Homework 2 Report

Semi-Structured Data: Top rated Movies from 1990-2020

1

Marriah Lewis

08.30.2021

IST 652 Summer 2021

[1]How Georgia Became a Filmmaking Hub. (2016–2021). [Image]. http://therhinelawfirm.com/georgia-became-filmmaking-hub/

INTRODUCTION

The purpose of this assignment is to create a program that can process semi-structured data. After cleaning and preparing the data, the following questions below will be answered.

Questions

1. Based on ratings, can we predict if the metascore will be good or bad?
2. Does the year a movie is released dictate if the movie will be on the top?
3. What is the IMDb distribution when variables such as metascores and rating are incorporated?

Packages used:

```
from bs4 import BeautifulSoup

import pandas as pd

import requests

import matplotlib.pyplot as plt
```

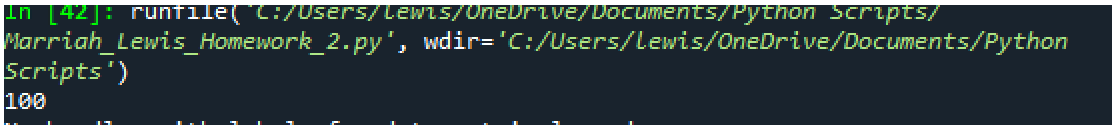
DATA

```
#Importing the URL from IMDb

#top 1000, United States-English in ascending order from IMDb (1990-2020)

url='https://www.imdb.com/search/title/?release_date=1990-01-01,2020-12-31&groups=top
_1000&countries=us&languages=en&my_ratings=exclude&count=100'

response= requests.get(url)

#print(response.text[:500])

soup=BeautifulSoup(response.text, 'html.parser')

#print(type(soup))
```

movie_containers= soup.find_all('div', class_='lister-item mode-advanced') #Select all the 100 movie containers from a single page

# print(type(movie_containers))

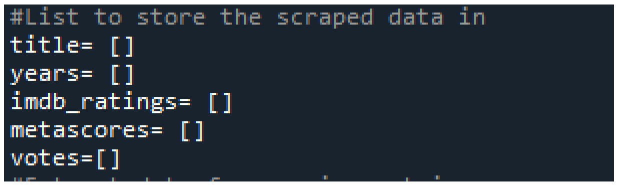
# print(len(movie_containers))



```
In [42]: runfile('C:/Users/lewis/OneDrive/Documents/Python Scripts/
Marriah_Lewis_Homework_2.py', wdir='C:/Users/lewis/OneDrive/Documents/Python
Scripts')
100
```

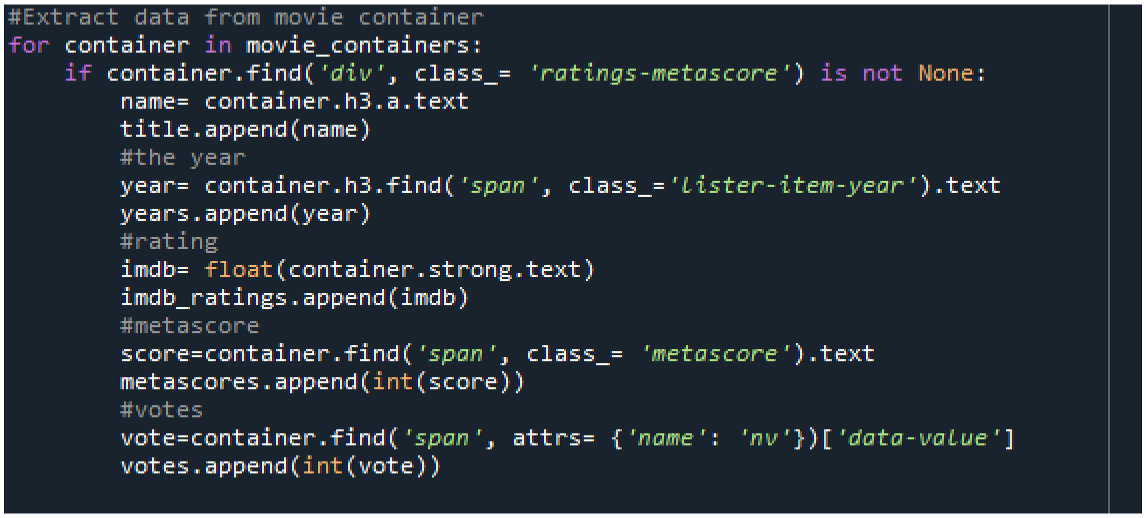
Cleaning/Preparation

First, a list was created to store each scraped data.



```
#List to store the scraped data in
title= []
years= []
imdb_ratings= []
metascores= []
votes=[]
```

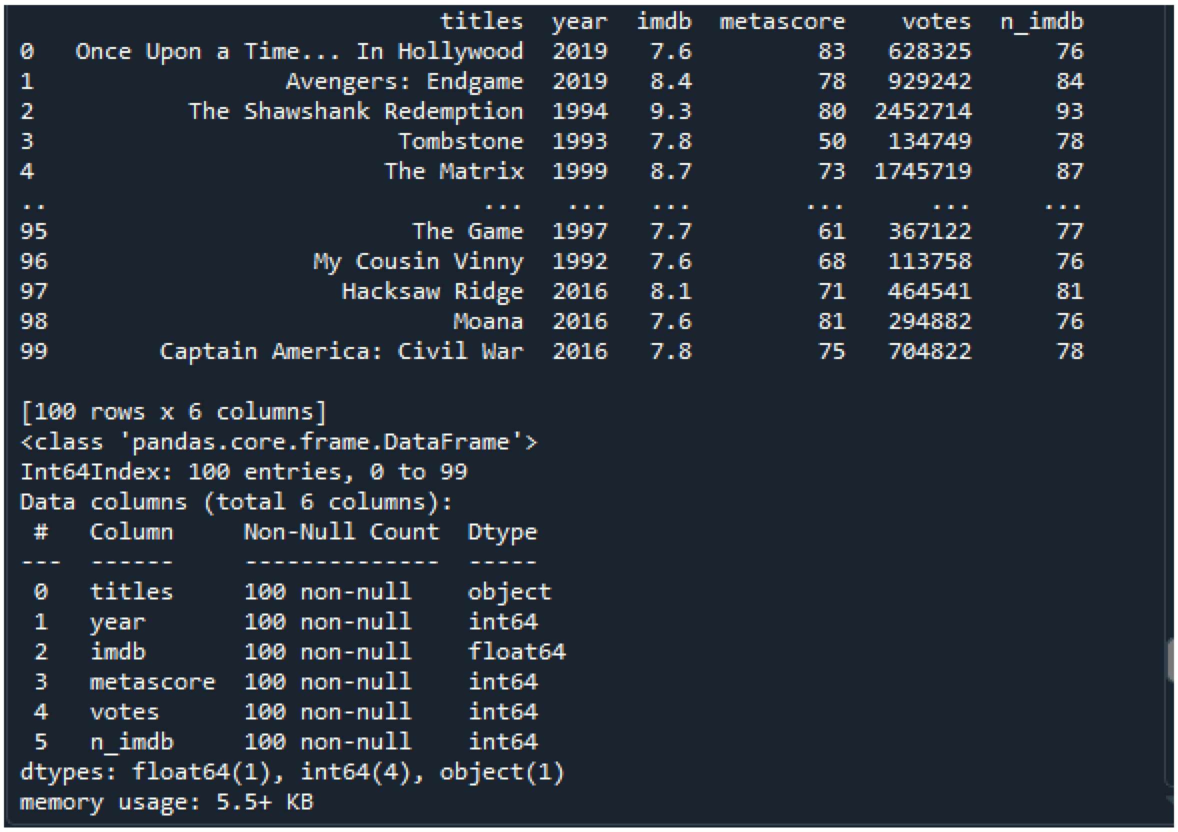
After creating the lists, the extraction of the data from the movie containers was next. To make sure I documented the classes correctly an inspection of the url page was reviewed to confirm the following:



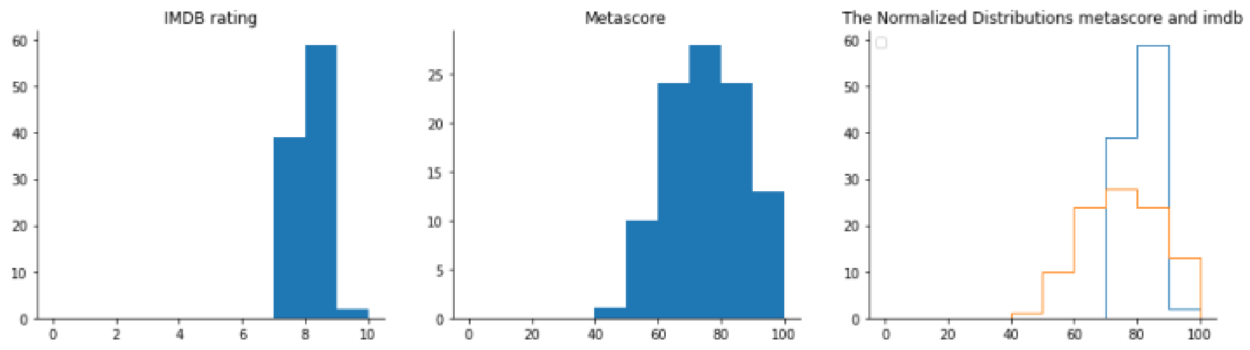
```
#Extract data from movie container
for container in movie_containers:
    if container.find('div', class_= 'ratings-metascore') is not None:
        name= container.h3.a.text
        title.append(name)
        #the year
        year= container.h3.find('span', class_='lister-item-year').text
        years.append(year)
        #rating
        imdb= float(container.strong.text)
        imdb_ratings.append(imdb)
        #metascore
        score=container.find('span', class_= 'metascore').text
        metascores.append(int(score))
        #votes
        vote=container.find('span', attrs= {'name': 'nv'})['data-value']
        votes.append(int(vote))
```

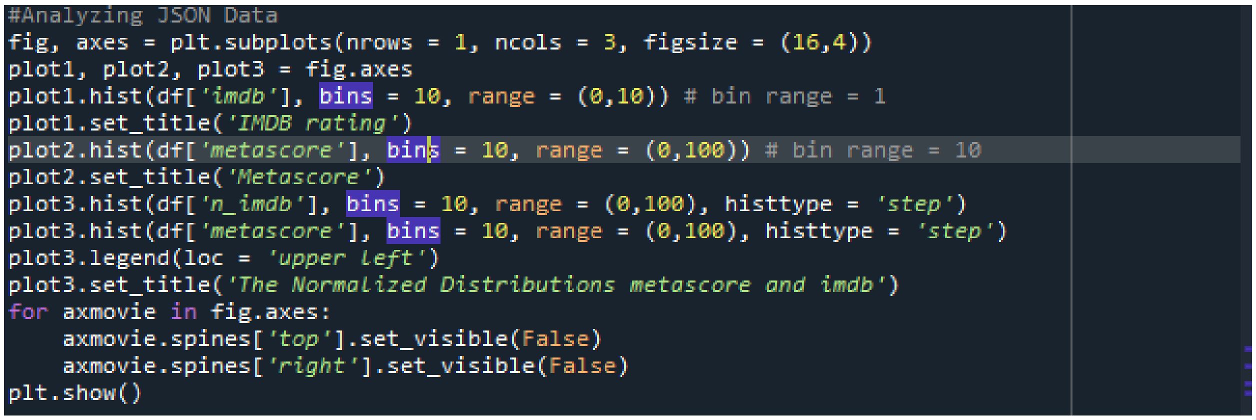
After extraction the data collected was moved to a dataframe and the columns were renamed for a cleaner and organized display. There was a ValueError that was populating when trying to process the year the movie was released so in order to avoid that error. The year values were converted to integers. Lastly, metascore and imdb rating were normalized in order to plot the distribution between imdb and metascore shown in the histogram below. Cleaning was complete and the dataframe was saved as a json file for further analysis. The Json file format was a little off so I used JsonFormatter.

RESULTS/VISUALIZATIONS



```
                            titles  year  imdb  metascore    votes  n_imdb
0   Once Upon a Time... In Hollywood  2019   7.6         83   628325      76
1                Avengers: Endgame  2019   8.4         78   929242      84
2         The Shawshank Redemption  1994   9.3         80  2452714      93
3                        Tombstone  1993   7.8         50   134749      78
4                       The Matrix  1999   8.7         73  1745719      87
..                             ...   ...   ...        ...      ...     ...
95                        The Game  1997   7.7         61   367122      77
96                 My Cousin Vinny  1992   7.6         68   113758      76
97                   Hacksaw Ridge  2016   8.1         71   464541      81
98                           Moana  2016   7.6         81   294882      76
99      Captain America: Civil War  2016   7.8         75   704822      78

[100 rows x 6 columns]
<class 'pandas.core.frame.DataFrame'>
Int64Index: 100 entries, 0 to 99
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   titles     100 non-null    object
 1   year       100 non-null    int64
 2   imdb       100 non-null    float64
 3   metascore  100 non-null    int64
 4   votes      100 non-null    int64
 5   n_imdb     100 non-null    int64
dtypes: float64(1), int64(4), object(1)
memory usage: 5.5+ KB
```





```
#Analyzing JSON Data
fig, axes = plt.subplots(nrows = 1, ncols = 3, figsize = (16,4))
plot1, plot2, plot3 = fig.axes
plot1.hist(df['imdb'], bins = 10, range = (0,10)) # bin range = 1
plot1.set_title('IMDB rating')
plot2.hist(df['metascore'], bins = 10, range = (0,100)) # bin range = 10
plot2.set_title('Metascore')
plot3.hist(df['n_imdb'], bins = 10, range = (0,100), histtype = 'step')
plot3.hist(df['metascore'], bins = 10, range = (0,100), histtype = 'step')
plot3.legend(loc = 'upper left')
plot3.set_title('The Normalized Distributions metascore and imdb')
for axmovie in fig.axes:
    axmovie.spines['top'].set_visible(False)
    axmovie.spines['right'].set_visible(False)
plt.show()
```

CONCLUSION

When observing the IMDb histogram the majority of the ratings are between 6 and 8. Few movies have a rating of more than 8, and even fewer have a rating of less than 4. The metascore rating distribution reflects that of a normal distribution, with the majority of ratings being average and peaking at around 50. However, on the last graph that compares the imdb ratings with the metascore the distribution is highly skewed towards the upper region of the average ratings. The metascore has much more equal distribution. Could it be because viewers tend to have a more dichotomous way of viewing movies? If a movie is really good (high scores between 8 and 10) versus a bad movie that is scored lower in the 3 and 4s. After further review of the year, the year does not show any importances therefore the graph was discarded. More information is needed to analyze further what variables can predict whether a movie does well or not. The next step would be gathering more data about those movies from different websites.