain Marvel), we feel that 'Endgame' represents an ideal scenario to utilize the programming and analytical skills learned and discussed in class.

Utilizing Python, web scraping techniques and other programing tools, it is our goal to predict the likely outcome of details of Avengers Endgame. We specifically want to attempt to predict a likely rating and gross revenue based on regression and other extrapolation methods from the data of the preceding films noting potential dependent and independent variables based on similar factors such as cast, director, month of release, total film budget, and others. Additionally, we aim to explore the data from the previous films visually representing them in order to see any potential patterns that can help in the prediction of the outcome of the of Endgame.

# II. Searching for Data Sources

While there are numerous data sources available (and most of them online), we decided to focus on information gathered from 3 main sources listed below:

| | |
|---|---|
| **1. IMDB** | "IMDb (Internet Movie Database) is an online database of information related to films, television programs, home videos and video games, and internet streams, including cast, production crew and personnel biographies, plot summaries, trivia, and fan reviews and ratings. An additional fan feature, message boards, was abandoned in February 2017. Originally a fan-operated website, the database is owned and operated by IMDb.com, Inc., a subsidiary of Amazon. As of October 2018, IMDb has approximately 5.3 million titles (including episodes) and 9.3 million personalities in its database, as well as 83 million registered users."[1] |
| **2. Rotten Tomatoes** | "Rotten Tomatoes is an American review-aggregation website for film and television. The company was launched in August 1998 by three undergraduate students at the University of California, Berkeley: Senh Duong, Patrick Y. Lee, and Stephen Wang. The name "Rotten Tomatoes" derives from the practice of audiences throwing rotten |

[1] https://en.wikipedia.org/wiki/IMDb

| | |
|---|---|
| | tomatoes when disapproving of a poor stage performance. Since January 2010, Rotten Tomatoes has been owned by Flixster, which was in turn acquired by Warner Bros. in 2011. In February 2016, Rotten Tomatoes and its parent site Flixster were sold to Comcast's Fandango. Warner Bros. retained a minority stake in the merged entities, including Fandango."[2] |
| **3. Metacritic (Metascore)** | "Metacritic is a website that aggregates reviews of media products: films, TV shows, music albums, video games, and formerly, books. For each product, the scores from each review are averaged (a weighted average). Metacritic was created by Jason Dietz, Marc Doyle, and Julie Doyle Roberts in 1999. The site provides an excerpt from each review and hyperlinks to its source. A color of green, yellow or red summarizes the critics' recommendations. It has been described as the video game industry's "premier" review aggregator. Metacritic's scoring converts each review into a percentage, either mathematically from the mark given, or which the site decides subjectively from a qualitative review. Before being averaged, the scores are weighted according to the critic's fame, stature, and volume of reviews."[3] |

The sites were primarily selected for their extensive and diverse rating systems. As each site had a specific audience and user base that are in some ways, classified differently from each other, we aimed to get the overall ratings for the movies based on an average of the three sites ratings and details per movie listed within the Marvel Cinematic Universe. Additionally, we opted to exclude other sites as several of them lacked the complete movie listings or had an overall low number of user engagement required for any potential regression analysis that would be conducted.

[2] https://en.wikipedia.org/wiki/Rotten_Tomatoes

[3] https://en.wikipedia.org/wiki/Metacritic

## III. Web Scraping and Understanding HTML Structure

While each of the 3 main website sources selected contained the necessary information, each website was formatted differently. Some were plainly encoded while others required us to dig deep into nestled areas within layers of CSS code. The following details how we were about to scrape the data from the web source using the *BeautifulSoup* package in Pyhton.

For the Internet Movie Database (IMDB), in order to simplify the scraping, we wanted to find a page that combined all of the Marvel Studio movies in a single page. Aside from simplicity, this would allow us to ensure that the movies were all listed within the site. After a series of queries exploring the IMDB website, we were able to find that it was possible to have a single result page cite all the Marvel Studio movies: [https://www.imdb.com/list/ls026690821/](https://www.imdb.com/list/ls026690821/). This included the first movie of the Marvel Cinematic Universe 2008's 'Iron Man' up to future Marvel movies that have yet to be released. Currently though, we at this stage, would like to discover the data specifically from Iron Man (2008) to Avengers: Endgame (2019). To begin our web scraping, we imported requests and used get() function to requests the contents of the web page.

```
Import requests
url = 'https://www.imdb.com/list/ls026690821/'
resp = requests.get(url)
```

Afterwards, we imported the *BeautifulSoup* from the bs4 package and converted the request into a text file. As can be seen, the server returned an HTML page as a response.

```
From bs4 import BeautifulSoup
soup = BeautifulSoup(resp.text)
```

As the information points we intended to scrape included the movie titles, year-released, IMDB ratings, votes, and gross from the website, we viewed the specific page using Google's Chrome browser and utilizing the developer tools and right clicking 'inspect' to check where the specific content was stored in the code as viewed through the developers page. We created a container to scrape the information of the first Marvel movie – "Iron Man." Additionally, we learned that information about the movie was stored under a 'div' file named 'lister-item-content'. As a result, the following Python code was executed:

```
movie_titles = soup.find_all('div', class_ = 'lister-item-content')
print(type(movie_titles))
print(len(movie_titles))
```

By printing the type and length of the container – 'movie_titles', we observe that the result displayed 29 bs4.element.Resultset, which means there are 29 similar containers as the first movie. We then scraped the information we needed for the first movie and wrote a for loop to scrape all the other movies we would like to discover similarly.

```
First_movie = movie_titles[0]
First_movie
```

The code above returned the content of the first container.  However, the HTML file remained too long for us to read the information. Therefore, we endeavored to scrape movie titles, year-released, and other information by steps.  First, we found that the movie title was specifically stored under an ‘h3’ file, and therefore tried to extract the information under this tag.

```
First_movie.h3
```

And the result shows:

```
<h3 class=”lister-item-header”>
    <span class=”lister-item-index unbold text-primary”>1.</span>
      <a href=”/title/tt0371746/?ref_=ttls_li_tt”>Iron Man</a>
   <span class=”lister-item-year text-muted unbold”>(2008)</span>
</h3>
```

To extract the movie title:

```
first_name = first_movie.h3.a.text
```

To extract the year-released:

```
first_year = first_movie.h3.find(‘span’, class_ = ‘lister-item-year text-muted unbold’)
first_year = first_year.text[1:5]
```

Next, we would wanted to output the specific rating of the movie. Unfortunately, the information was not stored under the specific ‘h3’ tag and therefore necessitated us to return to the ‘div’ file.

To find the IMDB ratings:

```
first_rate = first_movie.div.find(‘span’, class_ = ‘ipl-rating-star__rating’).text
first_rate = float(first_rate)
```

To find the number of votes:

```
first_votes = first_movie.find(‘span’, attrs = {‘name’:’nv’})
first_v = first_votes[‘data-value’]
first_votes = int(first_v)
```

To find the gross box office:

```
first_gross = int(first_g)
```

After finding all the information we wanted to extract, we created a for-loop, the first step of which was to declare some list that could be used to append data in them.

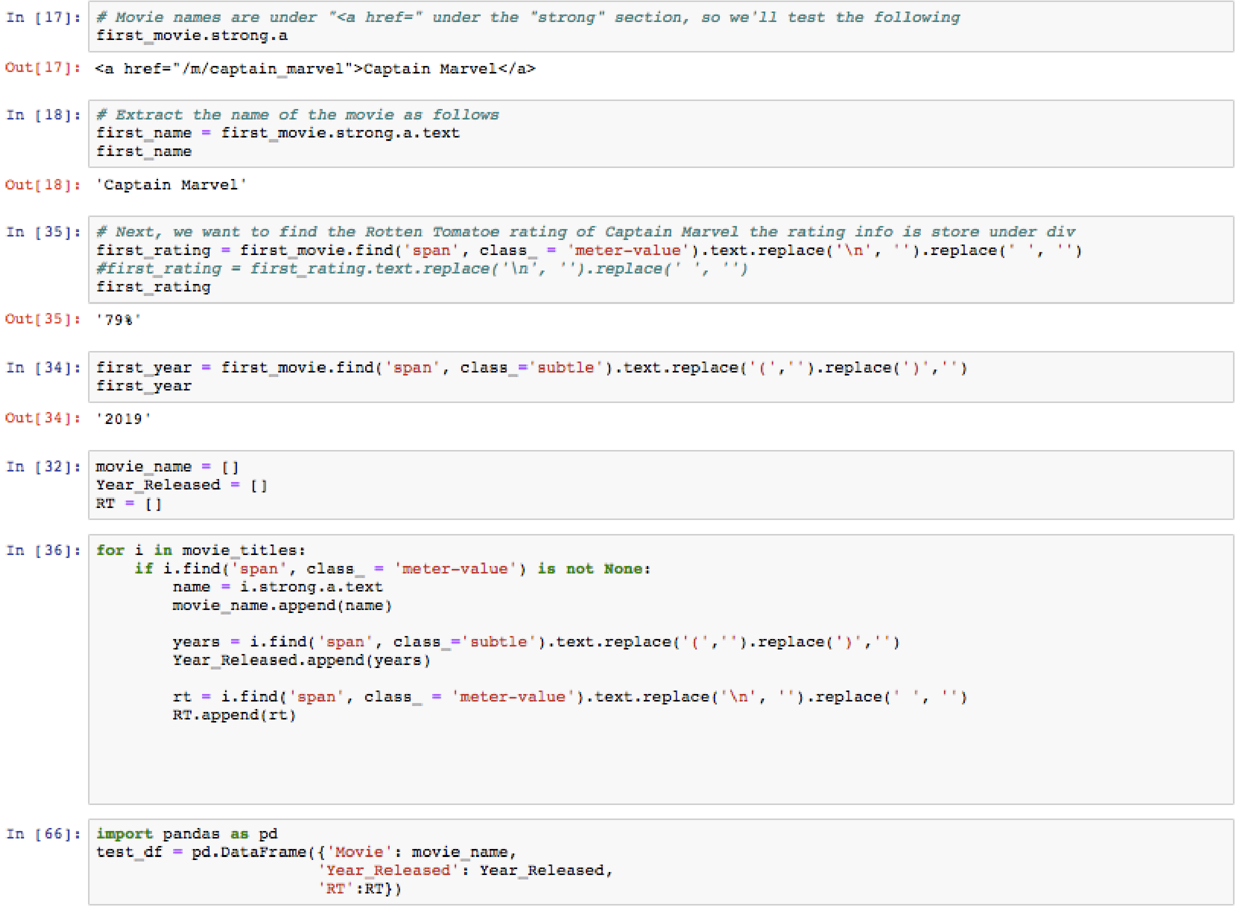
```
Movie_name = []
years = []
imdb_ratings = []
metascore = []
votes = []
gross = []
```

In this webpage, there are several movies that haven’t been released. In order to scrape the movies that have already released, we generated a for-loop and avoid those without a Metascore on the IMDB webpage.

```
For I in movie_titles:
  if i.find(‘div’, class_ = ‘metascore favorable’) is not None:
    # The name
    name = i.h3.a.text
    movie_name.append(name)
    # The year
        year = i.h3.find(‘span’, class_ = ‘lister-item-year text-muted unbold’).text
    years.append(year)
    # The IMDB rating
         rates = i.find(‘span’, class_= ‘ipl-rating-star__rating’).text
    imdb_ratings.append(float(rates))
    # The Metascore
       m_score = i.find(‘span’, class_ = ‘metascore favorable’).text
    metascore.append(int(m_score))
    # The number of votes
       vote = i.find(‘span’, attrs = {‘name’:’nv’})[‘data-value’]
    votes.append(vote)
    #The number of gross
        grossv = i.find_all(‘span’, attrs = {‘name’:’nv’})[1][‘data-value’]
    grossv =locale.atoi(grossv)
    gross.append(int(grossv))
```

After scraping all the movie data from the IMDB website, the information was collated into a single dataframe and then exported as a .csv file to use for data visualization.

Similar to what was done for scraping the IMDB site, *BeautifulSoup* was also utilized to scrape the information from Rotten Tomatoes. The webpage specifically targeted for this was the “Marvel Cinematic Universe” sub section in Rotten Tomatoes. However, as the page contained both the movies and television shows, certain modifications were required to output only the target information desired. As we were only adding additional data onto the existing exported database created, only the movie titles, year-released and the ratings from this site would be scraped and the output used to append the existing file already created from the previous scaping of the IMDB website, the process of which is detailed below:



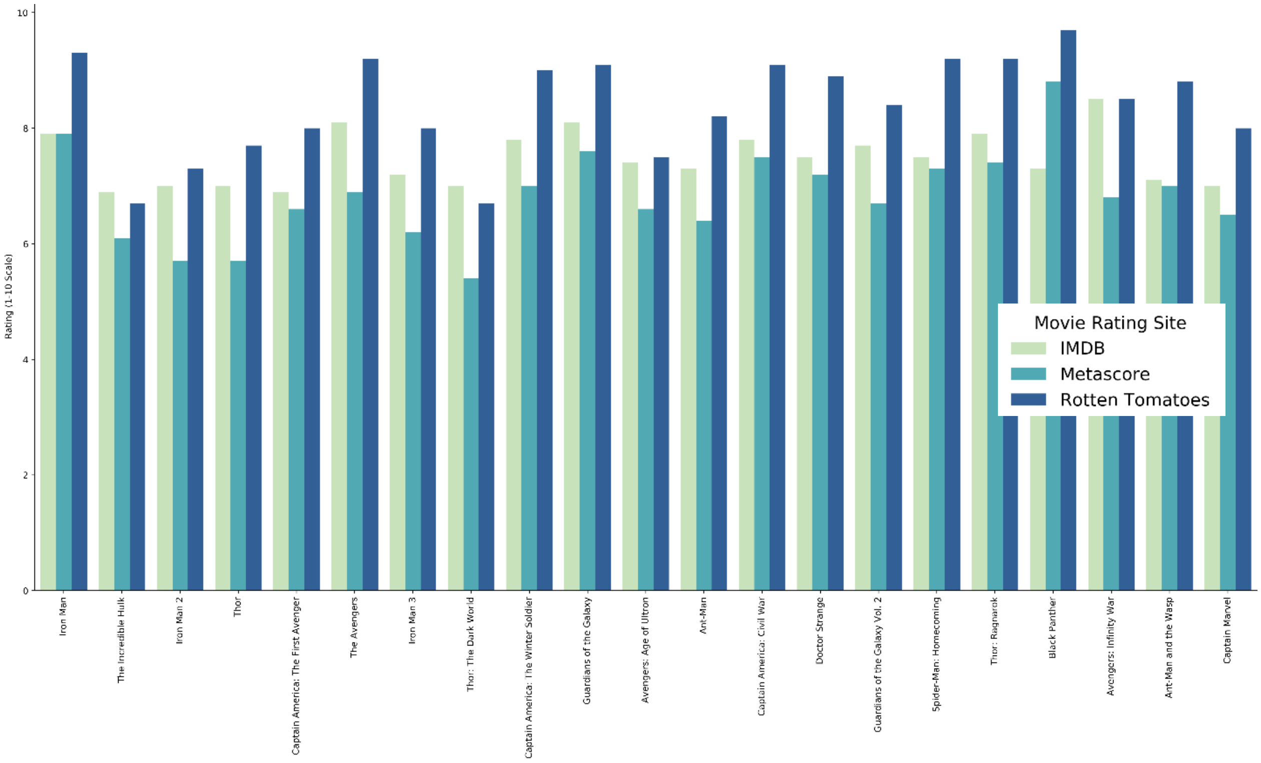
```
In [17]: # Movie names are under "<a href=" under the "strong" section, so we'll test the following
         first_movie.strong.a

Out[17]: <a href="/m/captain_marvel">Captain Marvel</a>

In [18]: # Extract the name of the movie as follows
         first_name = first_movie.strong.a.text
         first_name

Out[18]: 'Captain Marvel'

In [35]: # Next, we want to find the Rotten Tomatoe rating of Captain Marvel the rating info is store under div
         first_rating = first_movie.find('span', class_ = 'meter-value').text.replace('\n', '').replace(' ', '')
         #first_rating = first_rating.text.replace('\n', '').replace(' ', '')
         first_rating

Out[35]: '79%'

In [34]: first_year = first_movie.find('span', class_='subtle').text.replace('(','').replace(')','')
         first_year

Out[34]: '2019'

In [32]: movie_name = []
         Year_Released = []
         RT = []

In [36]: for i in movie_titles:
             if i.find('span', class_ = 'meter-value') is not None:
                 name = i.strong.a.text
                 movie_name.append(name)

                 years = i.find('span', class_='subtle').text.replace('(','').replace(')','')
                 Year_Released.append(years)

                 rt = i.find('span', class_ = 'meter-value').text.replace('\n', '').replace(' ', '')
                 RT.append(rt)

In [66]: import pandas as pd
         test_df = pd.DataFrame({'Movie': movie_name,
                                 'Year_Released': Year_Released,
                                 'RT':RT})
```

Due to a nuance in the formatting of the data, we were able to determine that while the above code still generated data from both the movies and television series, it was observed that the year-released of TV series are shown specifically formatted as "TV Show, 2018 –". Conversely, for year-released in movies, the data is shown as simply "2018". With this insight, it was possible to calculate the length of year-released for each row and drop the rows whose year-released length were found to be longer than 4 digits. In the end, it was possible to appended the Avengers: Endgame in the last row by adding: ({'Movie': 'Avengers: Endgame', 'Year_Released': 2019}, ignore_index = True) and save the data frame to an CSV file.

## IV. Analysis through Visualizations

After scraping the Marvel cinematic universe ratings from IMDB, Rotten Tomatoes, and Metascore, we began exploring the aggregated data frame through visualizations. Given that ratings from each movie review had unique scales (e.g., 1-5, 1-10, and 1-100), we created a consistent scale for comparison from 1 to 10. Additionally, as the ratings were in different columns (by review site), we manipulated the data frame using the pandas melt function to list all of the movie rating sources in one column. This pre-processing step would help us to visualize the data into a bar chart via matplotlib and seaborn, including Source as a "hue" or "color" criteria, see below:

**Table 4.1 – Movie Rating by Source**



The bar chart above is not easily legible given all the bars. As such, the mean movie rating was calculated and plotted to more easily interpret the results and identify the highest and lowest rated Marvel movies (see Table 4.2 on the top of the next page). The results of the simplified bar chart show that Black Panther and Iron Man were the highest rated Marvel movies, and Thor: The Dark World and The Incredible Hulk were the lowest rated.

To better understand the story of how and why we keep seeing endless Marvel movies from Hollywood, we ran three preliminary, univariate linear regressions using release date as the predictor variable and movie production budget, US box office revenues, and worldwide box office revenues as the target variables (amounts shown in USD billions). Table 4.3 displays the results of our preliminary regression analyses. While production budget has been marginally increasing, US and especially the worldwide box office revenues have significantly outperformed the incremental production budgets, see below.

**Table 4.2 – Average Movie Rating**

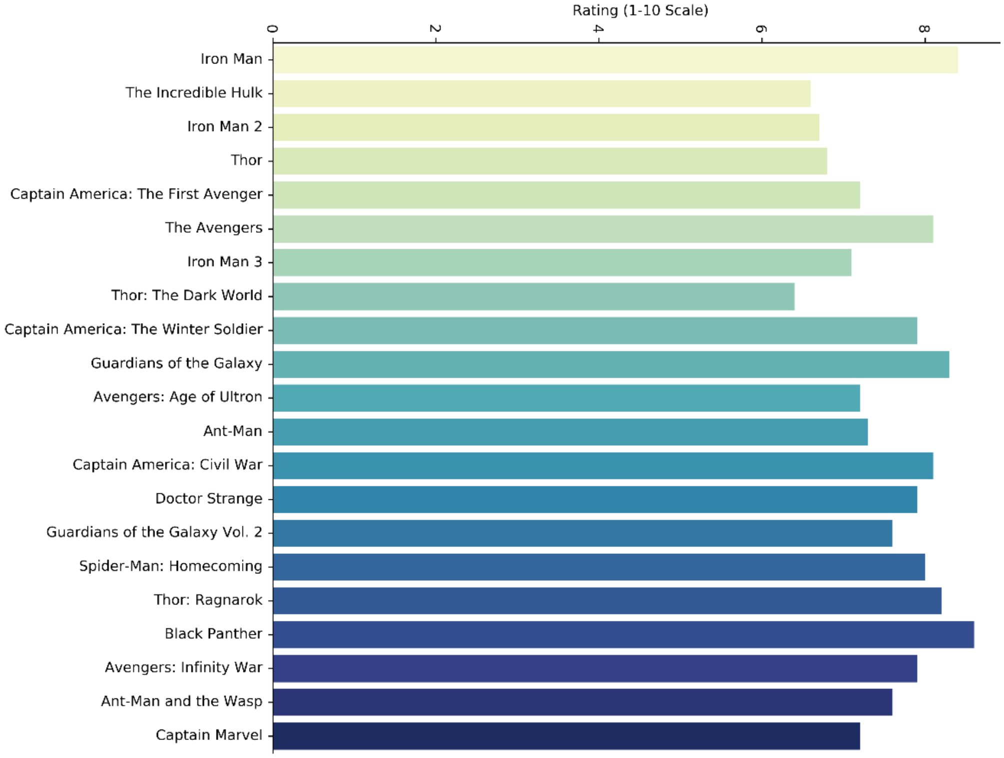
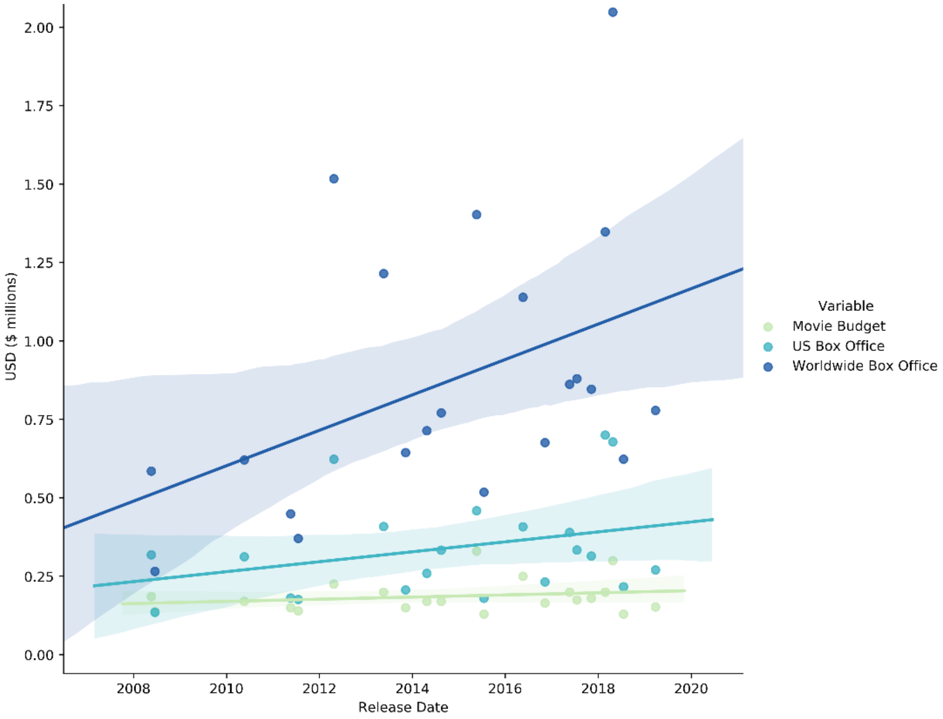


**Table 4.3 – Preliminary Regressions (for visualization purposes only)**



To further analyze the US and worldwide box office revenues via visualizations, we created "joint plots" using seaborn that consist of kernel density estimation plots ("KDE") and contour plots. See Tables 4.4 and 4.5 for the joint plots.

**Table 4.4 – US Box Office KDE**

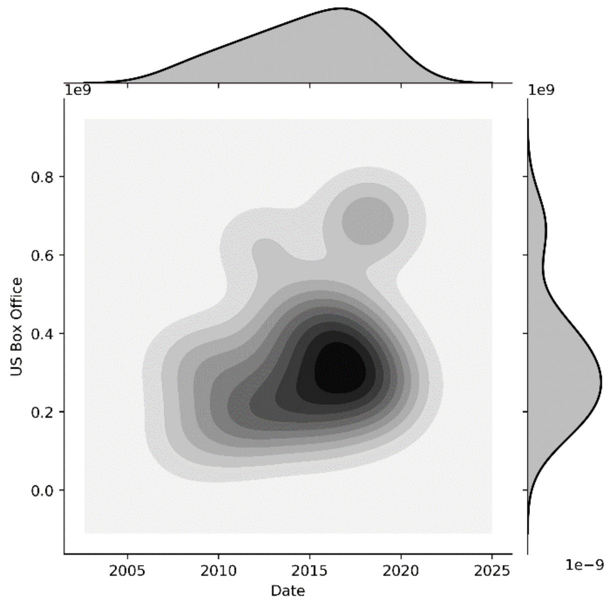
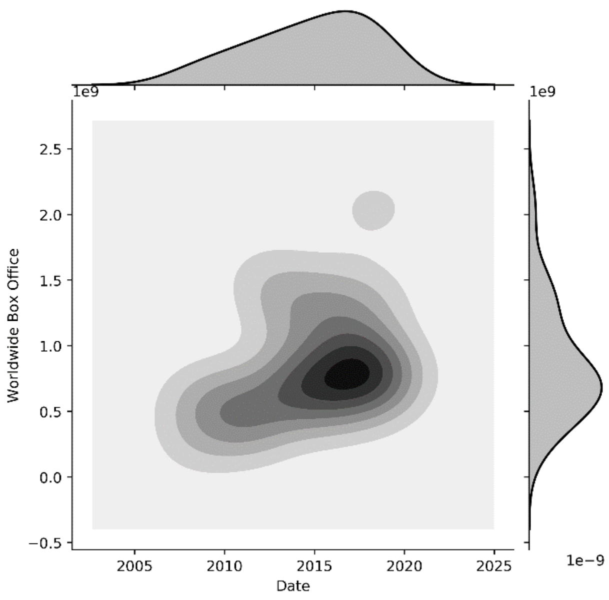


**Table 4.5 – Worldwide Box Office KDE**



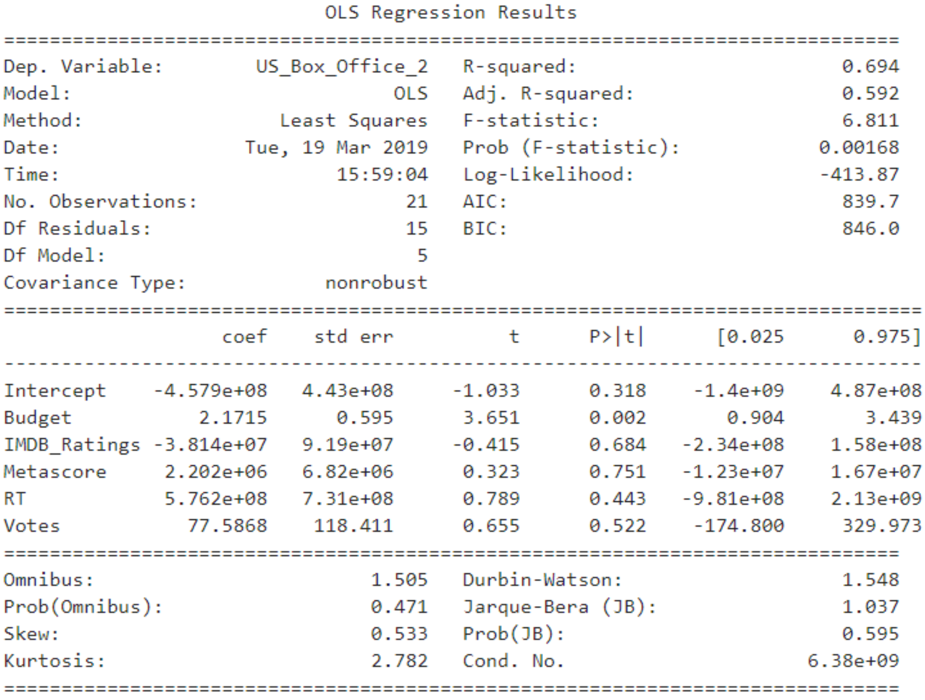
The most basic explanation[4] of KDE reduces this estimation methodology as follows:

- KDE is a non-parametric way to estimate the probability density function of a random variable, and
- KDE is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample

[4] Source: https://en.wikipedia.org/wiki/Kernel_density_estimation

## V. Regression Analysis and Overall Predictions

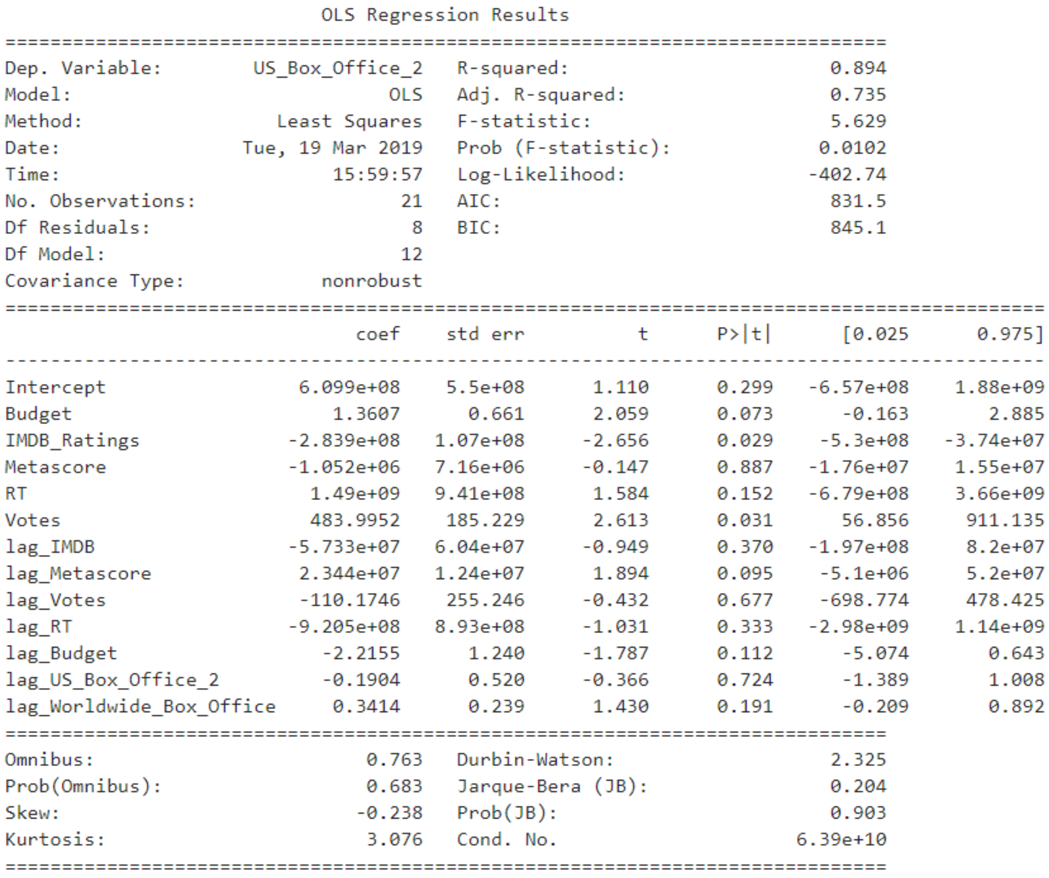
The final step of our project was to perform regression analyses on the movie data that was scraped from IMDB, Rotten Tomatoes, and Metascore. We relied on the statsmodel library for Python to check the linearity of the data (using scatter plots of the main variables of interest) and to make sure that the linear regression method could be applied. As a result, several specifications of ordinary least squares multivariate regressions were performed, and the output of the first regression is presented below:



```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:        US_Box_Office_2   R-squared:                       0.694
Model:                            OLS   Adj. R-squared:                  0.592
Method:                 Least Squares   F-statistic:                     6.811
Date:                Tue, 19 Mar 2019   Prob (F-statistic):            0.00168
Time:                        15:59:04   Log-Likelihood:                -413.87
No. Observations:                  21   AIC:                             839.7
Df Residuals:                      15   BIC:                             846.0
Df Model:                           5                                         
Covariance Type:            nonrobust                                         
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept    -4.579e+08   4.43e+08     -1.033      0.318    -1.4e+09    4.87e+08
Budget           2.1715      0.595      3.651      0.002       0.904       3.439
IMDB_Ratings -3.814e+07   9.19e+07     -0.415      0.684   -2.34e+08    1.58e+08
Metascore     2.202e+06   6.82e+06      0.323      0.751   -1.23e+07    1.67e+07
RT            5.762e+08   7.31e+08      0.789      0.443   -9.81e+08    2.13e+09
Votes           77.5868    118.411      0.655      0.522    -174.800     329.973
==============================================================================
Omnibus:                        1.505   Durbin-Watson:                   1.548
Prob(Omnibus):                  0.471   Jarque-Bera (JB):                1.037
Skew:                           0.533   Prob(JB):                        0.595
Kurtosis:                       2.782   Cond. No.                     6.38e+09
==============================================================================
```

The budget was the only significant variable for predicting new movies' box office. Some of the ratings had positive coefficients while others had negative coefficients indicating potential multicollinearity. However, having budget as a significant variable is reasonable because usually movies with larger budgets include more popular actors and expensive special effects, therefore people tend to like and rate such movies more. Based on this specification, we received a prediction for the new Avengers to have a US Box Office equal to $383,518,800. This forecast appears to underestimate the US Box Office revenues given recent big-budget Marvel films and the US Box Office revenues they made. For example, Black Panther with a $200 million budget garnered $700 million in US Box Office revenues, while Avengers: Infinity War with a $300 million budget garnered $680 million in US Box Office revenues. In an effort to increase the model prediction accuracy, more variables were included in our next regression run.

We believed that the success of the next Marvel movie release might by dependent on the ratings of prior Marvel movie releases. For example, if people rated the previous movie highly, there might be greater likelihood of moviegoers going to the cinemas to see the continuation of Marvel movies. Thus, we included lagged variables, and the new regression output is provided below:



```
                            OLS Regression Results
==============================================================================
Dep. Variable:        US_Box_Office_2   R-squared:                       0.894
Model:                            OLS   Adj. R-squared:                  0.735
Method:                 Least Squares   F-statistic:                     5.629
Date:                Tue, 19 Mar 2019   Prob (F-statistic):             0.0102
Time:                        15:59:57   Log-Likelihood:                -402.74
No. Observations:                  21   AIC:                             831.5
Df Residuals:                       8   BIC:                             845.1
Df Model:                          12
Covariance Type:            nonrobust
============================================================================================
                              coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------
Intercept                6.099e+08    5.5e+08      1.110      0.299   -6.57e+08    1.88e+09
Budget                      1.3607      0.661      2.059      0.073      -0.163       2.885
IMDB_Ratings            -2.839e+08   1.07e+08     -2.656      0.029    -5.3e+08   -3.74e+07
Metascore               -1.052e+06   7.16e+06     -0.147      0.887   -1.76e+07    1.55e+07
RT                         1.49e+09   9.41e+08      1.584      0.152   -6.79e+08    3.66e+09
Votes                     483.9952    185.229      2.613      0.031      56.856     911.135
lag_IMDB                -5.733e+07   6.04e+07     -0.949      0.370   -1.97e+08     8.2e+07
lag_Metascore            2.344e+07   1.24e+07      1.894      0.095    -5.1e+06     5.2e+07
lag_Votes                -110.1746    255.246     -0.432      0.677    -698.774     478.425
lag_RT                  -9.205e+08   8.93e+08     -1.031      0.333   -2.98e+09    1.14e+09
lag_Budget                 -2.2155      1.240     -1.787      0.112      -5.074       0.643
lag_US_Box_Office_2        -0.1904      0.520     -0.366      0.724      -1.389       1.008
lag_Worldwide_Box_Office    0.3414      0.239      1.430      0.191      -0.209       0.892
==============================================================================
Omnibus:                        0.763   Durbin-Watson:                   2.325
Prob(Omnibus):                  0.683   Jarque-Bera (JB):                0.204
Skew:                          -0.238   Prob(JB):                        0.903
Kurtosis:                       3.076   Cond. No.                     6.39e+10
==============================================================================
```

According to this model, the new forecast for the 2019 movie is $663,084,300. We believe that new Avengers would be as successful as Black Panther (2018) and Avengers: Infinity Wars (2018). However, we acknowledge the fact our forecast is subject to limitations because it relies solely on financial information and ratings. There are many possible unaccounted factors which are hard to measure (consumer opinions, quality of the plot, and so on) and which might be very significant in predicting the Avengers' US Box Office.

## VI. Conclusion

The entire project was conducted through 5 distinct steps that include searching for and importing data sources, understanding the underlying HTML structure of a website, scraping the data output, running analysis on the output, and predicting metrics for Avengers: Endgame. Using the *BeautifulSoup* package from Python to extract and parse the data from three main websites, namely, IMDB, Rotten Tomatoes, and Metacritic, we were able to scrape the necessary information for the project. After scraping the relevant fields which include the movie title, year-released, IMDB ratings, votes, and gross from the IMDB website, we get a list of useful relevant information needed for all the Marvel Studio films thus far. Similarly, by following the same steps to scrape Marvel Movie ratings from Rotten Tomatoes. Based on our web scraping and data cleaning, we visualize the aggregated data via matplotlib and seaborn from Python. According to our data exploration and data analysis, we find out that Black Panther and Iron Man are the highest rated Marvel movies, and Thor: The Dark World and The Incredible Hulk have the lowest ratings. After running several specifications of OLS multivariate regression and readjusting our models, our forecast US Box Office for Avengers: Endgame is $315,205,100. We conclude that Avengers: Endgame would be a quite the hit like other notable marvel movies. Additionally, running a second modified run on the same model with additional assumptions yielded at higher $663,084,300 prediction for the US Box Office.

As the entire process goes through the complete data science process, from searching for and acquiring the necessary target data to analyzing and presenting the findings. This project provides an excellent opportunity to take advantage of web scraping techniques and methodologies that reduce the time required to extract data from various potential online sources. Now we are able to apply our web-scraping knowledge and skills to other data driven projects and tasks and present the findings and takeaways in a professional and logical way.

## VII. References

Kulkarni A., Shivananda A. (2019) Extracting the Data. In: Natural Language Processing Recipes. Apress, Berkeley, CA

## VIII. Addendum

Slight differences in charts between presentation and report

- As Captain Marvel is currently in theaters, there is a variance between some of the values with regards to the Captain Marvel movie from when we presented and this report. While the overall effect is negligible, there will be some charts that will be affected visually however the end result is a m ore accurate prediction of the US Box office of Avengers: Endgame.