PROJECT 3:

Problem statement – You are provided with a dataset of Hollywood movies along with major information such as genre, director, release date, gross etc. Ensure the data is clean and plot the below items on a graph

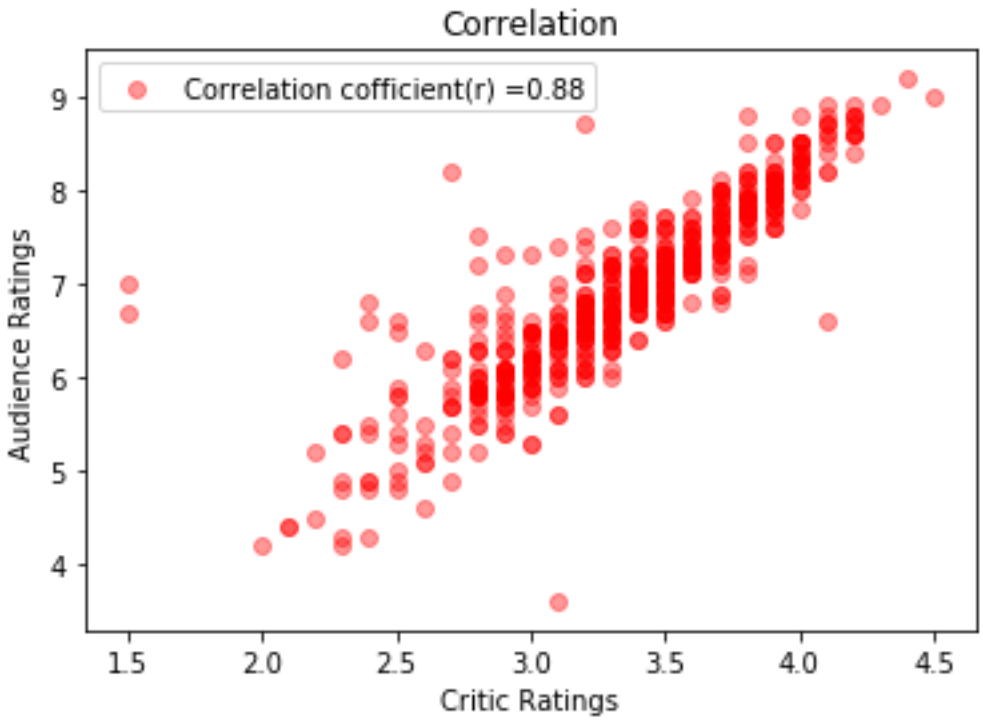
1. Find the best genre to make a movie which is more probable to be get a better rating.
2. Find the best year for each genre with respect to rating
3. Improve the data quality and ensure the data types are as per the classification of numeric and categorical variable
4. Find the destruction associated with audience rating and critics rating . Plot audience rating against the critics rating to find if there is a relation between the two variables.
5. Plot a stacked histogram to find out the budget spent for each genre of movies
6. For top 10 revenue making studios, plot the gross percentage with respect to genre and find outliers, if any.

```
@author: DHRUV
"""

#Analysis of Tinseltown


import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns



# Importing the Datafile

dataset = pd.read_csv('Movies_Structured.csv')

dataset.info()


#1&4 Establishing Basic Principals

dataset.Genre.dtypes

x = dataset['Critic Rating']
```

```
y = dataset['Audience Rating']
plt.scatter(x, y, color = 'Red', alpha = 0.4)
plt.xlabel('Critic Ratings')
plt.ylabel('Audience Ratings')
plt.title('Correlation')
plt.legend(['Correlation cofficient(r) =0.88'], loc='upper left')
```

 # This establishes that there is a very strong Co-relation between audience ratings and the critics ratings.

 # We have used audience ratings in the rest of project to reflect ratings as a whole because of this strong co-relation.

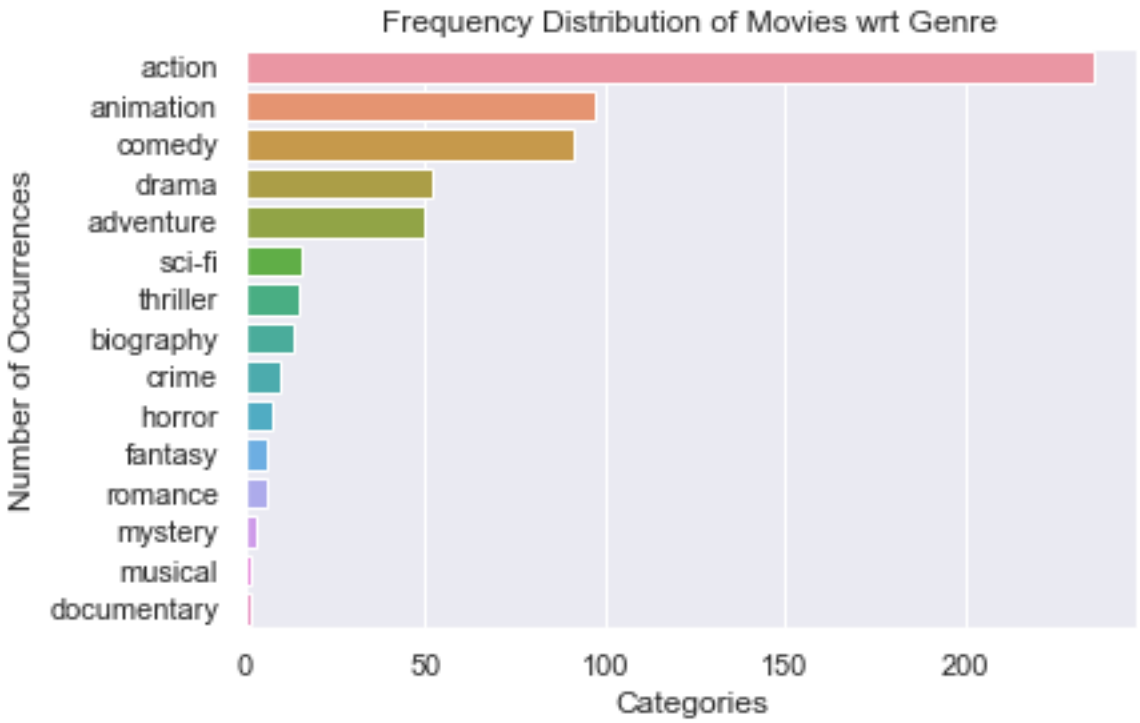


 # The following code shows the frequency distribution of genres

```
zhawruh_frequency = dataset.Genre.value_counts()
```

```
zhawruh_frequency = zhawruh_frequency.to_frame()
zhawruh_frequency = zhawruh_frequency.reset_index()
zhawruh_frequency.columns =['Genre', 'Frequency']
```

```
sns.set(style="darkgrid")
sns.barplot(zhawruh_frequency.Frequency, zhawruh_frequency.Genre, alpha=1.0)
plt.title('Frequency Distribution of Movies wrt Genre')
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Categories', fontsize= 12)
plt.show()
```



Q1 : Find the best genre to make a movie which is more probable to be get a better rating.

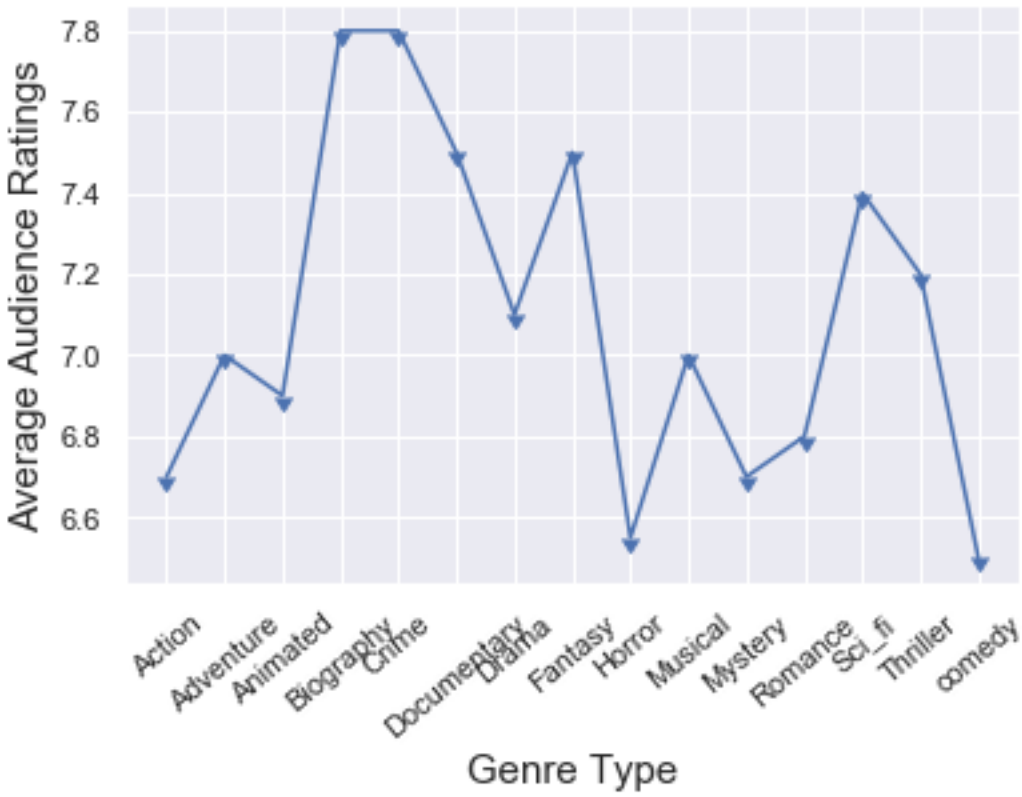
A1 **The following code shows the average audience ratings with respect to various Genres**

```
comedy = dataset[dataset.Genre == 'comedy']
comedy = comedy.loc[:608 ,['Genre', 'Audience Rating']]
comedy = comedy.reset_index()
comedy = comedy.drop(['index'], axis=1)

thriller = dataset[dataset.Genre == 'thriller']
thriller = thriller.loc[:608, ['Genre', 'Audience Rating']]
thriller = thriller.reset_index()
thriller = thriller.drop(['index'], axis=1)



Action = dataset[dataset.Genre == 'action']
Action = Action.loc[:608, ['Genre','Audience Rating']]
Action = Action.reset_index()
Action  = Action.drop(['index'], axis=1)

Animated = dataset[dataset.Genre == 'animation']
Animated = Animated.loc[:608, ['Genre', 'Audience Rating']]
Animated = Animated.reset_index()
Animated = Animated.drop(['index'], axis=1)

Drama = dataset[dataset.Genre == 'drama']
Drama = Drama.loc[:608, ['Genre', 'Audience Rating']]
Drama = Drama.reset_index()
Drama = Drama.drop(['index'], axis=1)

Adventure = dataset[dataset.Genre == 'adventure']
Adventure = Adventure.loc[:608, ['Genre', 'Audience Rating']]
Adventure = Adventure.reset_index()
```

```
Adventure = Adventure.drop(['index'], axis=1)


Sci_fi = dataset[dataset.Genre == 'sci-fi']
Sci_fi = Sci_fi.loc[:608, ['Genre', 'Audience Rating']]
Sci_fi = Sci_fi.reset_index()
Sci_fi = Sci_fi.drop(['index'], axis=1)



Biography = dataset[dataset.Genre == 'biography']
Biography = Biography.loc[:608, ['Genre', 'Audience Rating']]
Biography = Biography.reset_index()
Biography = Biography.drop(['index'], axis=1)



Crime = dataset[dataset.Genre == 'crime']
Crime = Crime.loc[:608, ['Genre', 'Audience Rating']]
Crime = Crime.reset_index()
Crime = Crime.drop(['index'], axis=1)

Horror = dataset[dataset.Genre == 'horror']
Horror = Horror.loc[:608, ['Genre', 'Audience Rating']]
Horror = Horror.reset_index()
Horror = Horror.drop(['index'], axis=1)



Romance = dataset[dataset.Genre == 'romance']
Romance = Romance.loc[:608, ['Genre', 'Audience Rating']]
Romance = Romance.reset_index()
Romance = Romance.drop(['index'], axis=1)
```

```
Fantasy = dataset[dataset.Genre == 'fantasy']
Fantasy = Fantasy.loc[:608, ['Genre', 'Audience Rating']]
Fantasy = Fantasy.reset_index()
Fantasy = Fantasy.drop(['index'], axis=1)


Mystery = dataset[dataset.Genre == 'mystery']
Mystery = Mystery.loc[:608, ['Genre', 'Audience Rating']]
Mystery = Mystery.reset_index()
Mystery = Mystery.drop(['index'], axis=1)


Musical = dataset[dataset.Genre == 'musical']
Musical = Musical.loc[:608, ['Genre', 'Audience Rating']]
Musical = Musical.reset_index()
Musical = Musical.drop(['index'], axis=1)


Documentary = dataset[dataset.Genre == 'documentary']
Documentary = Documentary.loc[:608, ['Genre', 'Audience Rating']]
Documentary = Documentary.reset_index()
Documentary = Documentary.drop(['index'], axis=1)


My_Vocab = {}
My_Vocab  ={'Action': 6.7,
       'comedy': 6.5,
```

```
    'Horror': 6.55,
    'Thriller': 7.2,
    'Animated' :6.9,
    'Sci_fi' : 7.4,
    'Biography' : 7.8,
    'Adventure' : 7.0,
    'Crime' : 7.8,
    'Documentary' : 7.5,
    'Drama' : 7.1,
    'Fantasy' : 7.5,
    'Musical' : 7.0,
    'Mystery' : 6.7,
    'Romance' : 6.8
    }

plt.plot(*zip(*sorted(My_Vocab.items())),marker = 11)
plt.xticks(rotation = 40)
plt.xlabel('Genre Type', fontsize = 15)
plt.ylabel('Average Audience Ratings', fontsize = 15)
plt.show()
```

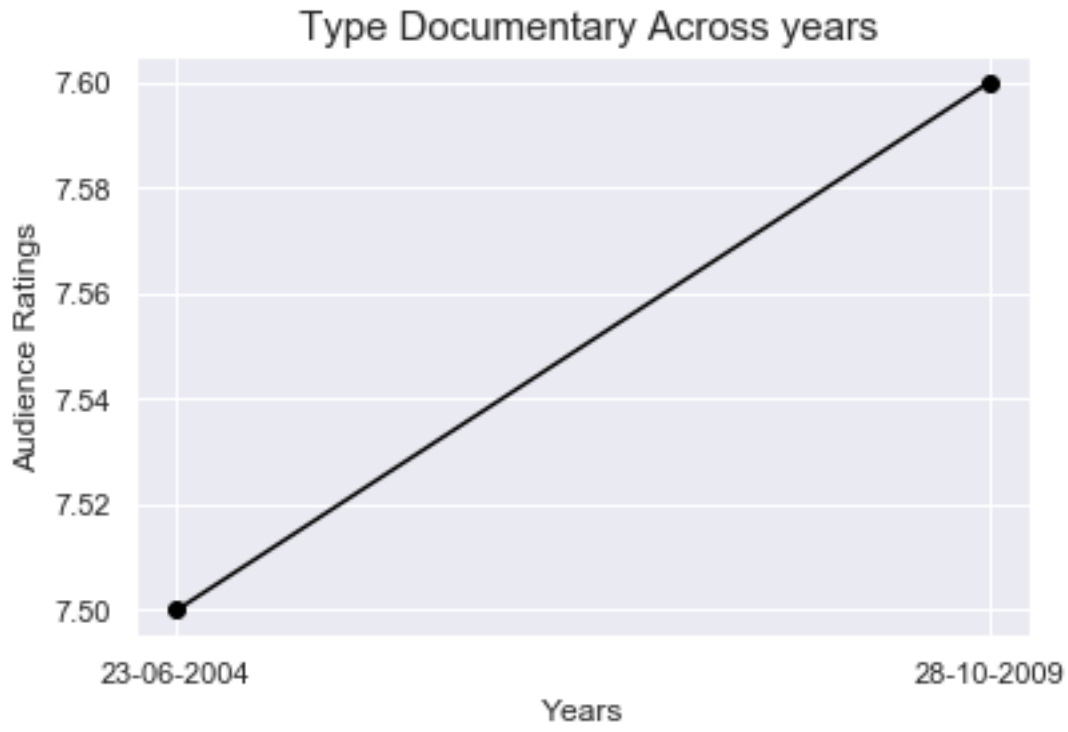


As you can clearly infer from the above genres Biography & Crime have the best avg ratings (7.8)

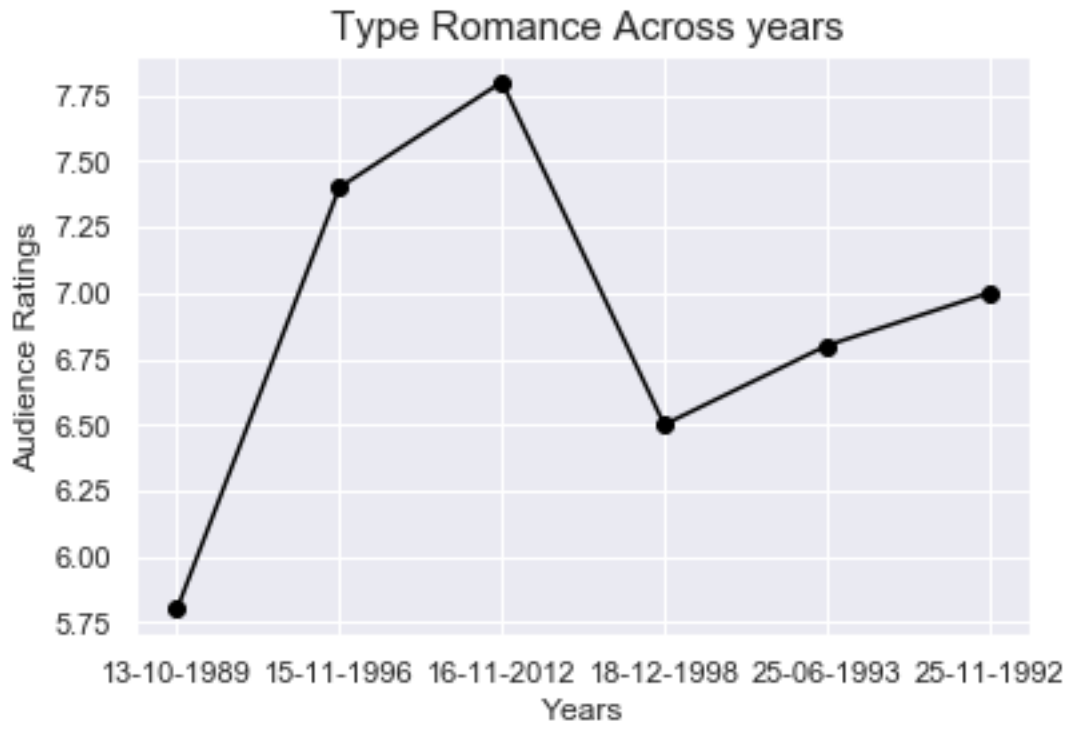
Q2 Find the best year for each genre with respect to rating

A2 – We will plot each genre across the years and use the output to evaluate the best year for each genre

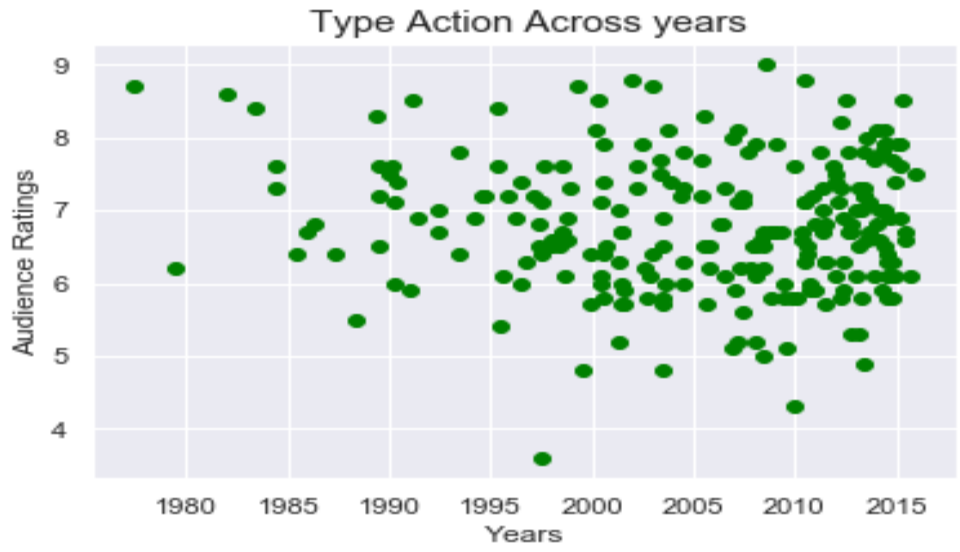
```
Doc = dataset[dataset.Genre == 'documentary']
Doc = Doc.loc[:608, ['Release Date', 'Audience Rating']]
Doc.columns = ['Release_Date', 'Audience_Rating']
plt.title(' Type Documentary Across years', fontsize = 15)
plt.xlabel('Years')
plt.ylabel('Audience Ratings')
plt.plot(Doc.Release_Date, Doc.Audience_Rating, marker = 'o', color = 'Black')
```




```
Ro = dataset[dataset.Genre == 'romance']

Ro = Ro.loc[:608, ['Release Date', 'Audience Rating']]

Ro.columns = ['Release_Date', 'Audience_Rating']

Ro = Ro.sort_values('Release_Date')

plt.title(' Type Romance Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(Ro.Release_Date, Ro.Audience_Rating, marker = 'o', color = 'Black')

plt.figure(figsize=(12,24))
```

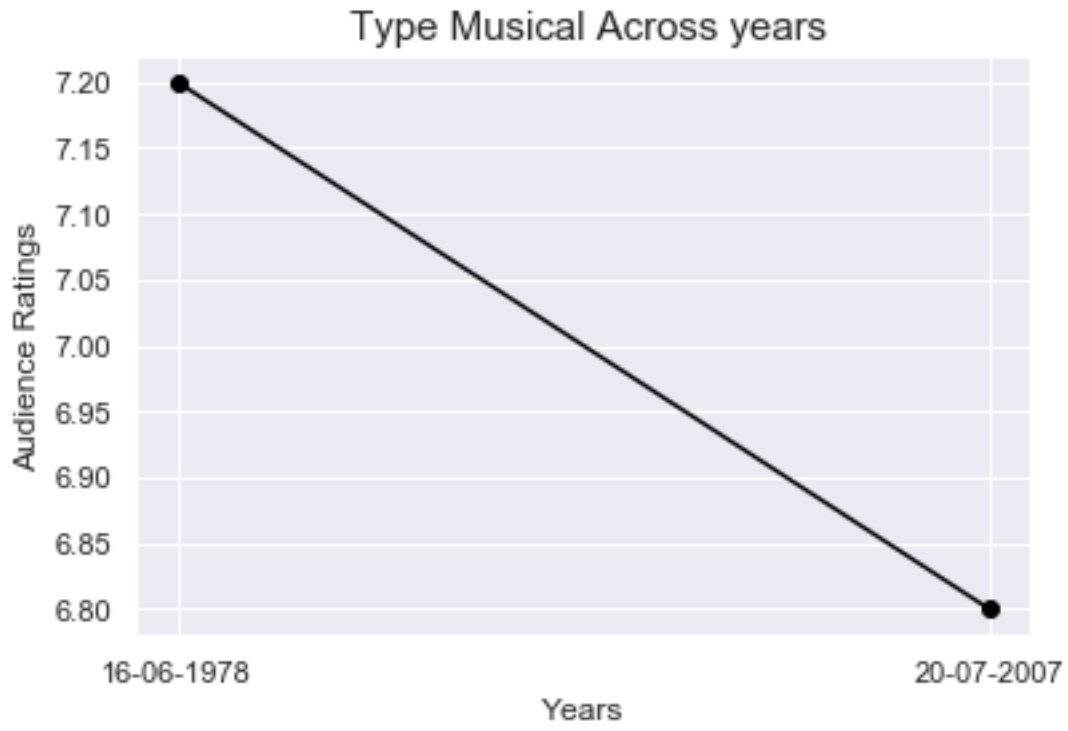



```
Act = dataset[dataset.Genre == 'action']

Act = Act.loc[:608, ['Release Date', 'Audience Rating']]

Act.columns = ['Release_Date', 'Audience_Rating']

plt.title(' Type Action Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

sorted_values = Act.sort_values('Release_Date')

#plt.plot(sorted_values.Release_Date, sorted_values.Audience_Rating, color = 'Green')

plt.scatter(sorted_values.Release_Date, sorted_values.Audience_Rating, color = 'Green')
```

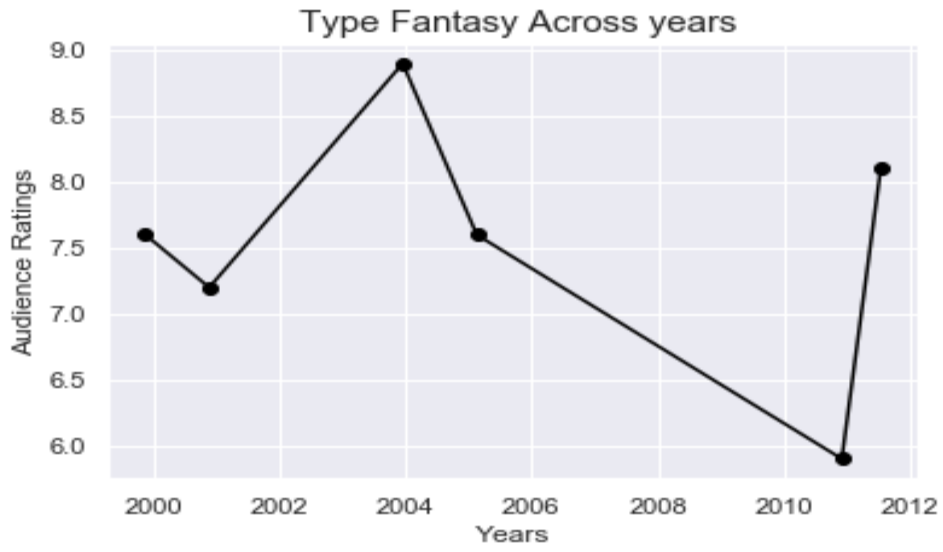



```
Myst = dataset[dataset.Genre == 'mystery']

Myst = Myst.loc[:608, ['Release Date', 'Audience Rating']]

Myst.columns = ['Release_Date', 'Audience_Rating']

Myst = Myst.sort_values('Release_Date')

plt.title(' Type Mystery Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(Myst.Release_Date, Myst.Audience_Rating, color = 'Black', marker ='o')
```

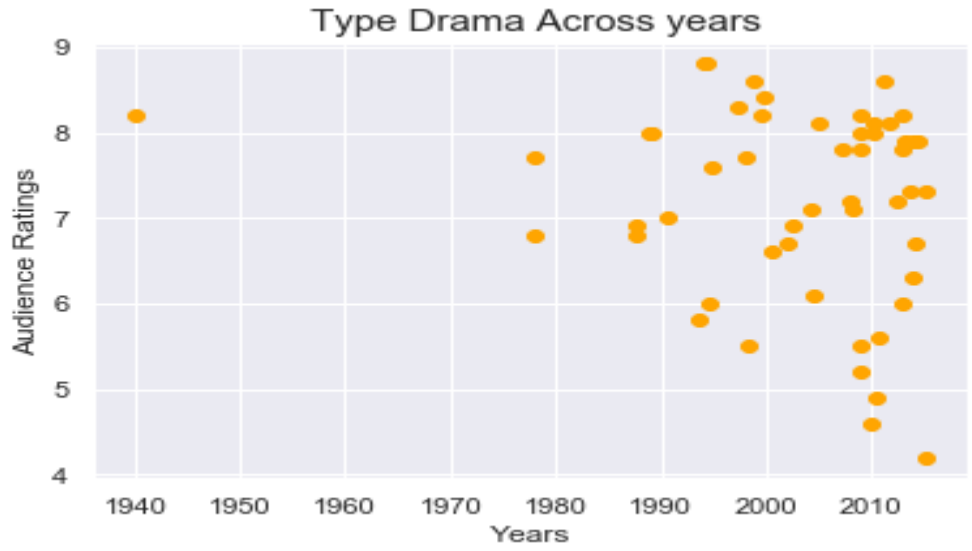

```
Mus = dataset[dataset.Genre == 'musical']

Mus = Mus.loc[:608, ['Release Date', 'Audience Rating']]

Mus.columns = ['Release_Date', 'Audience_Rating']

Mus = Mus.sort_values('Release_Date')

plt.title(' Type Musical Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(Mus.Release_Date, Mus.Audience_Rating, color = 'Black', marker ='o')
```



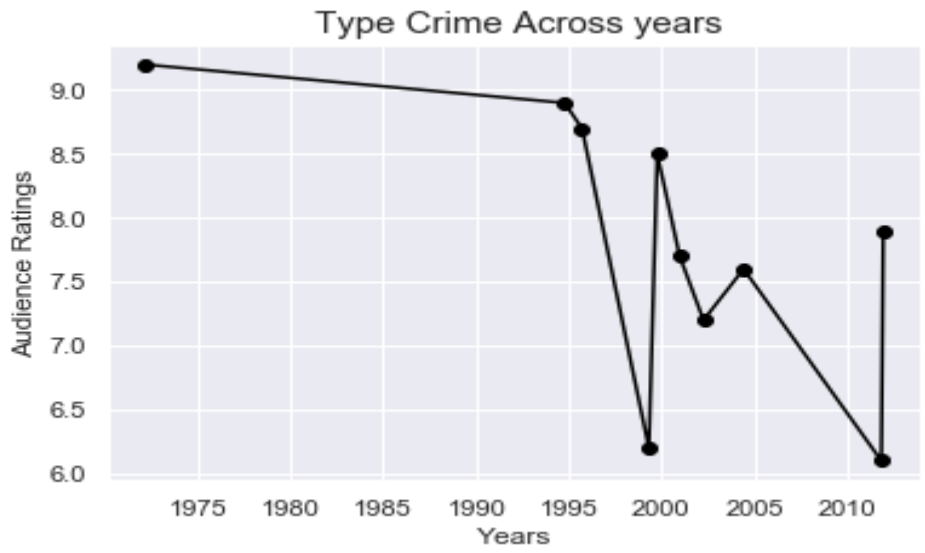

```
Fan = dataset[dataset.Genre == 'fantasy']

Fan = Fan.loc[:608, ['Release Date', 'Audience Rating']]

Fan.columns = ['Release_Date', 'Audience_Rating']

Fan = Fan.sort_values('Release_Date')

plt.title(' Type Fantasy Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(Fan.Release_Date, Fan.Audience_Rating, color = 'Black', marker ='o')
```



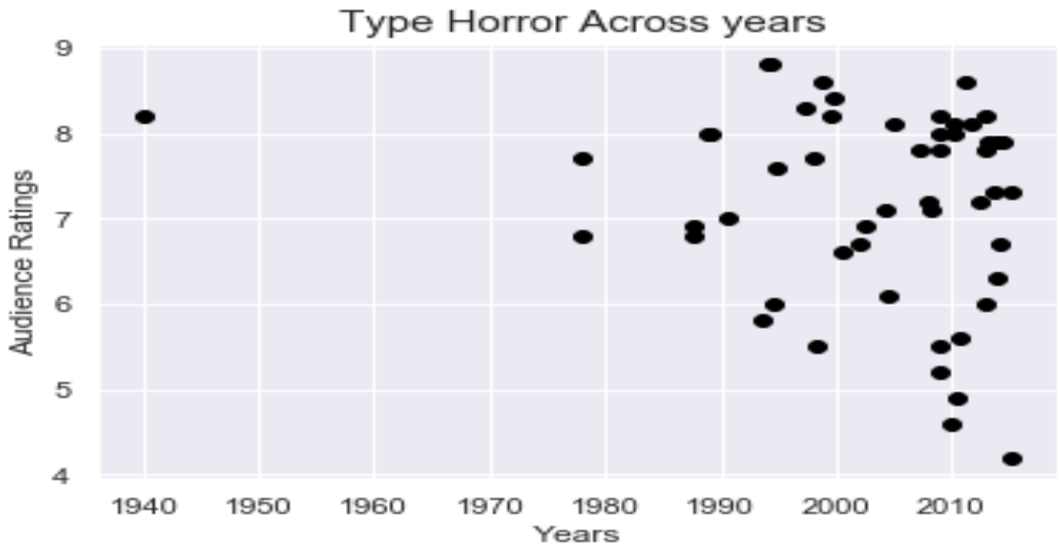

```
Dra = dataset[dataset.Genre == 'drama']

Dra = Dra.loc[:608, ['Release Date', 'Audience Rating']]

Dra.columns = ['Release_Date', 'Audience_Rating']

Dra = Dra.sort_values('Release_Date')

plt.title(' Type Drama Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.scatter(Dra.Release_Date, Dra.Audience_Rating, color = 'Orange')
```



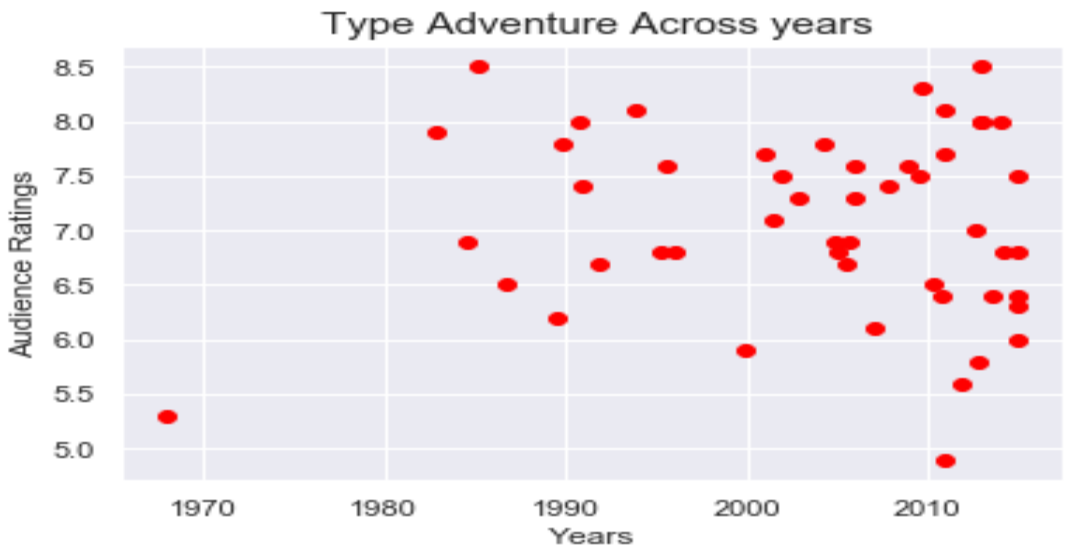
```
Cri = dataset[dataset.Genre == 'crime']
Cri = Cri.loc[:608, ['Release Date', 'Audience Rating']]
Cri.columns = ['Release_Date', 'Audience_Rating']
Cri = Cri.sort_values('Release_Date')
plt.title(' Type Crime Across years', fontsize = 15)
plt.xlabel('Years')
plt.ylabel('Audience Ratings')
plt.plot(Cri.Release_Date, Cri.Audience_Rating, color = 'Black', marker ='o')
```



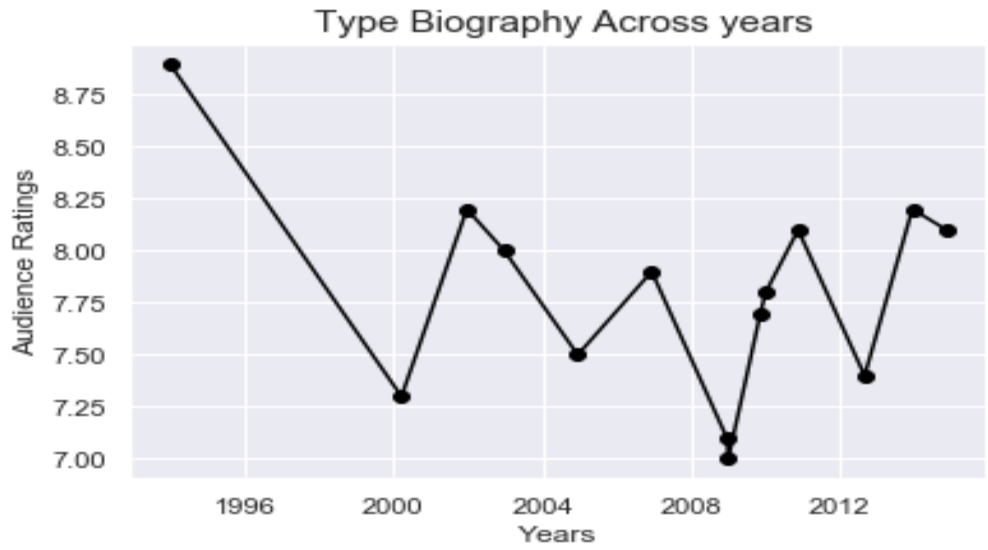

```
Hor = dataset[dataset.Genre == 'horror']
Hor = Hor.loc[:608, ['Release Date', 'Audience Rating']]
Hor.columns = ['Release_Date', 'Audience_Rating']
Hor = Hor.sort_values('Release_Date')
plt.title(' Type Horror Across years', fontsize = 15)
plt.xlabel('Years')
plt.ylabel('Audience Ratings')
plt.plot(Dra.Release_Date, Dra.Audience_Rating, color = 'Black', marker ='o')
```



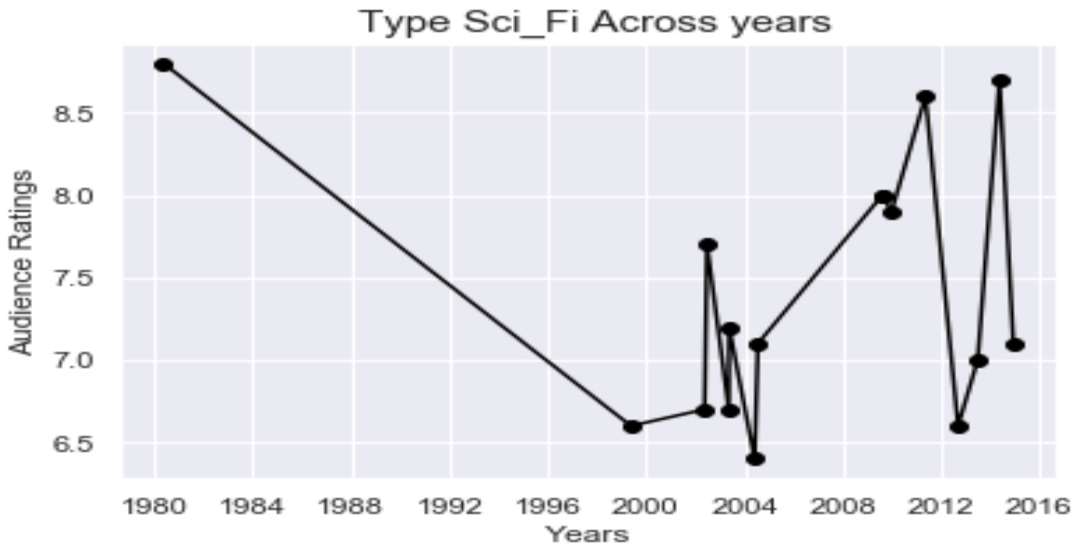

```
Adv = dataset[dataset.Genre == 'adventure']

Adv = Adv.loc[:608, ['Release Date', 'Audience Rating']]

Adv.columns = ['Release_Date', 'Audience_Rating']

Adv = Adv.sort_values('Release_Date')

plt.title(' Type Adventure Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.scatter(Adv.Release_Date, Adv.Audience_Rating, color = 'Red', marker ='o')
```



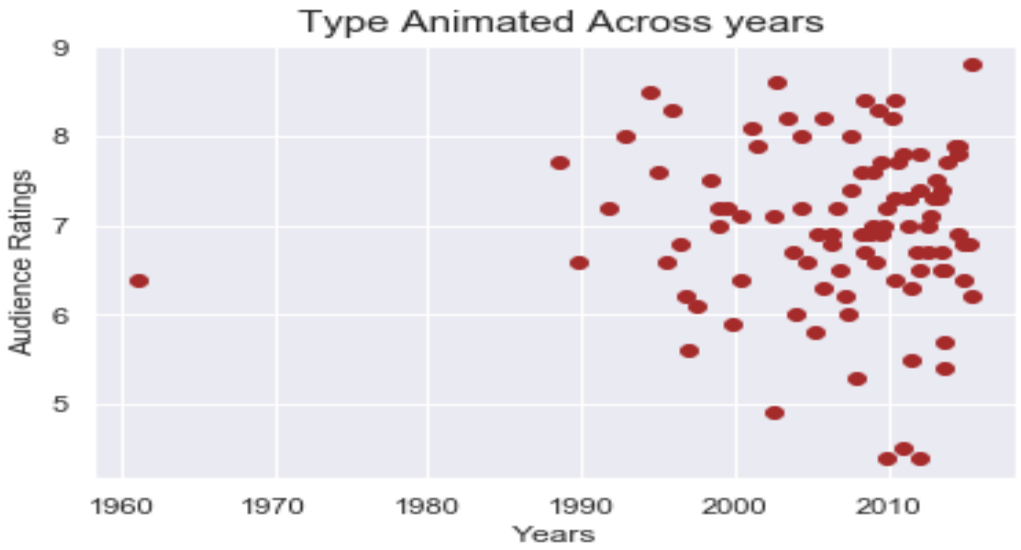

```
Bio = dataset[dataset.Genre == 'biography']

Bio = Bio.loc[:608, ['Release Date', 'Audience Rating']]

Bio.columns = ['Release_Date', 'Audience_Rating']

Bio = Bio.sort_values('Release_Date')

plt.title(' Type Biography Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(Bio.Release_Date, Bio.Audience_Rating, color = 'Black', marker ='o')

plt.figure(figsize = (42,36))
```



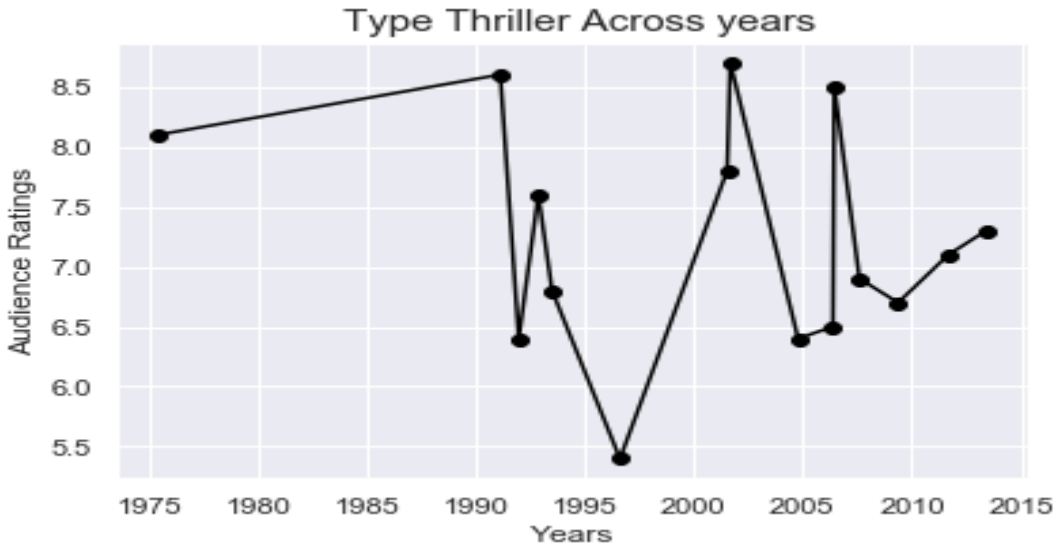

```
dde = dataset[dataset.Genre == 'sci-fi']

dde = dde.loc[:608, ['Release Date', 'Audience Rating']]

dde.columns = ['Release_Date', 'Audience_Rating']

dde = dde.sort_values('Release_Date')

plt.title(' Type Sci_Fi Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(dde.Release_Date, dde.Audience_Rating, color = 'Black', marker ='o')
```



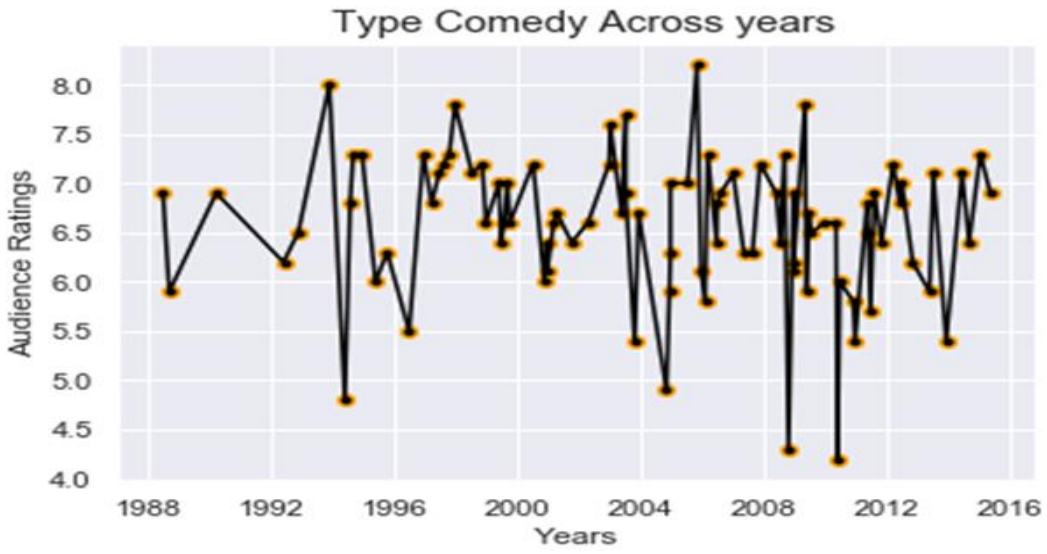

```
Ani = dataset[dataset.Genre == 'animation']

Ani = Ani.loc[:608, ['Release Date', 'Audience Rating']]

Ani.columns = ['Release_Date', 'Audience_Rating']

Ani = Ani.sort_values('Release_Date')

plt.title(' Type Animated Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.scatter(Ani.Release_Date, Ani.Audience_Rating, color = 'Black')
```




```
Thr = dataset[dataset.Genre == 'thriller']

Thr = Thr.loc[:608, ['Release Date', 'Audience Rating']]

Thr.columns = ['Release_Date', 'Audience_Rating']

Thr = Thr.sort_values('Release_Date')

plt.title(' Type Thriller Across years', fontsize = 15)

plt.xlabel('Years')

plt.ylabel('Audience Ratings')

plt.plot(Thr.Release_Date, Thr.Audience_Rating, color = 'Black', marker ='o')
```



```
Com = dataset[dataset.Genre == 'comedy']
Com = Com.loc[:608, ['Release Date', 'Audience Rating']]
Com.columns = ['Release_Date', 'Audience_Rating']
Com = Com.sort_values('Release_Date')
plt.title(' Type Comedy Across years', fontsize = 15)
plt.xlabel('Years')
plt.ylabel('Audience Ratings')
plt.scatter(Com.Release_Date, Com.Audience_Rating, color = 'Orange')
plt.plot(Com.Release_Date, Com.Audience_Rating, color = 'Black', marker = '.')
```



Q3 Improve the data quality and ensure the data types are as per the classification of numeric and categorical variable

A3 The following code optimizes the fields and ensures that the types are in accordance with the data that is captured.

```
dataset[['Movie Title', 'Director','Genre',
    'Release Day', 'Studio']] = dataset[['Movie Title', 'Director', 'Genre',
```

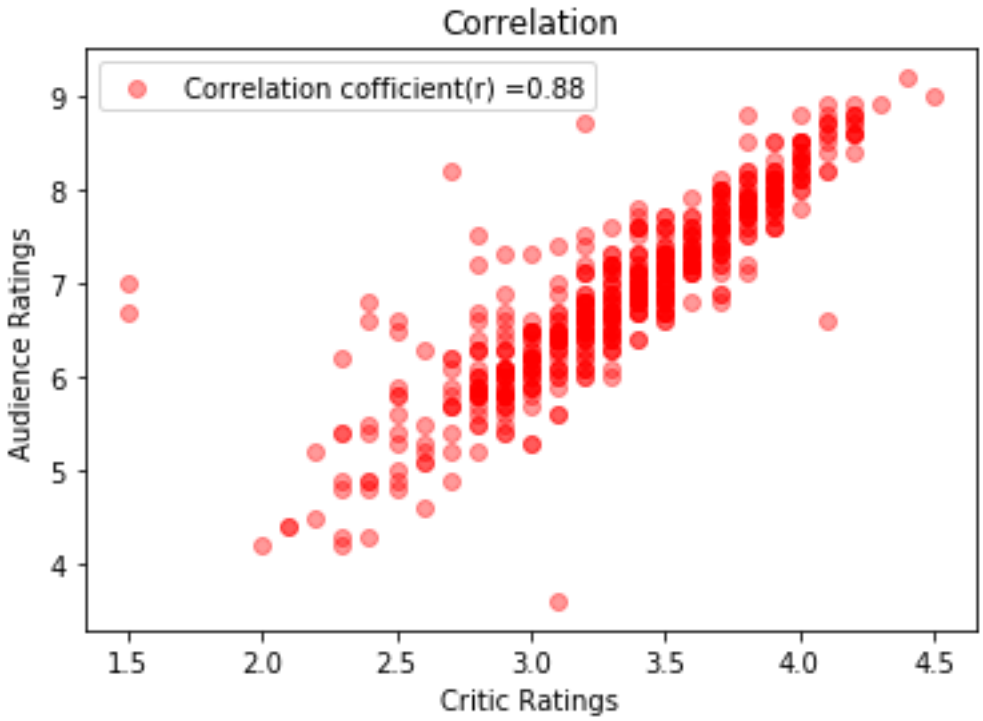
```
        'Release Day', 'Studio']].astype('category')
dataset['Release Date']= pd.to_datetime(dataset['Release Date'])



dataset['Adjusted Gross ($mill)', 'Budget ($mill)', 'Gross ($mill)',
       'Overseas ($mill)', 'Overseas %', 'Profit ($mill)',
       'Profit %', 'Runtime', 'US ($mill)',
       'Gross % US'] = pd.to_numeric(dataset['Adjusted Gross ($mill)',
       'Budget ($mill)', 'Gross ($mill)', 'Overseas ($mill)',
       'Overseas %', 'Profit ($mill)',
       'Profit %', 'Runtime', 'US ($mill)', 'Gross % US'])
dataset[['Audience Rating',
        'Critic Rating']] = dataset[['Audience Rating',
        'Critic Rating']].astype(float)


#dataset.info()
dataset.dtypes
```

Q4 Find the destruction associated with audience rating and critics rating . Plot audience rating against the critics rating to find if there is a relation between the two variables.

A4 Using the code below we have plotted a graph of audience rating vs critic rating which establishes that there is a very strong Co-relation between the two ratings. The co-relation coefficient is 0.88 (difference from 1 is only 0.12) hence these two ratings nearly mirror each other.

```
dataset.Genre.dtypes
x = dataset['Critic Rating']
y = dataset['Audience Rating']
plt.scatter(x, y, color = 'Red', alpha = 0.4)
plt.xlabel('Critic Ratings')
```

```
plt.ylabel('Audience Ratings')

plt.title('Correlation')

plt.legend(['Correlation cofficient(r) =0.88'], loc='upper left')
```

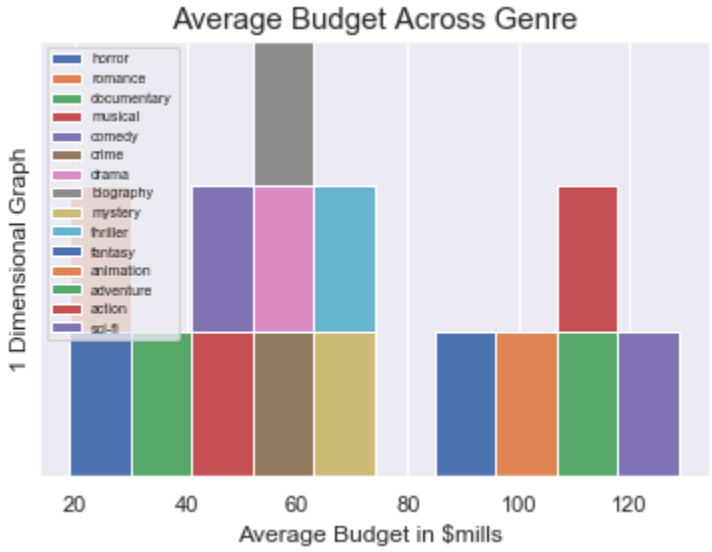


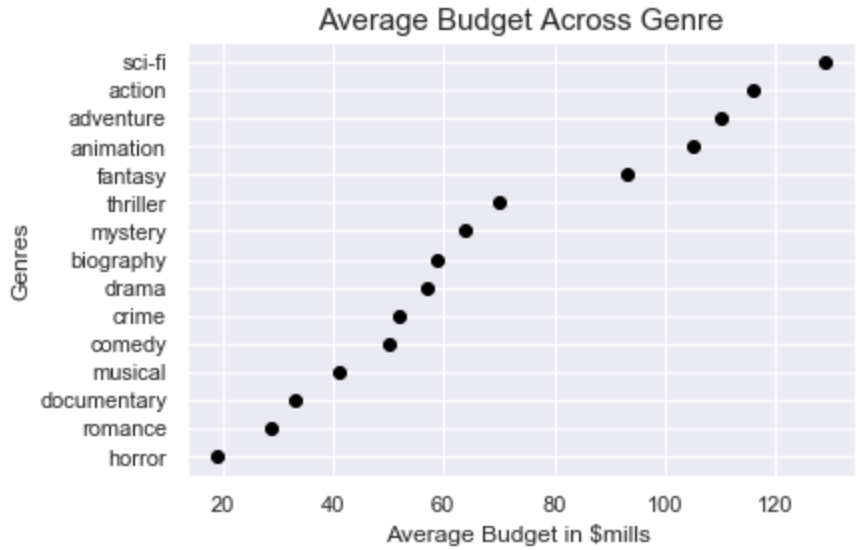
Q5 Plot a stacked histogram to find out the budget spent for each genre of movies

```
Genre_Budget = dataset.Genre.value_counts()
Genre_Budget = pd.DataFrame(Genre_Budget)
Genre_Budget = Genre_Budget.drop(columns = ['Genre'])
Genre_Budget = Genre_Budget.reset_index()
Genre_Budget.columns = ['GENRE']
Genre_Budget['Budget ($mill)'] = [27474, 10144, 4540,
        2937, 5467, 2061,
        1049, 818, 522,
        150, 175, 557,
        191, 81, 66]
Genre_Budget['Average_Buget ($mill)'] = [116, 105, 50,
        57, 110, 129,
        70, 59, 52, 19,
        29, 93, 64,
```

41, 33]

```
#Uncomment As per your Requirement
#Genre_Budget.columns = ['GENRE', 'Budget', 'Avgbudget']
Genre_Budget.columns = ['GENRE', 'Avgbudget']
#Genre_Budget = Genre_Budget.sort_values('Budget')
sns.set(style="whitegrid")
#Genre_Budget = Genre_Budget.drop(columns=['Budget'])
Genre_Budget = Genre_Budget.sort_values('Avgbudget')
#Genre_Budget = Genre_Budget.sort_values('Budget')
Genre_Budget.set_index('GENRE').T.plot(kind='Hist', stacked=True)
plt.legend(loc = 'upper right')
plt.title('Average Budget Across Genre', fontsize = 15)
plt.xlabel('Average Budget in $mills ', fontsize=12)
plt.ylabel('Genres', fontsize = 12)
plt.scatter(Genre_Budget.Avgbudget, Genre_Budget.GENRE, color = 'Black')
#plt.scatter(Genre_Budget.Budget, Genre_Budget.GENRE, color = 'black')
Genre_Budget.set_index('GENRE').T.plot(kind='Hist', stacked=True)
plt.yticks([]) #removing the Y axis
plt.ylabel('1 Dimensional Graph', fontsize = 12)
plt.title('Average Budget Across Genre', fontsize = 15)
plt.xlabel('Average Budget in $mills ', fontsize=12)
```



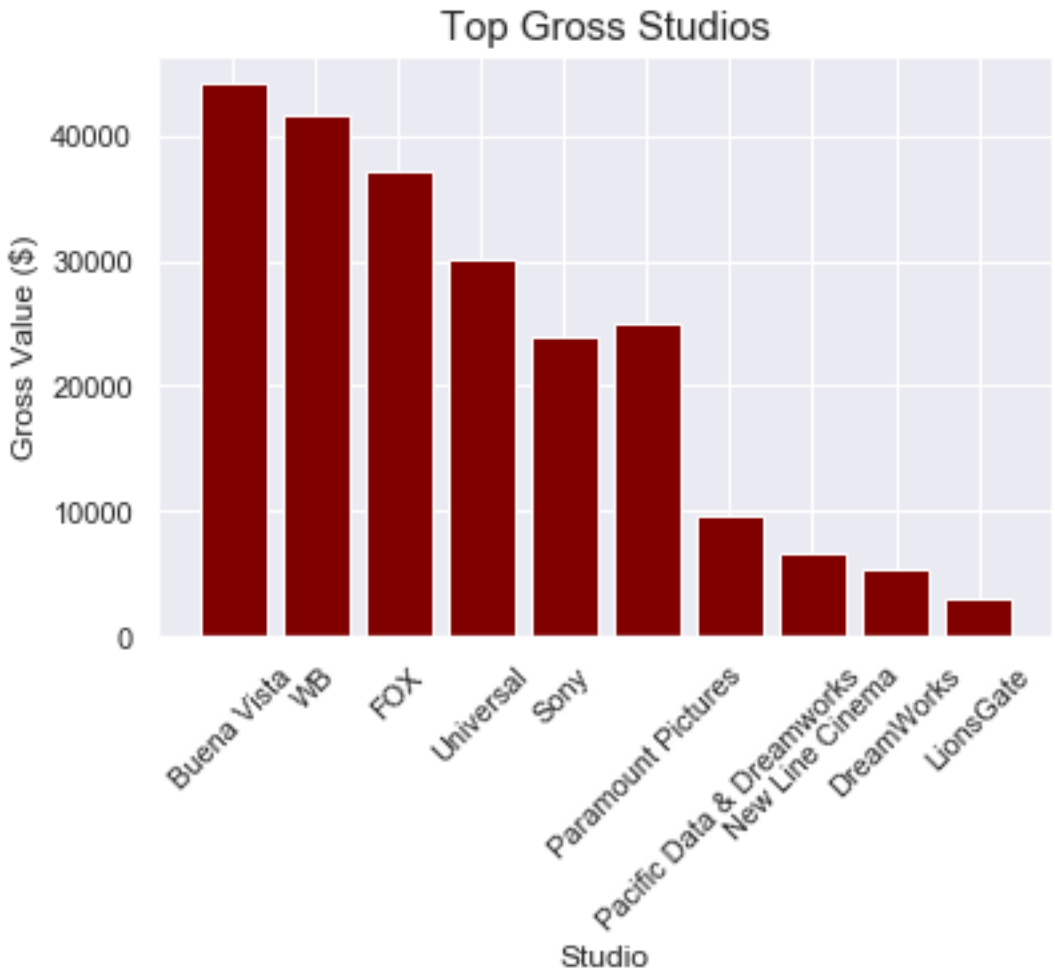


Q6 For top 10 revenue making studios, plot the gross percentage with respect to genre and find outliers, if any.

A6 The approach used was to first identify the top 10 revenue grossers and then plot the average gross percentage of each genre for every top revenue grossing studio

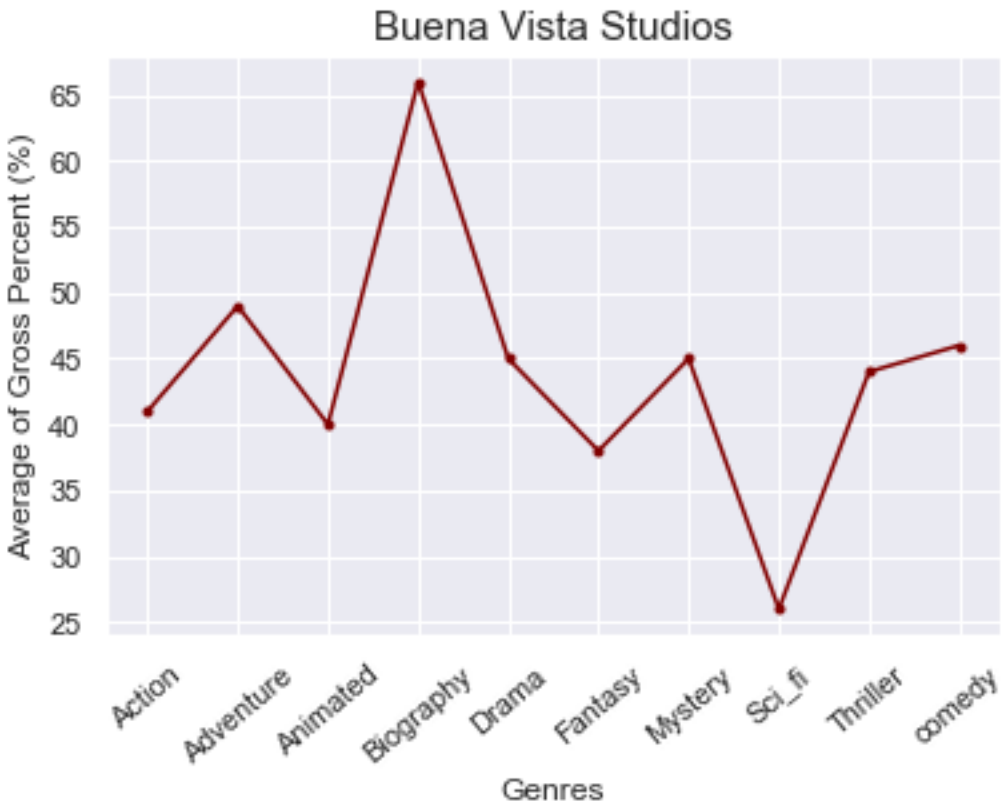
```
Studio_Frequency = dataset.Studio.value_counts()

Studio_Frequency = Studio_Frequency.reset_index()

Studio_Frequency.columns = ['Studio', 'Frequncy']


My_Vocab1 = {}

My_Vocab1  ={'Buena Vista': 44194,

        'WB': 41669,

        'FOX': 37217,

        'Universal': 30133,

        'Sony' : 23861,

        'Paramount Pictures' : 25077,

        'Pacific Data & Dreamworks' : 9556,

        'New Line Cinema' : 6584,

        'DreamWorks' : 5373,
```

```
    'LionsGate' : 3094
    }
plt.bar(My_Vocab1.keys(), My_Vocab1.values(), color='Maroon')
plt.xticks(rotation = 45)
plt.xlabel('Studio', fontsize = 12)
plt.ylabel('Gross Value ($)', fontsize = 12)
plt.title('Top Gross Studios', fontsize = 15)
plt.show()
```




```
Gross_Percent = pd.DataFrame(My_Vocab1,index=['i',])
Gross_Percent = Gross_Percent.transpose()
```

```
Gross_Percent = Gross_Percent.reset_index()
Gross_Percent.columns = ['Top Studios', 'XYZ']
Gross_Percent = Gross_Percent.drop(axis = 1, columns = 'XYZ')



My_Vocab_BV  ={'Action': 41,
        'comedy': 46,
        'Thriller': 44,
        'Animated' :40,
        'Sci_fi' : 26,
        'Biography' : 66,
        'Adventure' : 49,
        'Drama' : 45,
        'Fantasy' : 38,
        'Mystery' : 45
        }
plt.plot(*zip(*sorted(My_Vocab_BV.items())),marker = '.', color = 'maroon')
plt.xticks(rotation = 40)
plt.xlabel('Genres', fontsize = 12)
plt.ylabel('Average of Gross Percent (%)', fontsize = 12)
plt.title('Buena Vista Studios', fontsize = 15)
plt.show()
```



My_Vocab_WB ={'Action': 43,

'comedy': 47,

'Horror': 48,

'Thriller': 39,

'Animated' : 52,

'Sci_fi' : 35,

'Biography' : 83,

'Adventure' : 36,

'Crime' : 48,

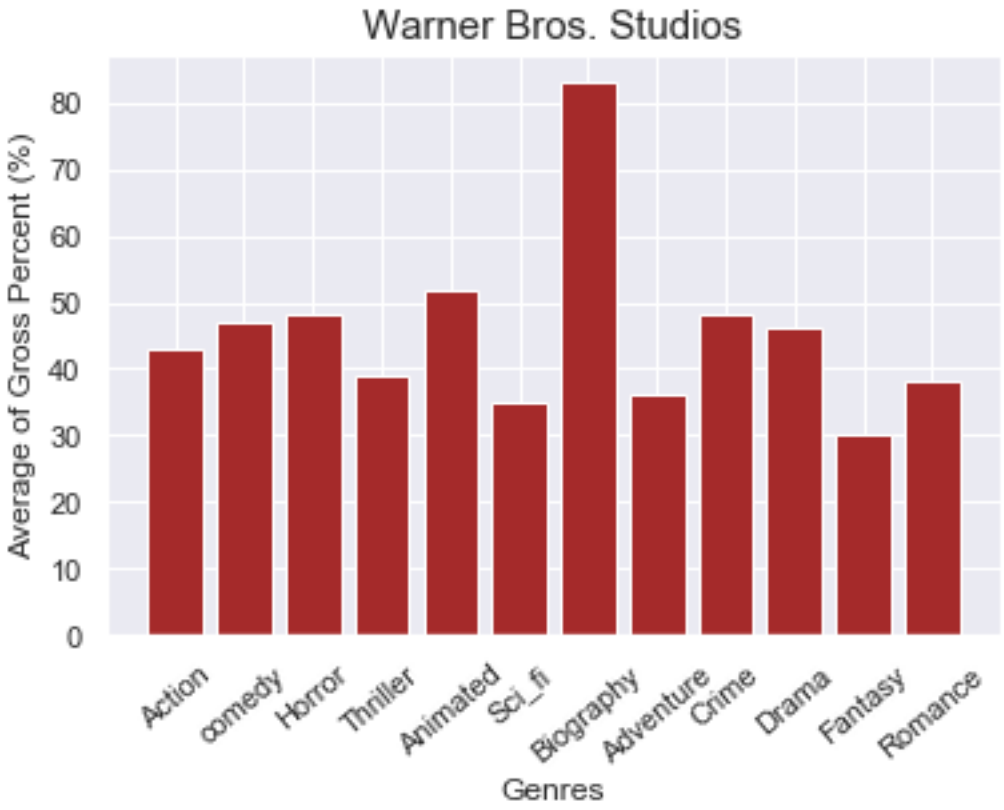
'Drama' : 46,

'Fantasy' : 30,

'Romance' : 38

}

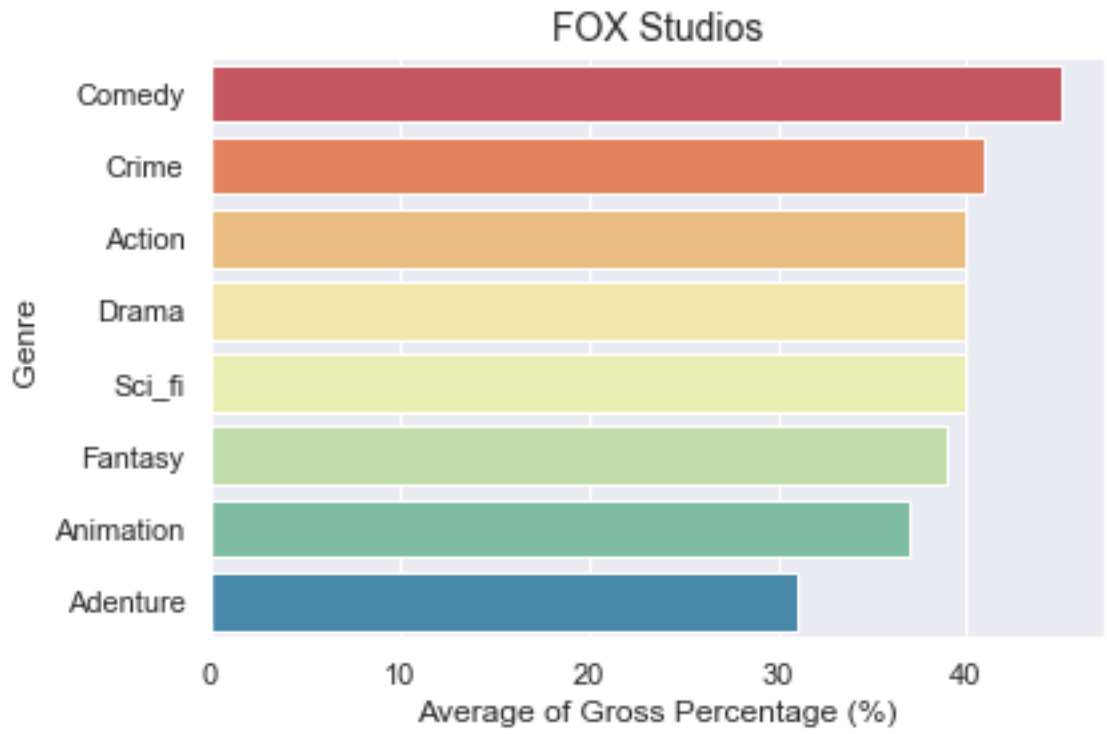
```
plt.bar(My_Vocab_WB.keys(), My_Vocab_WB.values(), color='Brown')
plt.xticks(rotation = 40)
plt.xlabel('Genres', fontsize = 12)
plt.ylabel('Average of Gross Percent (%)', fontsize = 12)
plt.title('Warner Bros. Studios', fontsize = 15)
```




```
Fox_Frame = pd.DataFrame()
Fox_Frame['Genre'] = ['Action', 'Adenture', 'Animation',
        'Comedy', 'Crime','Drama', 'Fantasy', 'Sci_fi']
Fox_Frame['Value'] = [40,31,37,45,41,40,39,40]
Fox_Frame = Fox_Frame.sort_values('Value', ascending = False)
Fox_Frame = Fox_Frame.reset_index()
```

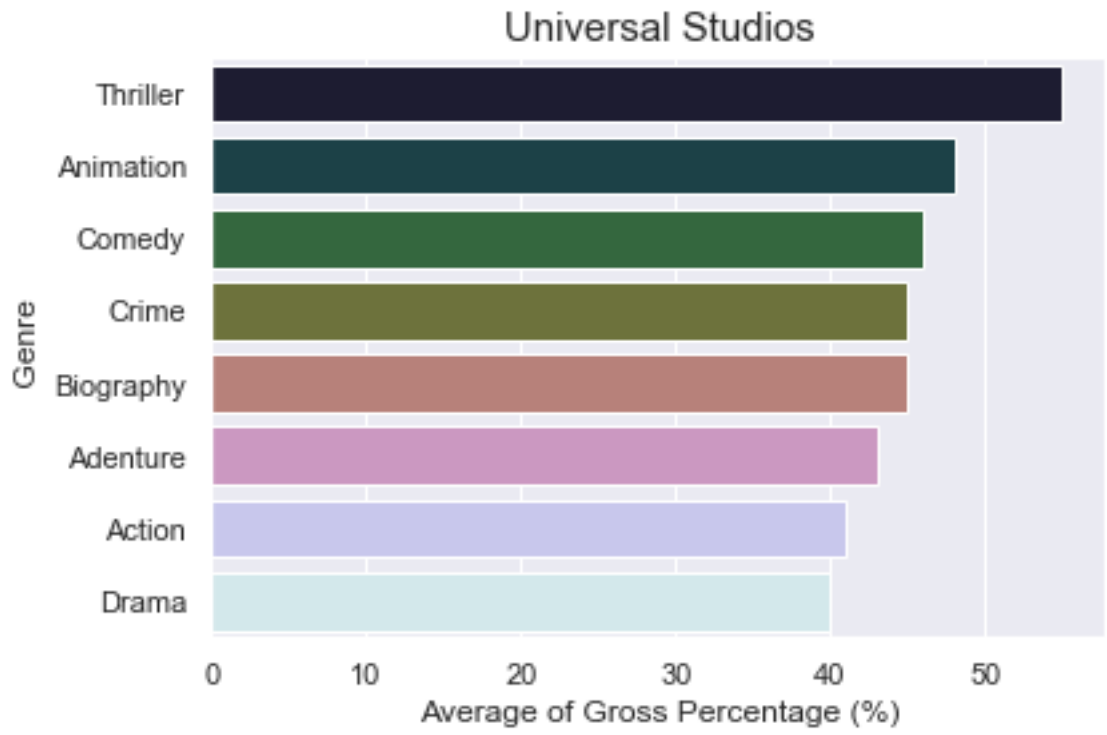
```
Fox_Frame = Fox_Frame.drop(columns = ['index'])
```

```
sns.barplot(Fox_Frame.Value, Fox_Frame.Genre, palette = 'Spectral')
plt.xlabel('Average of Gross Percentage (%)',fontsize = 12)
plt.ylabel(' Genre',fontsize = 12)
plt.title('FOX Studios',fontsize = 14)
```



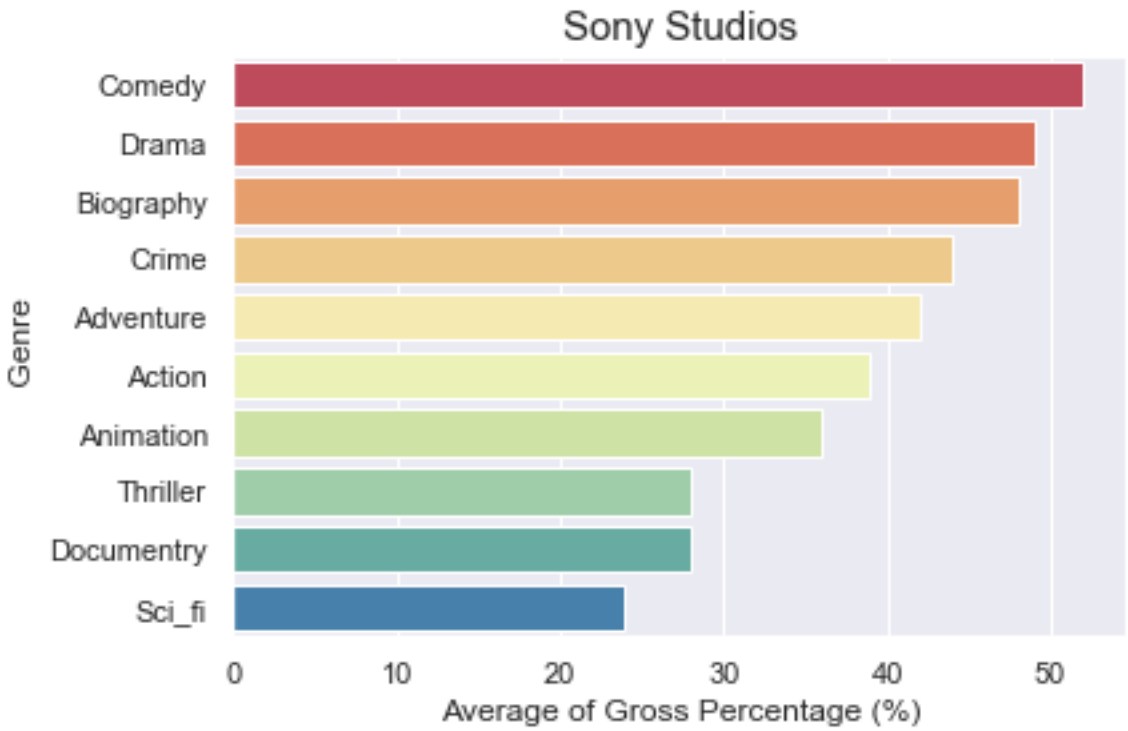

```
Universal_Frame = pd.DataFrame()
Universal_Frame['Genre'] = ['Action', 'Adenture', 'Animation',
      'Comedy', 'Crime','Drama', 'Thriller', 'Biography']
Universal_Frame['Value'] = [41,43,48,46,45,40,55,45]
Universal_Frame = Universal_Frame.sort_values('Value', ascending = False)
Universal_Frame = Universal_Frame.reset_index()
Universal_Frame = Universal_Frame.drop(columns = ['index'])
```

```
sns.barplot(Universal_Frame.Value, Universal_Frame.Genre, palette = 'cubehelix')
plt.xlabel('Average of Gross Percentage (%)',fontsize = 12)
plt.ylabel(' Genre',fontsize = 12)
plt.title('Universal Studios',fontsize = 15)
```



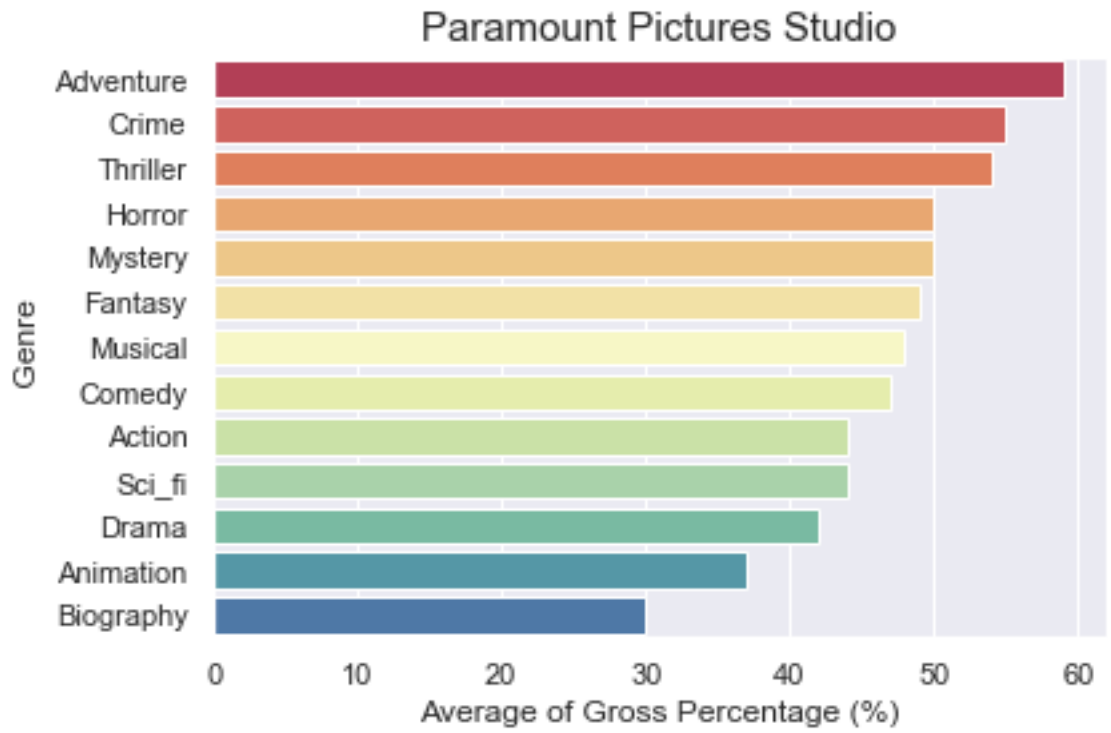

```
Sony_Frame = pd.DataFrame()
Sony_Frame['Genre'] = ['Action', 'Adventure', 'Animation',
      'Comedy', 'Crime','Drama', 'Thriller', 'Biography',
      'Documentry', 'Sci_fi']
Sony_Frame['Value'] = [39, 42, 36, 52, 44, 49, 28, 48,28,24]
Sony_Frame = Sony_Frame.sort_values('Value', ascending = False)
Sony_Frame = Sony_Frame.reset_index()
Sony_Frame = Sony_Frame.drop(columns = ['index'])

sns.barplot(Sony_Frame.Value, Sony_Frame.Genre, palette = 'Spectral')
```

```
plt.xlabel('Average of Gross Percentage (%)',fontsize = 12)
plt.ylabel(' Genre',fontsize = 12)
plt.title('Sony Studios',fontsize = 15)
```

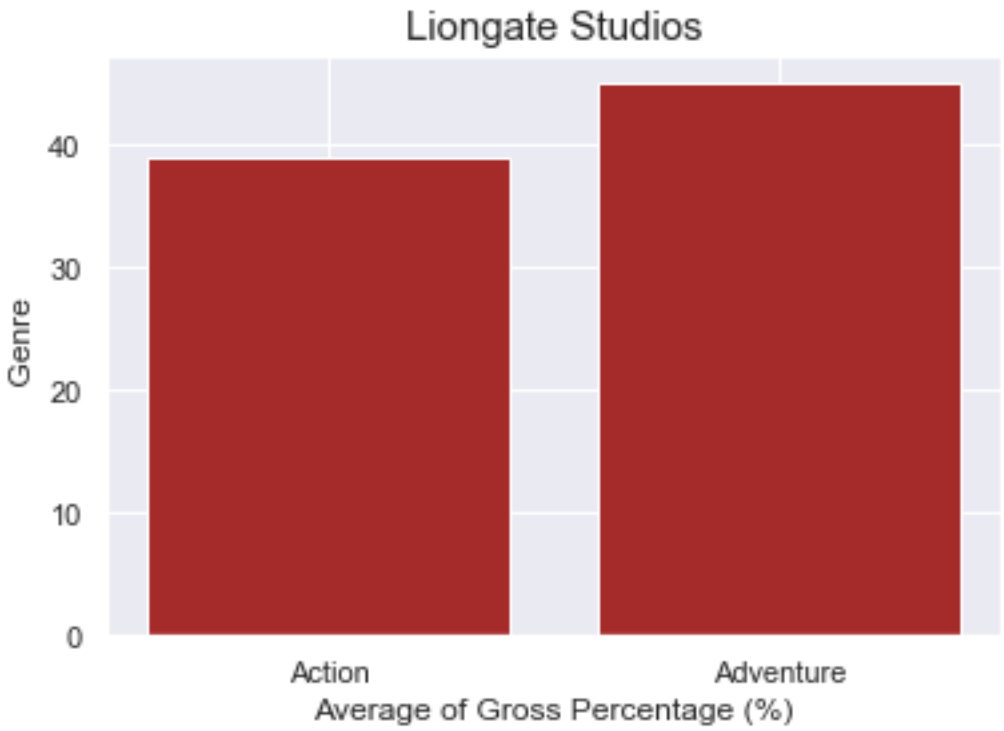



```
Paramount_Frame = pd.DataFrame()
Paramount_Frame['Genre'] = ['Action', 'Adventure', 'Animation',
      'Comedy', 'Crime','Drama', 'Thriller', 'Biography',
      'Sci_fi', 'Fantasy', 'Horror', 'Musical', 'Mystery']
Paramount_Frame['Value'] = [44, 59, 37, 47, 55, 42, 54, 30, 44, 49, 50, 48,50]
Paramount_Frame = Paramount_Frame.sort_values('Value', ascending = False)
Paramount_Frame = Paramount_Frame.reset_index()
Paramount_Frame = Paramount_Frame.drop(columns = ['index'])

sns.barplot(Paramount_Frame.Value, Paramount_Frame.Genre, palette = 'Spectral')
plt.xlabel('Average of Gross Percentage (%)',fontsize = 12)
```

```
plt.ylabel(' Genre',fontsize = 12)
plt.title('Paramount Pictures Studio',fontsize = 15)
```




```
Liongate_Frame = pd.DataFrame()
Liongate_Frame['Genre'] = ['Action', 'Adventure']
Liongate_Frame['Value'] = [39, 45]
Liongate_Frame = Liongate_Frame.sort_values('Value', ascending = True)
Liongate_Frame = Liongate_Frame.reset_index()
Liongate_Frame = Liongate_Frame.drop(columns = ['index'])

plt.bar(Liongate_Frame.Genre, Liongate_Frame.Value, color = 'Brown')
plt.xlabel('Average of Gross Percentage (%)',fontsize = 12)
plt.ylabel(' Genre',fontsize = 12)
plt.title('Liongate Studios',fontsize = 15)
```


Liongate Studios
40
30
20
10
0
Genre
Action
Adventure
Average of Gross Percentage (%)