student

January 31, 2022

0.1 Final Project Submission

Please fill out: * Student name: Maliha Momtaj * Student pace: self paced / part time / full time: Part time * Scheduled project review date/time: Feb 1, 2022 * Instructor name: Claude Fried * Blog post URL: https://dev.to/maliha1009/data-manipulation-in-python-4ag8

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

[2]:
```
import matplotlib.pyplot as plt
```

[3]:
```
import pandas as pd
import numpy as np
```

[4]:
```
#Dataframe tmdb to be exported and name as csv1.
#This dataframe has movie ratings, release date which will be used to find the␣
 ↪correlation.

#export tmdb.movies.csv.gz as csv 1 into dataframe
csv1 = pd.read_csv("zippeddata/tmdb.movies.csv.gz")
csv1.head()
```

[4]:

	Unnamed: 0	genre_ids	id	original_language	original_title	popularity	release_date
0	0	[12, 14, 10751]	12444	en	Harry Potter and the Deathly Hallows: Part 1	33.533	2010-11-19
1	1	[14, 12, 16, 10751]	10191	en	How to Train Your Dragon	28.734	2010-03-26
2	2	[12, 28, 878]	10138	en	Iron Man 2	28.515	2010-05-07
3	3	[16, 35, 10751]	862	en	Toy Story	28.005	1995-11-22
4	4	[28, 878, 12]	27205	en	Inception	27.920	2010-07-16

```
                                          title  vote_average  vote_count
0  Harry Potter and the Deathly Hallows: Part 1           7.7       10788
1                   How to Train Your Dragon           7.7        7610
2                                  Iron Man 2           6.8       12368
3                                   Toy Story           7.9       10174
4                                   Inception           8.3       22186
```

[5]:
```
#Dataframe tn.movie to be exported and name as csv2.
#This dataframe has production budget which will be used to find the␣
 ↪correlation.

#export tn.movie_budgets.csv.gz as csv 2 into dataframe
csv2 = pd.read_csv("zippeddata/tn.movie_budgets.csv.gz")
csv2.head()
```

[5]:
```
   id  release_date                                       movie  \
0   1  Dec 18, 2009                                      Avatar
1   2  May 20, 2011  Pirates of the Caribbean: On Stranger Tides
2   3   Jun 7, 2019                                Dark Phoenix
3   4   May 1, 2015                  Avengers: Age of Ultron
4   5  Dec 15, 2017           Star Wars Ep. VIII: The Last Jedi

  production_budget domestic_gross worldwide_gross
0      $425,000,000   $760,507,625  $2,776,345,279
1      $410,600,000   $241,063,875  $1,045,663,875
2      $350,000,000    $42,762,350    $149,762,350
3      $330,600,000   $459,005,868  $1,403,013,963
4      $317,000,000   $620,181,382  $1,316,721,747
```

[6]:
```
#Dataframe csv1 & csv2 will be merged or joined to get a complete file that␣
 ↪list movie ratings, release date, budget.
#These data will be used to conduct the study.
#Merging csv2 & csv1 for the columns; prefix with popularity to make it distinct

df = csv2.set_index('movie').join(csv1.set_index('title'),lsuffix='popularity')
df
```

[6]:
```
                              idpopularity release_datepopularity  \
#Horror                                 16             Nov 20, 2015
(500) Days of Summer                    55             Jul 17, 2009
10 Cloverfield Lane                     54             Mar 11, 2016
10 Days in a Madhouse                   48             Nov 11, 2015
10 Things I Hate About You              63             Mar 31, 1999
…                                      …                        …
mother!                                 59             Sep 15, 2017
xXx                                     98              Aug 9, 2002
```

```
xXx: Return of Xander Cage            15           Jan 20, 2017
Ã l\'intÃ©rieur                       57           Apr 15, 2008
é ·æ± ä¸ è  (CJ7)                     2           Mar 7, 2008

                           production_budget domestic_gross worldwide_gross  \
#Horror                            $1,500,000            $0             $0
(500) Days of Summer               $7,500,000    $32,425,665     $34,439,060
10 Cloverfield Lane                $5,000,000    $72,082,999    $108,286,422
10 Days in a Madhouse             $12,000,000        $14,616         $14,616
10 Things I Hate About You        $13,000,000    $38,177,966     $60,413,950
…                                       …             …              …
mother!                           $30,000,000    $17,800,004     $42,531,076
xXx                               $70,000,000   $141,930,000    $267,200,000
xXx: Return of Xander Cage        $85,000,000    $44,898,413    $345,033,359
Ã l\'intÃ©rieur                    $3,000,000            $0        $895,932
é ·æ± ä¸ è  (CJ7)                 $20,000,000       $206,678     $47,300,771

                            Unnamed: 0           genre_ids        id  \
#Horror                        14656.0  [18, 9648, 27, 53]  301325.0
(500) Days of Summer               NaN                 NaN       NaN
10 Cloverfield Lane            17422.0       [53, 878, 18]  333371.0
10 Days in a Madhouse          15907.0                [18]  345003.0
10 Things I Hate About You         NaN                 NaN       NaN
…                                    …                 …         …
mother!                        20707.0      [18, 27, 9648]  381283.0
xXx                                NaN                 NaN       NaN
xXx: Return of Xander Cage     20651.0        [28, 12, 80]   47971.0
Ã l\'intÃ©rieur                    NaN                 NaN       NaN
é ·æ± ä¸ è  (CJ7)                  NaN                 NaN       NaN

                           original_language              original_title  \
#Horror                                   de                     #Horror
(500) Days of Summer                     NaN                         NaN
10 Cloverfield Lane                       en          10 Cloverfield Lane
10 Days in a Madhouse                     en       10 Days in a Madhouse
10 Things I Hate About You               NaN                         NaN
…                                          …                           …
mother!                                   en                     mother!
xXx                                      NaN                         NaN
xXx: Return of Xander Cage                en  xXx: Return of Xander Cage
Ã l\'intÃ©rieur                           NaN                         NaN
é ·æ± ä¸ è  (CJ7)                        NaN                        NaN

                            popularity release_date  vote_average  vote_count
#Horror                          6.099   2015-11-20           3.3       102.0
(500) Days of Summer               NaN          NaN           NaN         NaN
10 Cloverfield Lane             17.892   2016-03-11           6.9      4629.0
```

```
10 Days in a Madhouse            0.955   2015-11-20          5.4         7.0
10 Things I Hate About You         NaN          NaN          NaN         NaN
…                                  …            …            …           …
mother!                         15.227   2017-09-15          7.0      3458.0
xXx                                NaN          NaN          NaN         NaN
xXx: Return of Xander Cage      21.749   2017-01-20          5.6      2452.0
Ã l\'intÃ©rieur                   NaN          NaN          NaN         NaN
é ·æ± ä¸ è  (CJ7)                NaN          NaN          NaN         NaN

[6190 rows x 14 columns]
```

[7]:
```
#Since movie ratings or popularity will be used for this analysis, any value␣
 ↪"NaN" should be excluded from the popularity row.

#removing NaN, every row should have value

is_NaN = df[df['popularity']==np.NaN]
print(is_NaN.head())
```

```
Empty DataFrame
Columns: [idpopularity, release_datepopularity, production_budget,
domestic_gross, worldwide_gross, Unnamed: 0, genre_ids, id, original_language,
original_title, popularity, release_date, vote_average, vote_count]
Index: []
```

[8]:
```
#For the analysis, we would try to see if there is a specific genre get higher␣
 ↪or lower rating.
#Therefore, we would create a collective column that will flag which genre the␣
 ↪movie is.

# collective columns includes a specific genre
def is_genre(x, genre):
    try:
        if genre in x:
            return True
        else:
            return False
    except TypeError as te:
        return False
```

[9]:
```
# to flag specific genre
def is_comedy(x):
    return is_genre(x, 'Comedy')
```

[10]:
```
# to flag specific genre
def is_action(x):
    return is_genre(x, 'Action')
```

[11]:
```
# to flag specific genre
def is_romance(x):
    return is_genre(x, 'Romance')
```

[12]:
```
# to flag specific genre
def is_horror(x):
    return is_genre(x, 'Horror')
```

[13]:
```
#For this study, more dataframe to be loaded and named as name_basics to read␣
 ↪in pandas.

#loading more data from imdb.name.basics.csv.gz
name_basics = pd.read_csv('zippedData/imdb.name.basics.csv.gz')
```

[14]:
```
# Following 2 fields were not used for this analysis.
#str that contain multiple values, value was not used

name_basics['known_for_titles']=name_basics.known_for_titles.str.split('.')
name_basics['primary_profession']=name_basics.primary_profession.str.split('.')
```

[15]:
```
#For this study, more dataframe to be loaded and read and named as title_basics.

#loading more values imdb.title.basics.csv
title_basics = pd.read_csv('zippedData/imdb.title.basics.csv.gz')
```

[16]:
```
title_basics['genres'] = title_basics.genres.str.split(',')
```

[17]:
```
#loading more values into dataframe
title_ratings = pd.read_csv('zippedData/imdb.title.ratings.csv.gz')
```

[18]:
```
#loading more values into dataframe
title_principals = pd.read_csv('zippedData/imdb.title.principals.csv.gz')
```

[19]:
```
#loading more values into dataframe
title_crew = pd.read_csv('zippedData/imdb.title.crew.csv.gz')
```

[20]:
```
# to join files with ratings & basics
imdb_df = title_ratings.set_index('tconst').join(title_basics.
 ↪set_index('tconst'))
```

[21]:
```
# to merge the csv files
imdb_df = imdb_df.join(title_principals.set_index('tconst'))
```

[22]:
```
# to merge the csv files
imdb_df = imdb_df.join(title_crew.set_index('tconst'), rsuffix = 'crew')
```

[23]:
```
#to flag specific genre
imdb_df['is_comedy']=imdb_df.genres.apply(is_comedy)
```

[24]:
```
#to flag specific genre
imdb_df['is_action']=imdb_df.genres.apply(is_action)
```

[25]:
```
#to flag specific genre
imdb_df['is_horror']=imdb_df.genres.apply(is_horror)
```

[26]:
```
#to flag specific genre
imdb_df['is_romance']=imdb_df.genres.apply(is_romance)
```

[27]:
```
#The new combined file is as follows. This dataframe will be used to analyze␣
 ↪and find correlation between movie ratings, run time, release year.
imdb_df
```

[27]:

tconst	averagerating	numvotes	primary_title	original_title	start_year	\
tt0063540	7.0	77	Sunghursh	Sunghursh	2013	
tt0063540	7.0	77	Sunghursh	Sunghursh	2013	
tt0063540	7.0	77	Sunghursh	Sunghursh	2013	
tt0063540	7.0	77	Sunghursh	Sunghursh	2013	
tt0063540	7.0	77	Sunghursh	Sunghursh	2013	
…	…	…	…	…	…	
tt9916160	6.5	11	Drømmeland	Drømmeland	2019	
tt9916160	6.5	11	Drømmeland	Drømmeland	2019	
tt9916160	6.5	11	Drømmeland	Drømmeland	2019	
tt9916160	6.5	11	Drømmeland	Drømmeland	2019	
tt9916160	6.5	11	Drømmeland	Drømmeland	2019	

tconst	runtime_minutes	genres	ordering	nconst	\
tt0063540	175.0	[Action, Crime, Drama]	10.0	nm0006210	
tt0063540	175.0	[Action, Crime, Drama]	1.0	nm0474801	
tt0063540	175.0	[Action, Crime, Drama]	2.0	nm0904537	
tt0063540	175.0	[Action, Crime, Drama]	3.0	nm0756379	
tt0063540	175.0	[Action, Crime, Drama]	4.0	nm0474876	
…	…	…	…	…	
tt9916160	72.0	[Documentary]	3.0	nm2768724	
tt9916160	72.0	[Documentary]	4.0	nm4241788	
tt9916160	72.0	[Documentary]	5.0	nm6969694	
tt9916160	72.0	[Documentary]	6.0	nm3256778	
tt9916160	72.0	[Documentary]	7.0	nm0462955	

tconst	category	job	characters	\
tt0063540	composer	NaN	NaN	

```
tt0063540            actor       NaN  ["Kundan S. Prasad","Bajrangi"]
tt0063540          actress       NaN      ["Munni","Laila-E-Aasmaan"]
tt0063540            actor       NaN            ["Ganeshi N. Prasad"]
tt0063540            actor       NaN             ["Dwarka N. Prasad"]
…                      …       …                                  …
tt9916160         producer  producer                              NaN
tt9916160         composer       NaN                              NaN
tt9916160  cinematographer       NaN                              NaN
tt9916160           editor       NaN                              NaN
tt9916160           editor       NaN                              NaN

           directors                                    writers  is_comedy  \
tconst
tt0063540  nm0712540  nm0023551,nm1194313,nm0347899,nm1391276      False
tt0063540  nm0712540  nm0023551,nm1194313,nm0347899,nm1391276      False
tt0063540  nm0712540  nm0023551,nm1194313,nm0347899,nm1391276      False
tt0063540  nm0712540  nm0023551,nm1194313,nm0347899,nm1391276      False
tt0063540  nm0712540  nm0023551,nm1194313,nm0347899,nm1391276      False
…              …                                          …        …
tt9916160  nm5684093                                        NaN      False
tt9916160  nm5684093                                        NaN      False
tt9916160  nm5684093                                        NaN      False
tt9916160  nm5684093                                        NaN      False
tt9916160  nm5684093                                        NaN      False

           is_action  is_horror  is_romance
tconst
tt0063540       True      False       False
tt0063540       True      False       False
tt0063540       True      False       False
tt0063540       True      False       False
tt0063540       True      False       False
…               …         …           …
tt9916160      False      False       False
tt9916160      False      False       False
tt9916160      False      False       False
tt9916160      False      False       False
tt9916160      False      False       False

[629926 rows x 18 columns]
```

[28]:
```
# to see the columns
imdb_df.columns
```

[28]:
```
Index(['averagerating', 'numvotes', 'primary_title', 'original_title',
       'start_year', 'runtime_minutes', 'genres', 'ordering', 'nconst',
       'category', 'job', 'characters', 'directors', 'writers', 'is_comedy',
```

```
       'is_action', 'is_horror', 'is_romance'],
      dtype='object')
```

[29]:
```
#loading more values into dataframe
bom_df = pd.read_csv('zippedData/bom.movie_gross.csv.gz')
```

[30]:
```
#loading more values into dataframe
tn_budgets_df = pd.read_csv('zippedData/tn.movie_budgets.csv.gz')
```

[31]:
```
#merging 2 files
ib_df = imdb_df.set_index('primary_title').join(bom_df.set_index('title'))
```

[32]:
```
filtered_ib_df = ib_df[~np.isnan(ib_df['domestic_gross'])]
```

[33]:
```
#to merge
all_df = filtered_ib_df.join(tn_budgets_df.set_index('movie'),rsuffix='_tn')
```

[34]:
```
filtered_all_df = all_df[-np.isnan(all_df['id'])]
```

[35]:
```
filtered_all_df.to_csv('output.csv')
```

[36]:
```
#combined all csv files containing data from imdb, rt, tndb and renamed it as:

filtered_all_df
```

[36]:

	averagerating	numvotes	original_title	start_year	\
10 Cloverfield Lane	7.2	260383	10 Cloverfield Lane	2016	
10 Cloverfield Lane	7.2	260383	10 Cloverfield Lane	2016	
10 Cloverfield Lane	7.2	260383	10 Cloverfield Lane	2016	
10 Cloverfield Lane	7.2	260383	10 Cloverfield Lane	2016	
10 Cloverfield Lane	7.2	260383	10 Cloverfield Lane	2016	
…	…	…	…	…	
Zootopia	8.0	383446	Zootopia	2016	
Zootopia	8.0	383446	Zootopia	2016	
Zootopia	8.0	383446	Zootopia	2016	
Zootopia	8.0	383446	Zootopia	2016	
Zootopia	8.0	383446	Zootopia	2016	

	runtime_minutes	genres	\
10 Cloverfield Lane	103.0	[Drama, Horror, Mystery]	
10 Cloverfield Lane	103.0	[Drama, Horror, Mystery]	
10 Cloverfield Lane	103.0	[Drama, Horror, Mystery]	
10 Cloverfield Lane	103.0	[Drama, Horror, Mystery]	
10 Cloverfield Lane	103.0	[Drama, Horror, Mystery]	
…	…	…	
Zootopia	108.0	[Adventure, Animation, Comedy]	
Zootopia	108.0	[Adventure, Animation, Comedy]	

```
Zootopia                        108.0  [Adventure, Animation, Comedy]
Zootopia                        108.0  [Adventure, Animation, Comedy]
Zootopia                        108.0  [Adventure, Animation, Comedy]

                     ordering     nconst  category          job  …  \
10 Cloverfield Lane      10.0  nm6618222  producer     producer  …
10 Cloverfield Lane       1.0  nm0000422     actor          NaN  …
10 Cloverfield Lane       2.0  nm0935541   actress          NaN  …
10 Cloverfield Lane       3.0  nm0302330     actor          NaN  …
10 Cloverfield Lane       4.0  nm0341174     actor          NaN  …
…                         …          …         …            …  …
Zootopia                  5.0  nm0397174  director          NaN  …
Zootopia                  6.0  nm0601781  director          NaN  …
Zootopia                  7.0  nm1158544  director  co-director  …
Zootopia                  8.0  nm0714114    writer     story by  …
Zootopia                  9.0  nm2888684    writer     story by  …

                    is_romance studio domestic_gross  foreign_gross    year  \
10 Cloverfield Lane      False   Par.     72100000.0       38100000  2016.0
10 Cloverfield Lane      False   Par.     72100000.0       38100000  2016.0
10 Cloverfield Lane      False   Par.     72100000.0       38100000  2016.0
10 Cloverfield Lane      False   Par.     72100000.0       38100000  2016.0
10 Cloverfield Lane      False   Par.     72100000.0       38100000  2016.0
…                        …     …            …              …     …
Zootopia                 False     BV    341300000.0      682500000  2016.0
Zootopia                 False     BV    341300000.0      682500000  2016.0
Zootopia                 False     BV    341300000.0      682500000  2016.0
Zootopia                 False     BV    341300000.0      682500000  2016.0
Zootopia                 False     BV    341300000.0      682500000  2016.0

                       id  release_date production_budget  domestic_gross_tn  \
10 Cloverfield Lane  54.0  Mar 11, 2016        $5,000,000        $72,082,999
10 Cloverfield Lane  54.0  Mar 11, 2016        $5,000,000        $72,082,999
10 Cloverfield Lane  54.0  Mar 11, 2016        $5,000,000        $72,082,999
10 Cloverfield Lane  54.0  Mar 11, 2016        $5,000,000        $72,082,999
10 Cloverfield Lane  54.0  Mar 11, 2016        $5,000,000        $72,082,999
…                    …          …                 …                 …
Zootopia             57.0   Mar 4, 2016      $150,000,000       $341,268,248
Zootopia             57.0   Mar 4, 2016      $150,000,000       $341,268,248
Zootopia             57.0   Mar 4, 2016      $150,000,000       $341,268,248
Zootopia             57.0   Mar 4, 2016      $150,000,000       $341,268,248
Zootopia             57.0   Mar 4, 2016      $150,000,000       $341,268,248

                    worldwide_gross
10 Cloverfield Lane    $108,286,422
10 Cloverfield Lane    $108,286,422
10 Cloverfield Lane    $108,286,422
```

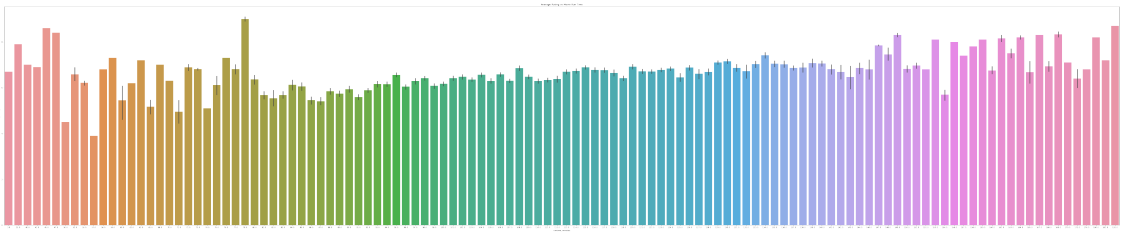
```
10 Cloverfield Lane    $108,286,422
10 Cloverfield Lane    $108,286,422
…                             …
Zootopia             $1,019,429,616
Zootopia             $1,019,429,616
Zootopia             $1,019,429,616
Zootopia             $1,019,429,616
Zootopia             $1,019,429,616

[13681 rows x 26 columns]
```

[37]:
```
#Average rating vs movie run time:
#The following graph shows the correlation between movie run time and movie␣
 ↪ratings.
#Based on the graph, movies run between 60 minutes to 130 minutes have higher␣
 ↪ratings.
#Any movie wuth higher than 60 mins have =<5 ratings.


#To draw bar graph:
plt.figure(figsize=(100,20))
plt.title("Average Rating vs Movie Run Time")
sns.barplot(x = 'runtime_minutes', y = 'averagerating', data = filtered_all_df)
```

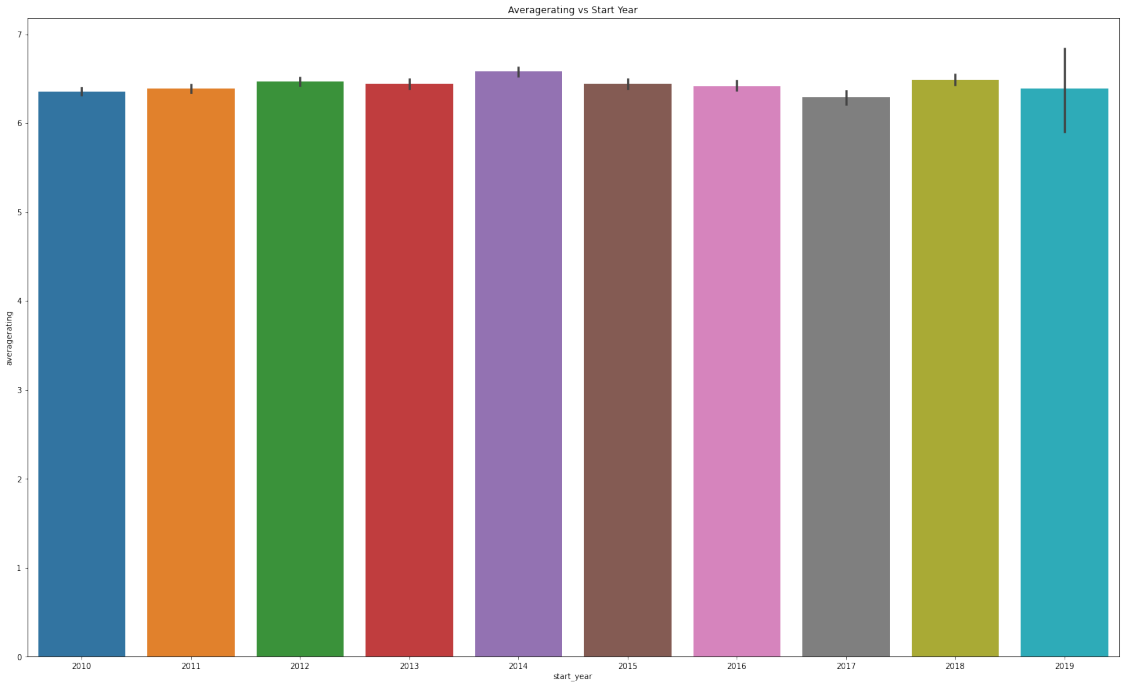
[37]:
```
<AxesSubplot:title={'center':'Average Rating vs Movie Run Time'},
xlabel='runtime_minutes', ylabel='averagerating'>
```



[38]:
```
#Average rating vs movie start year:
#The following graph shows the correlation between movie release year and movie␣
 ↪ratings.
#Based on the graph, the decrease in ratings for films in 2010 was presumably␣
 ↪caused by the recession of 2008 to 2010.
#But it also may be due to a reporting delay for low-budget films.
#Movies released in 2015 - 2019 seem to have rated higher than previous year.
#The lower ratings for the year of 2017 can be caused by advertising campaigns,␣
 ↪sequel fatigue or home streaming.
```

```
#To draw bar graph:
plt.figure(figsize=(25,15))
plt.title("Averagerating vs Start Year")
sns.barplot(x = 'start_year', y = 'averagerating', data = filtered_all_df)
```

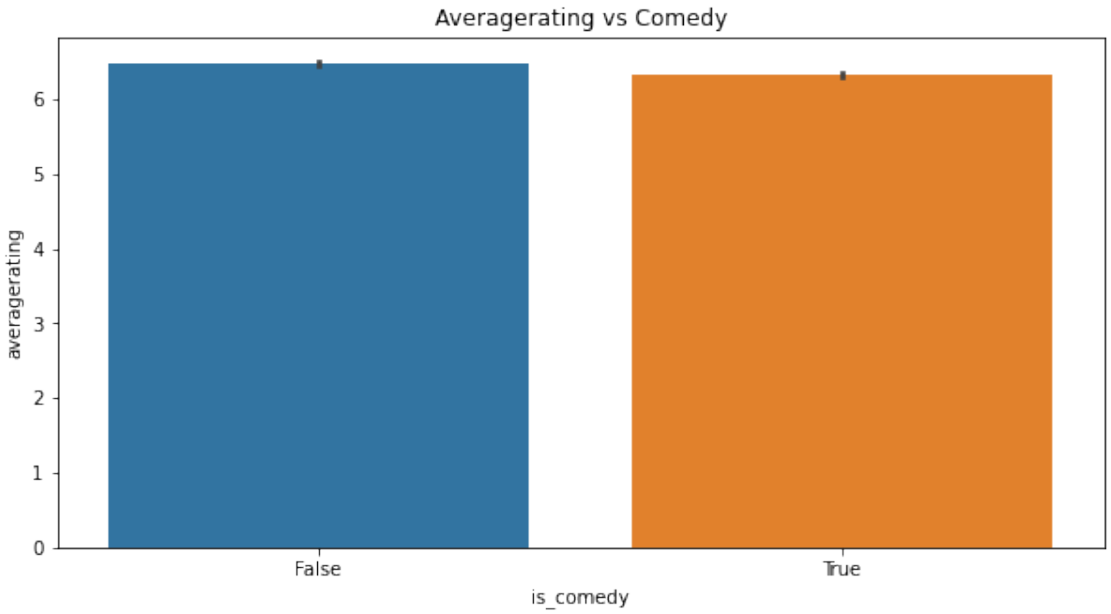
[38]:
```
<AxesSubplot:title={'center':'Averagerating vs Start Year'},
xlabel='start_year', ylabel='averagerating'>
```



[39]:
```
#Average rating vs Comedy:
#Genre "Comedy" receives overall higher ratings.

#To draw bar graph:
plt.figure(figsize=(10,5))
plt.title("Averagerating vs Comedy")
sns.barplot(x = 'is_comedy', y = 'averagerating', data = filtered_all_df)
```

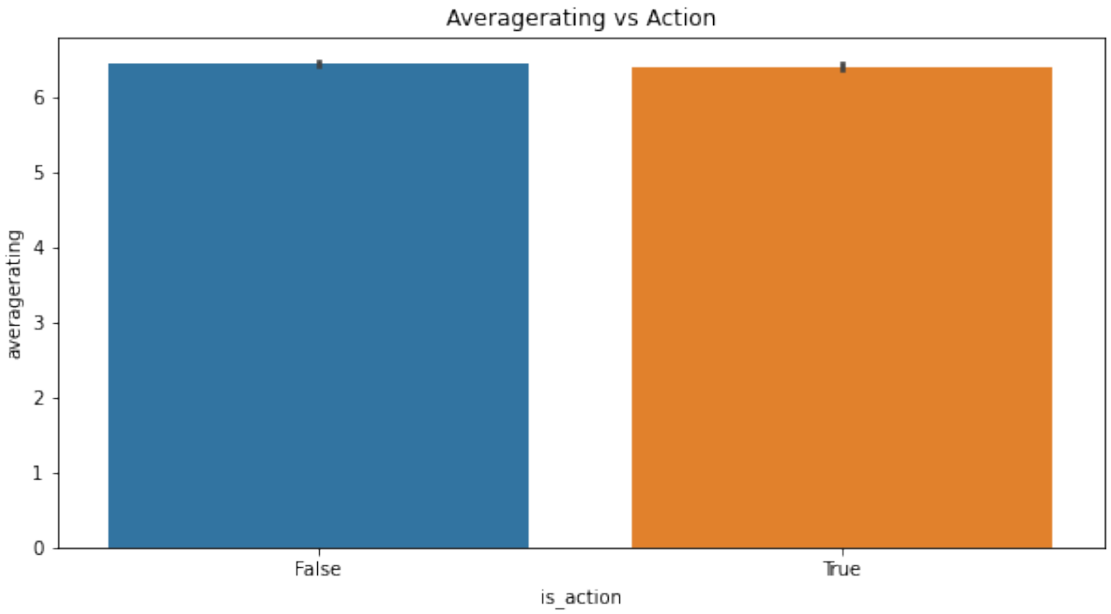
[39]:
```
<AxesSubplot:title={'center':'Averagerating vs Comedy'}, xlabel='is_comedy',
ylabel='averagerating'>
```



[40]:
```
#Average rating vs Action:
#Genre "Action" receives overall higher ratings.

#To draw bar graph:
plt.figure(figsize=(10,5))
plt.title("Averagerating vs Action")
sns.barplot(x = 'is_action', y = 'averagerating', data = filtered_all_df)
```

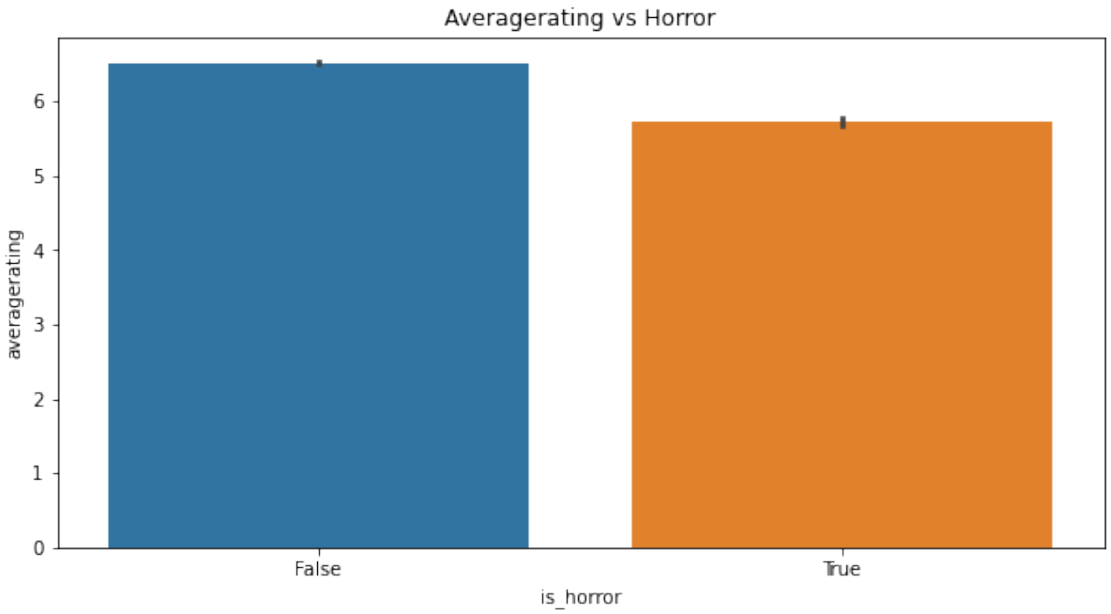
[40]:
```
<AxesSubplot:title={'center':'Averagerating vs Action'}, xlabel='is_action',
ylabel='averagerating'>
```



[41]:
```
#Average rating vs Horror:
#Genre "Horror" receives overall lower ratings than Comedy & Action.

#To draw bar graph:
plt.figure(figsize=(10,5))
plt.title("Averagerating vs Horror")
sns.barplot(x = 'is_horror', y = 'averagerating', data = filtered_all_df)
```

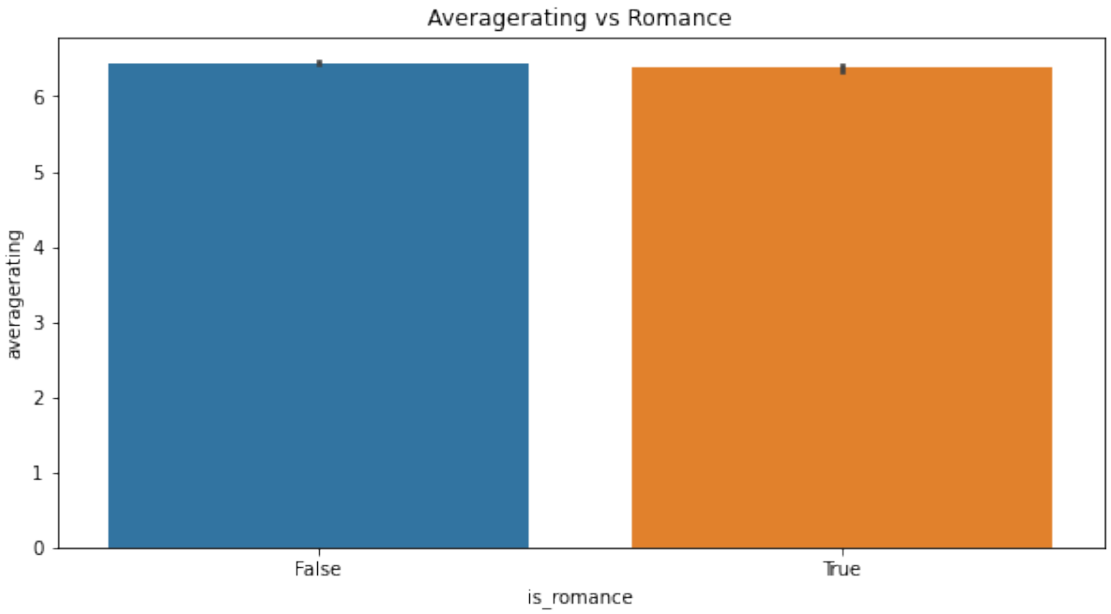
[41]:
```
<AxesSubplot:title={'center':'Averagerating vs Horror'}, xlabel='is_horror',
ylabel='averagerating'>
```



[42]:
```
#Average rating vs Romance:
#Genre "Romance" receives overall highest ratings than Comedy, Horror & Action.

#To draw bar graph:
plt.figure(figsize=(10,5))
plt.title("Averagerating vs Romance")
sns.barplot(x = 'is_romance', y = 'averagerating', data = filtered_all_df)
```

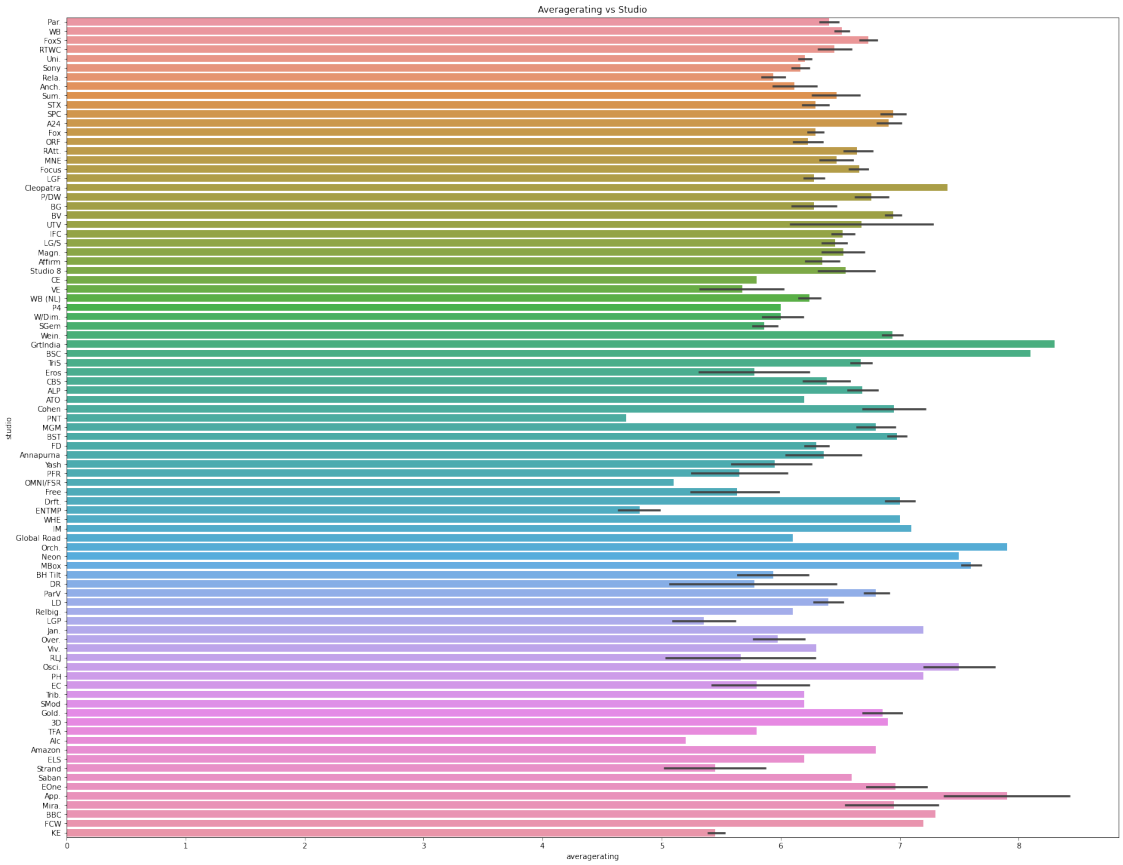
[42]:
```
<AxesSubplot:title={'center':'Averagerating vs Romance'}, xlabel='is_romance',
ylabel='averagerating'>
```



[43]:
```
#The following bar graph is to show the correlation between average rating and␣
 ↪studio.
#Movies produced in certain studio have higher ratings.
#For example, movies produced by Sony, Amazon, BBC receives =<7 ratings.

plt.figure(figsize=(25,20))
plt.title("Averagerating vs Studio")
sns.barplot(x = 'averagerating', y = 'studio', data = filtered_all_df)
```

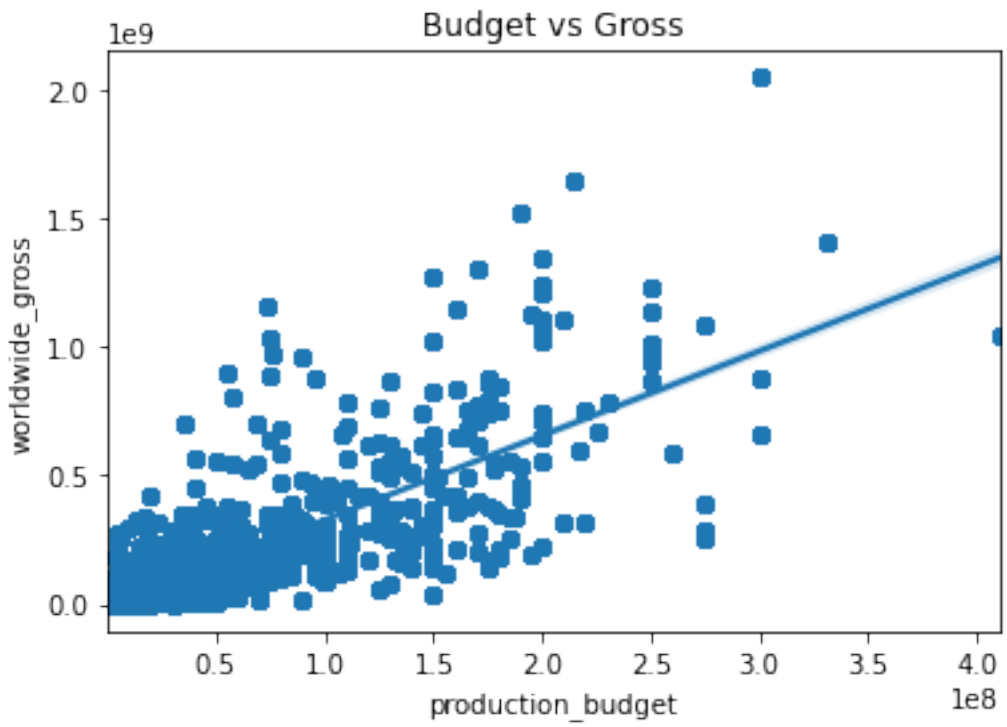
[43]:
```
<AxesSubplot:title={'center':'Averagerating vs Studio'}, xlabel='averagerating',
ylabel='studio'>
```



[44]:
```
#The following graph is to show the correlation between production_budget and
 ↪worldwide_gross
#Based on the graph, the production budget of =<$200,000,000 gives a movie
 ↪higher worldwide gross.
#Though, there are some outliers present in the data.

x=filtered_all_df.production_budget.apply(lambda x: int(x.replace(',','').
 ↪replace('$','')))
y=filtered_all_df.worldwide_gross.apply(lambda x: int(x.replace(',','').
 ↪replace('$','')))
sns.regplot(x=x, y=y)
plt.title("Budget vs Gross")
```

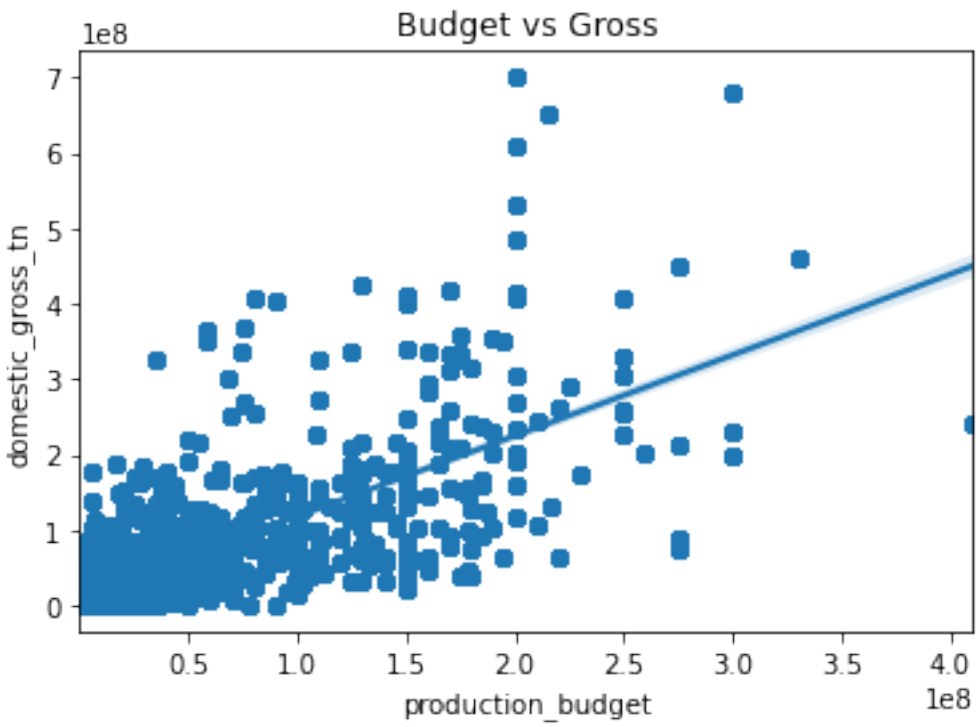
[44]:
```
Text(0.5, 1.0, 'Budget vs Gross')
```



[45]:
```
#The following graph is to show the correlation between production_budget and
 ↪domestic_gross
#Based on the graph, the production budget of =<$200,000,000 gives a movie
 ↪higher domestic gross.
#Though, there are some outliers present in the data.

x=filtered_all_df.production_budget.apply(lambda x: int(x.replace(',','').
 ↪replace('$','')))
y=filtered_all_df.domestic_gross_tn        .apply(lambda x: int(x.
 ↪replace(',','').replace('$','')))
sns.regplot(x=x, y=y)
plt.title("Budget vs Gross")
```

[45]:
```
Text(0.5, 1.0, 'Budget vs Gross')
```



[46]:
```
#Conclusions:
#Based on the analysis performed above, the conclusion can be as follows:
#Movie runtime should be between 60 minutes to 130.
#Since movie released in 2010 was effected by recession from prior years, it is␣
 ↪possible that movie released in 2022 or 2023 may be effected by Covid-19.
#Therefore, it is suggected that the targeted movie released year should be␣
 ↪2024 or onward.
#Movie ratings are based on following genre: Romance > Comedy > Action > Horror.
 ↪ Suggested genre for 1st time movie should be chosen from the 1st 3 genre.
#Movies produced in certain studio have higher ratings. Producing studio should␣
 ↪be chosen based on the ratings.
#The production budget should be <$200,000,000 to receive both domestic and␣
 ↪worldwide return.
#There are some outliers present in the data.
```

[ ]:

[ ]: