In [1]:
```
import pandas as pd
```

In [3]:
```
import os
os.getcwd()
```

Out[3]:
```
'C:\\Users\\AKSHAY\\Python Practice Projects'
```

In [5]:
```
movies = pd.read_csv(r"C:\Users\AKSHAY\OneDrive\Desktop\Code\Projects\Project Co
```

In [7]:
```
movies
```

Out[7]:

	Film	Genre	Rotten Tomatoes Ratings %	Audience Ratings %	Budget (million $)	Year of release
0	(500) Days of Summer	Comedy	87	81	8	2009
1	10,000 B.C.	Adventure	9	44	105	2008
2	12 Rounds	Action	30	52	20	2009
3	127 Hours	Adventure	93	84	18	2010
4	17 Again	Comedy	55	70	20	2009
...	...	...	...	...	...	...
554	Your Highness	Comedy	26	36	50	2011
555	Youth in Revolt	Comedy	68	52	18	2009
556	Zodiac	Thriller	89	73	65	2007
557	Zombieland	Action	90	87	24	2009
558	Zookeeper	Comedy	14	42	80	2011

559 rows × 6 columns

In [9]:
```
# WE ARE CHECKING THE LENGTH OF THE DATASET
```

In [11]:
```
len(movies)
```

Out[11]:
```
559
```

In [13]:
```
# WE ARE PRINTING THE FIRST 5 VALUES OF THE DATASET
```

In [15]:
```
movies.head()
```

Out[15]:

	Film	Genre	Rotten Tomatoes Ratings %	Audience Ratings %	Budget (million $)	Year of release
0	(500) Days of Summer	Comedy	87	81	8	2009
1	10,000 B.C.	Adventure	9	44	105	2008
2	12 Rounds	Action	30	52	20	2009
3	127 Hours	Adventure	93	84	18	2010
4	17 Again	Comedy	55	70	20	2009

In [17]:
```
# WE ARE PRINTING THE LAST 5 VALUES OF THE DATASET
```

In [19]:
```
movies.tail()
```

Out[19]:

	Film	Genre	Rotten Tomatoes Ratings %	Audience Ratings %	Budget (million $)	Year of release
554	Your Highness	Comedy	26	36	50	2011
555	Youth in Revolt	Comedy	68	52	18	2009
556	Zodiac	Thriller	89	73	65	2007
557	Zombieland	Action	90	87	24	2009
558	Zookeeper	Comedy	14	42	80	2011

In [21]:
```
# WE ARE PRINTING THE COLUMN NAMES OF THE DATASET
```

In [23]:
```
movies.columns
```

Out[23]:
```
Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
       'Budget (million $)', 'Year of release'],
      dtype='object')
```

In [25]:
```
# AS THE COLUMN NAMES CONSIST OF UNWANTED CHARACTERS, WE WILL CLEAN THEM
```

In [27]:
```
movies.columns = ['Film', 'Genre', 'CriticRating', 'AudienceRating', 'BudgetMill
```

In [29]:
```
movies
```

Out[29]:

	Film	Genre	CriticRating	AudienceRating	BudgetMillions	Year
0	(500) Days of Summer	Comedy	87	81	8	2009
1	10,000 B.C.	Adventure	9	44	105	2008
2	12 Rounds	Action	30	52	20	2009
3	127 Hours	Adventure	93	84	18	2010
4	17 Again	Comedy	55	70	20	2009
...	...	...	...	...	...	...
554	Your Highness	Comedy	26	36	50	2011
555	Youth in Revolt	Comedy	68	52	18	2009
556	Zodiac	Thriller	89	73	65	2007
557	Zombieland	Action	90	87	24	2009
558	Zookeeper	Comedy	14	42	80	2011

559 rows × 6 columns

In [31]:
```
# WE ARE PRINTING THE INFORMATION OF THE DATASET, (MISSING VALUES, DATA TYPE) ET
```

In [33]:
```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 559 entries, 0 to 558
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Film            559 non-null    object
 1   Genre           559 non-null    object
 2   CriticRating    559 non-null    int64 
 3   AudienceRating  559 non-null    int64 
 4   BudgetMillions  559 non-null    int64 
 5   Year            559 non-null    int64 
dtypes: int64(4), object(2)
memory usage: 26.3+ KB
```

In [35]:
```
# WE ARE CHECKING THE STATISTICAL DESCRIPTION OF THE DATASET
```

In [37]:
```
movies.describe()
```

Out[37]:

	CriticRating	AudienceRating	BudgetMillions	Year
count	559.000000	559.000000	559.000000	559.000000
mean	47.309481	58.744186	50.236136	2009.152057
std	26.413091	16.826887	48.731817	1.362632
min	0.000000	0.000000	0.000000	2007.000000
25%	25.000000	47.000000	20.000000	2008.000000
50%	46.000000	58.000000	35.000000	2009.000000
75%	70.000000	72.000000	65.000000	2010.000000
max	97.000000	96.000000	300.000000	2011.000000

In [39]:

```
# WE ARE CHANGING THE DTYPES OF THE COLUMNS
```

In [41]:

```
movies['Film'] = movies['Film'].astype('category')
movies['Genre'] = movies['Genre'].astype('category')
movies['Year'] = movies['Year'].astype('category')
```

In [43]:

```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 559 entries, 0 to 558
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype   
---  ------          --------------  -----   
 0   Film            559 non-null    category
 1   Genre           559 non-null    category
 2   CriticRating    559 non-null    int64   
 3   AudienceRating  559 non-null    int64   
 4   BudgetMillions  559 non-null    int64   
 5   Year            559 non-null    category
dtypes: category(3), int64(3)
memory usage: 36.5 KB
```

In [45]:

```
# WE ARE PRINTING THE CATEGORIES INSIDE THE 'GENRE' COLUMN
```

In [51]:

```
movies['Genre']
```

Out[51]:

```
0         Comedy
1      Adventure
2         Action
3      Adventure
4         Comedy
         ...    
554       Comedy
555       Comedy
556     Thriller
557       Action
558       Comedy
Name: Genre, Length: 559, dtype: category
Categories (7, object): ['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'R
omance', 'Thriller']
```

In [49]:

```
movies['Genre'].cat.categories
```

Out[49]:
```
Index(['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'Romance',
       'Thriller'],
      dtype='object')
```

In [55]:
```
# WE ARE CHECKING THE STATISTICAL DESCRIPTION OF THE DATASET
```

In [57]:
```
# HERE THE 'YEAR' COLUMN DOES NOT PRINT BECAUSE ITS DTYPE IS CATEGORY, BEFORE IT
```

In [53]:
```
movies.describe()
```

Out[53]:

	CriticRating	AudienceRating	BudgetMillions
count	559.000000	559.000000	559.000000
mean	47.309481	58.744186	50.236136
std	26.413091	16.826887	48.731817
min	0.000000	0.000000	0.000000
25%	25.000000	47.000000	20.000000
50%	46.000000	58.000000	35.000000
75%	70.000000	72.000000	65.000000
max	97.000000	96.000000	300.000000

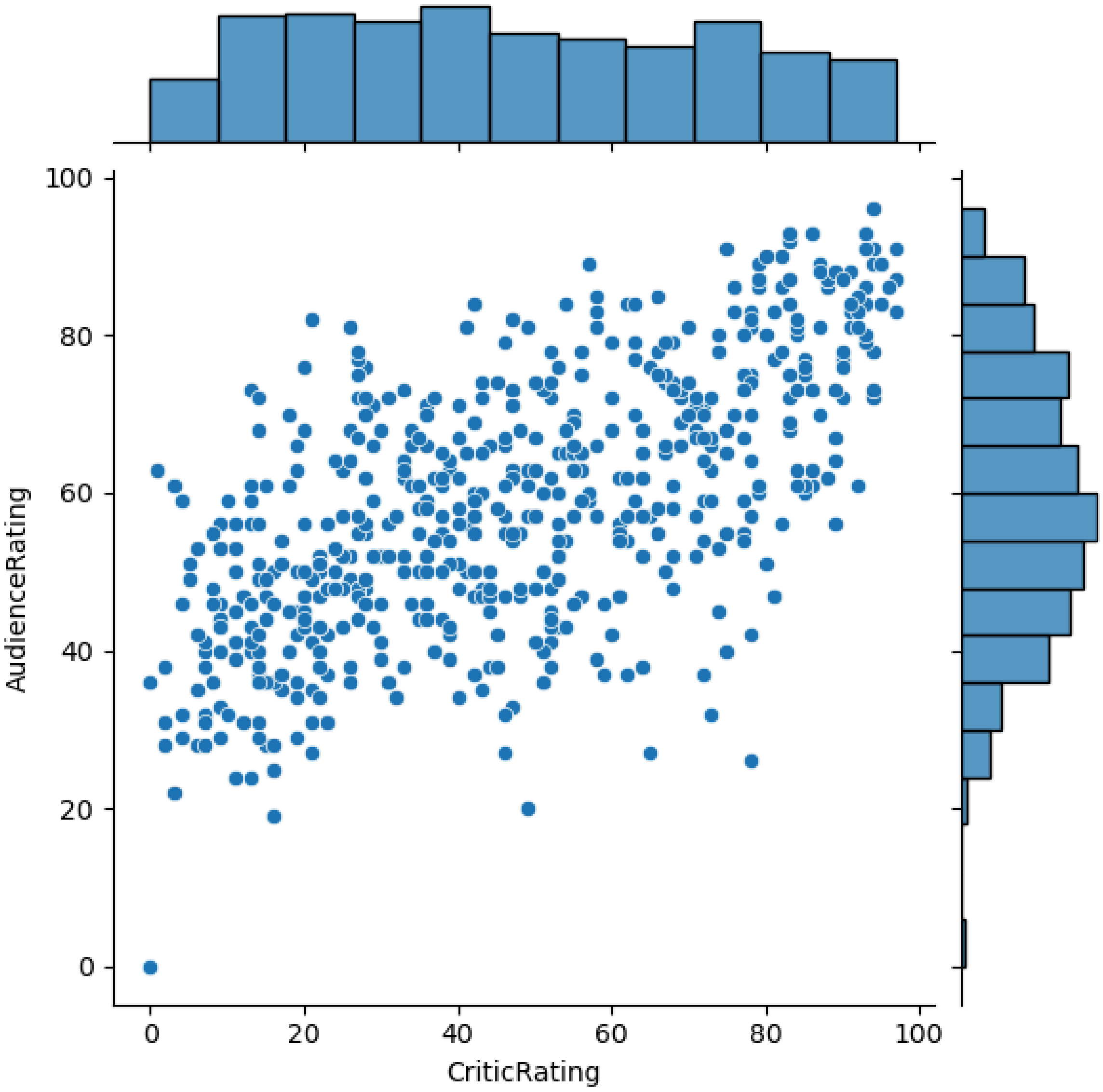
VISUALIZATIONS

In [62]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [64]:
```
import warnings
warnings.filterwarnings('ignore')
```

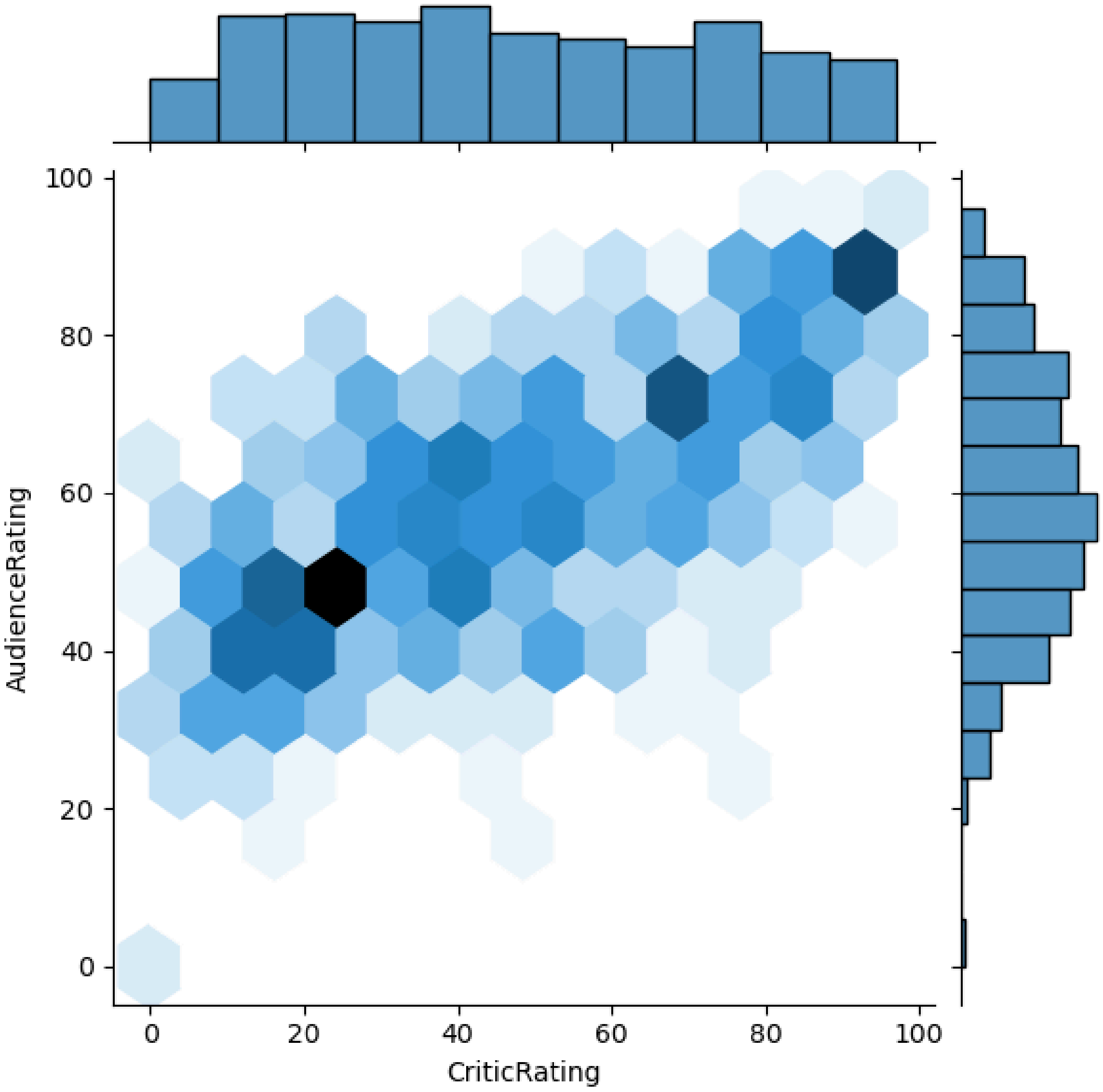
In [66]:
```
# jointplot shows the scatter plot of two variables
# x='CriticRating': Maps the CriticRating column to the x-axis.
# y='AudienceRating': Maps the AudienceRating column to the y-axis.
```

In [72]:
```
v1 = sns.jointplot(data = movies, x = 'CriticRating', y = 'AudienceRating')
plt.show()
```



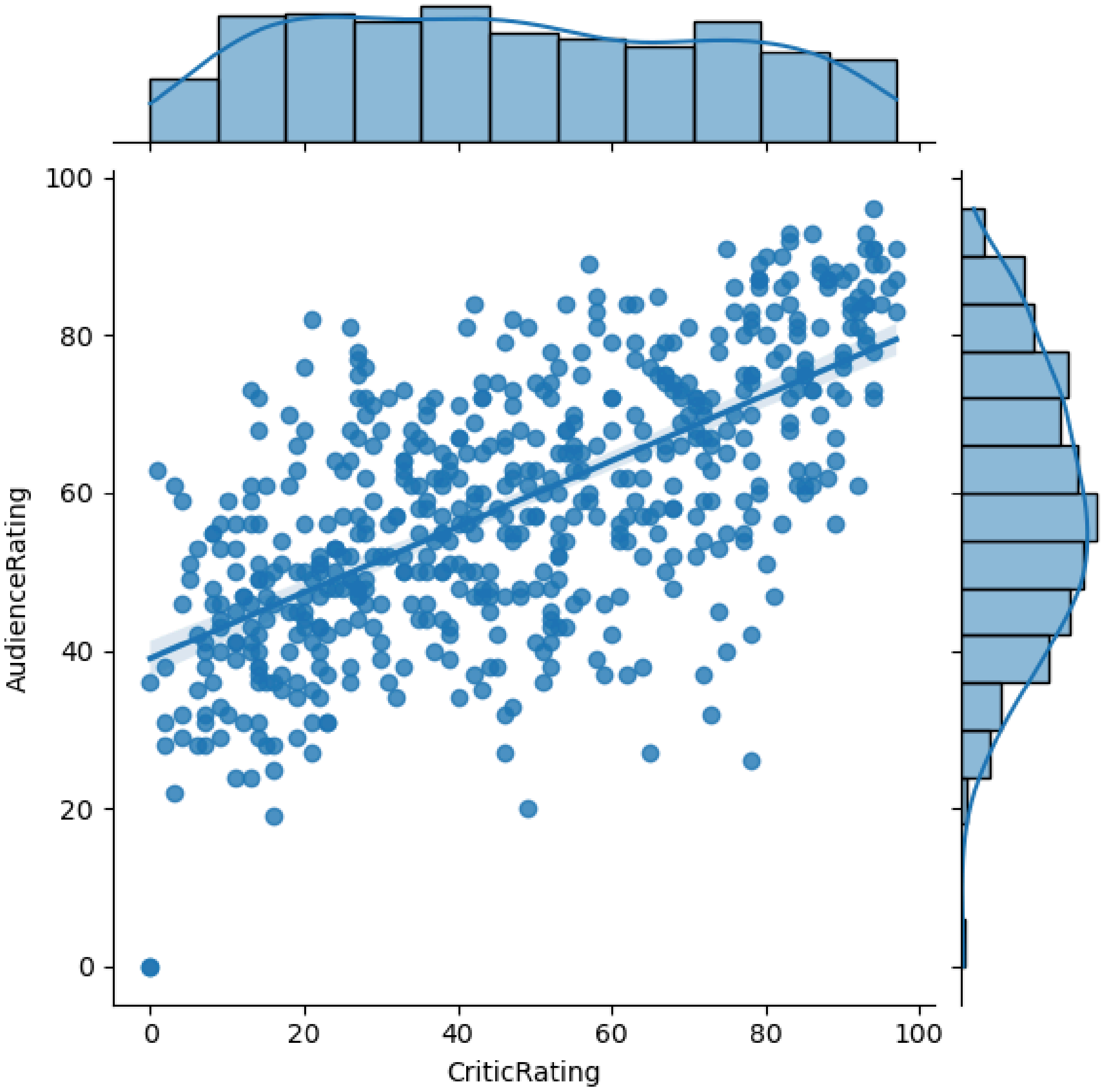
```
In [76]: # jointplot shows the scatter plot of two variables
         # x='CriticRating': Maps the CriticRating column to the x-axis.
         # y='AudienceRating': Maps the AudienceRating column to the y-axis.
         # kind='hex' specifies that a hexbin plot should be used in the central area
```

```
In [74]: v2 = sns.jointplot(data = movies, x = 'CriticRating', y = 'AudienceRating', kind
         plt.show()
```



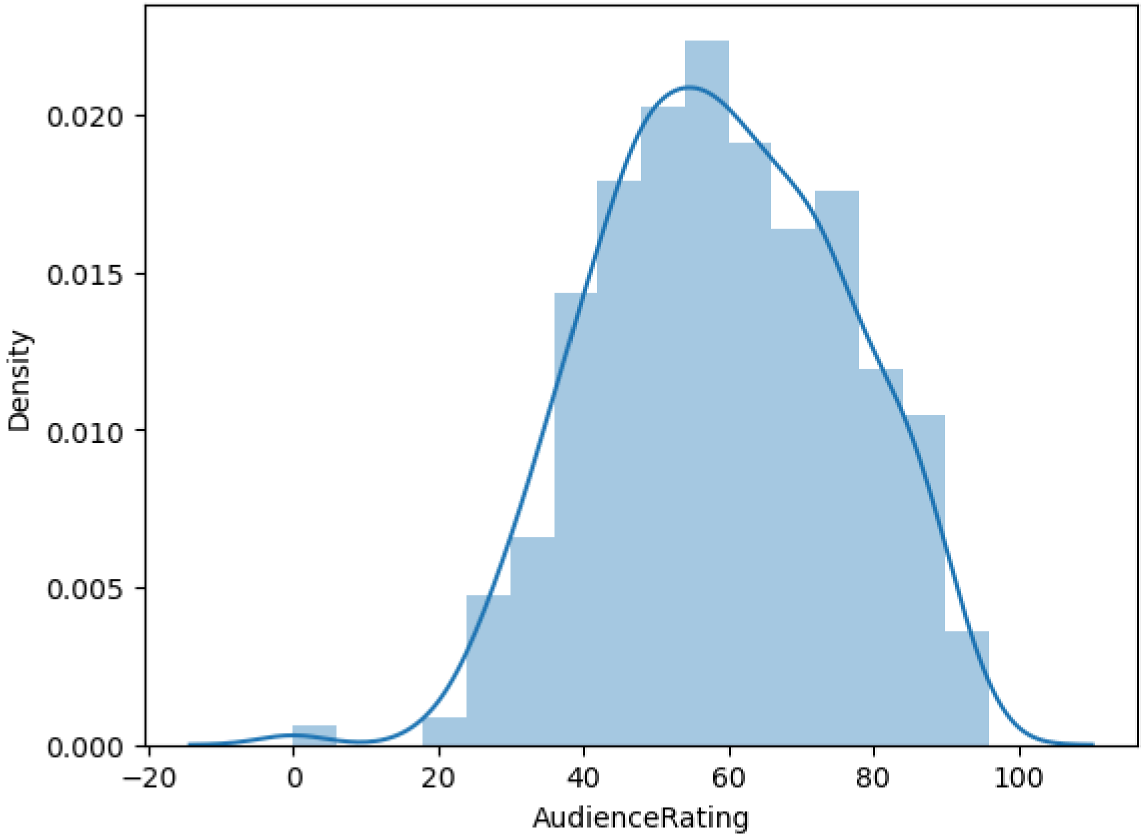
In [78]:
```
# jointplot shows the scatter plot of two variables
# x='CriticRating': Maps the CriticRating column to the x-axis.
# y='AudienceRating': Maps the AudienceRating column to the y-axis.
# kind='reg' specifies that a reg plot should be used in the central area
```

In [80]:
```
v3 = sns.jointplot(data = movies, x = 'CriticRating', y = 'AudienceRating', kind
plt.show()
```



```
In [82]: # distplot shows the distribution plot : combines hist() and kde()
```

```
In [84]: v5 = sns.distplot(movies.AudienceRating)
         plt.show()
```



In [86]:

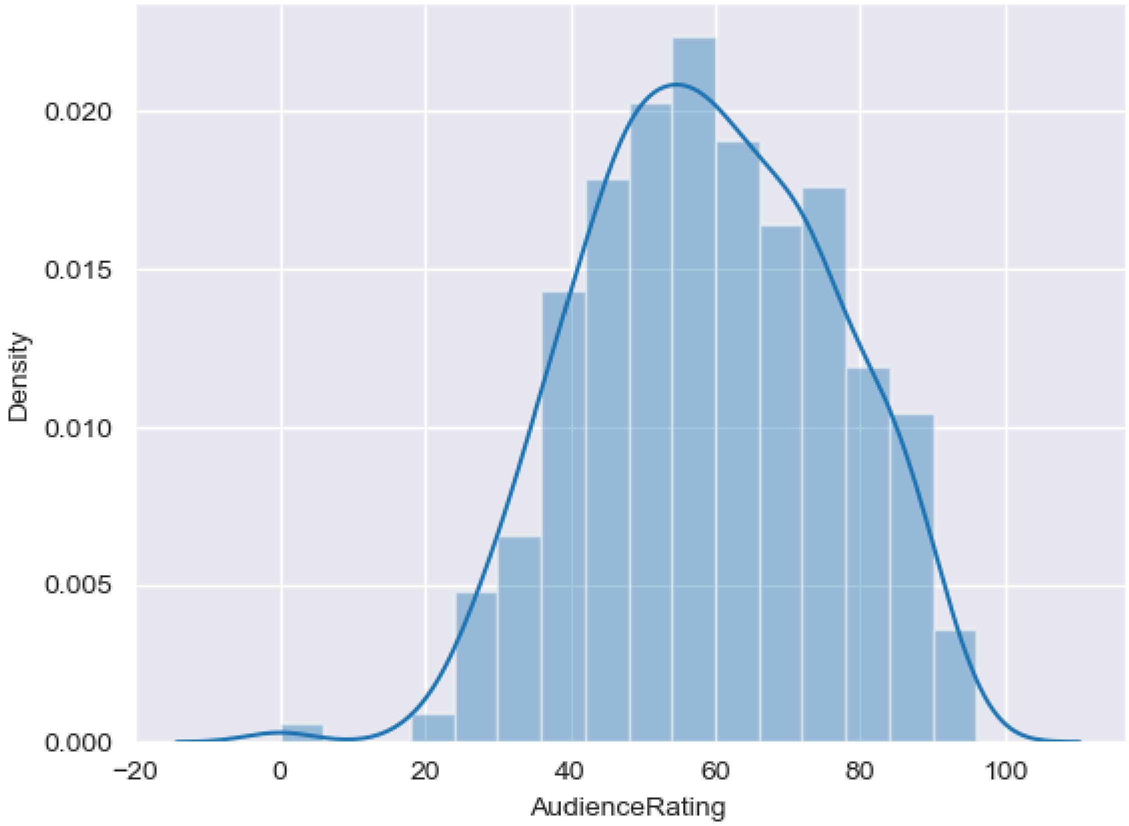
```
# WE ARE CHANGING THE BACKGROUND STYLE OF THE PLOT
```

In [100…

```
sns.set_style('darkgrid')
```

In [102…

```
v4_a = sns.distplot(movies.AudienceRating)
plt.show()
```



In [108…

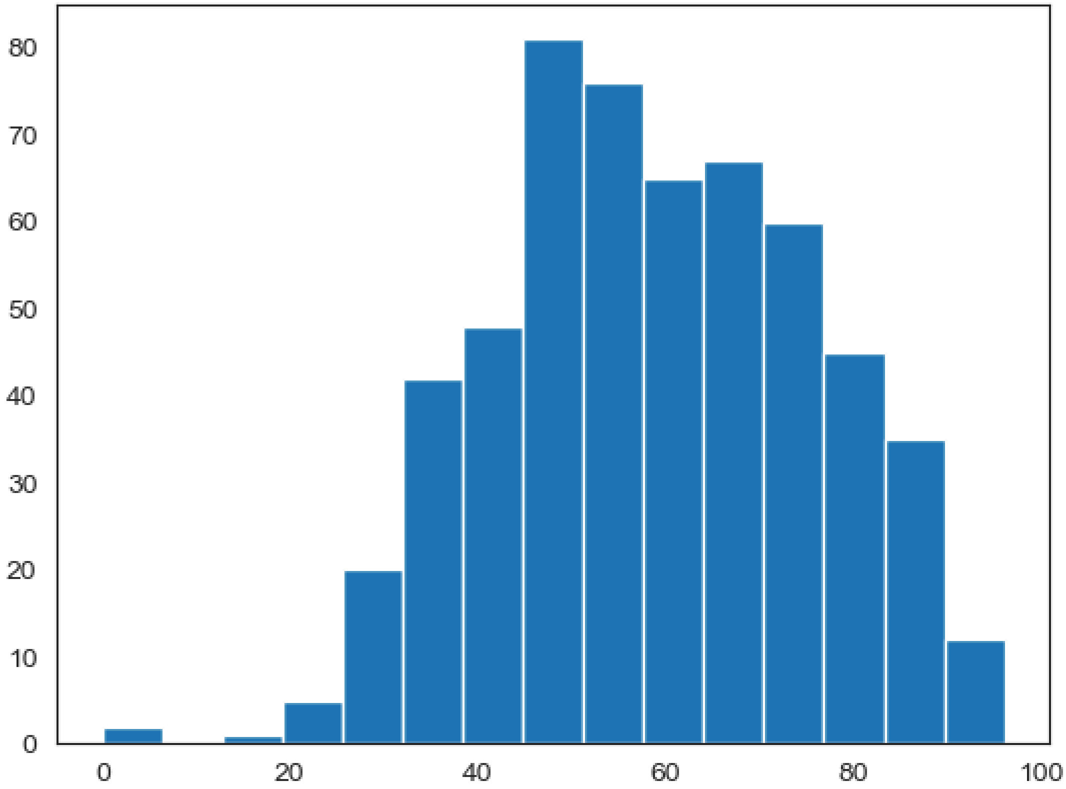
```
sns.set_style('white')
```

In [110…

```
# histplot(), histogram of the column
```

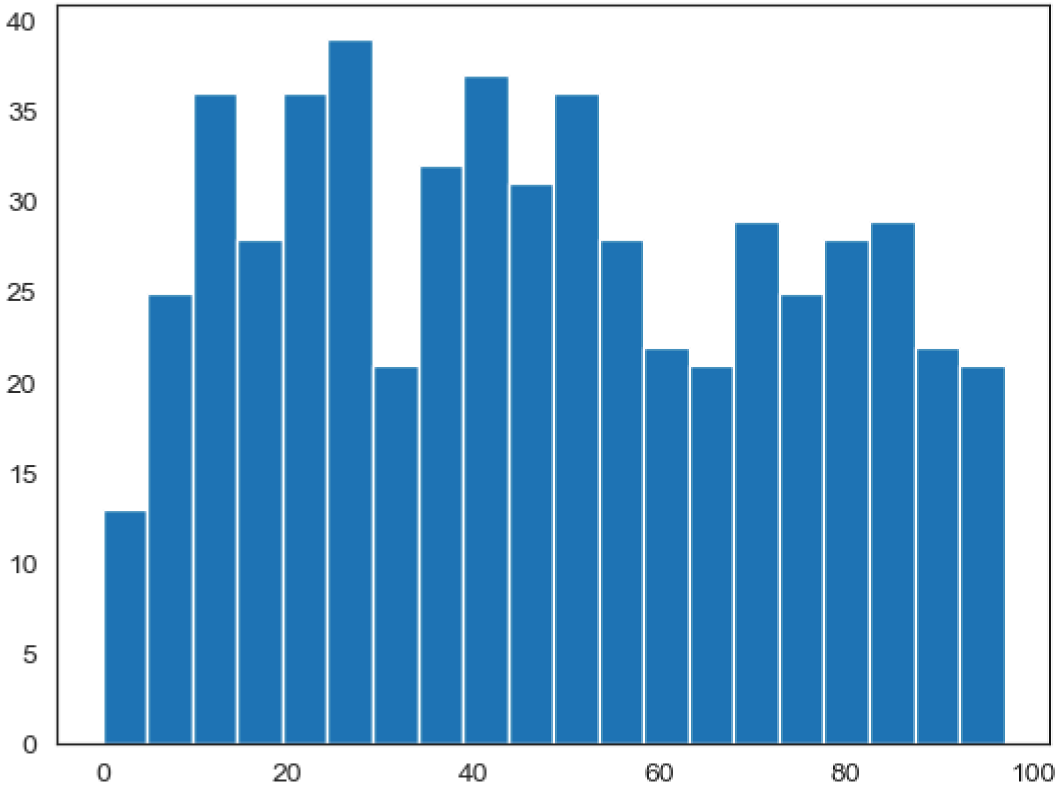
In [112…

```
v5 = plt.hist(movies.AudienceRating, bins = 15)
plt.show()
```



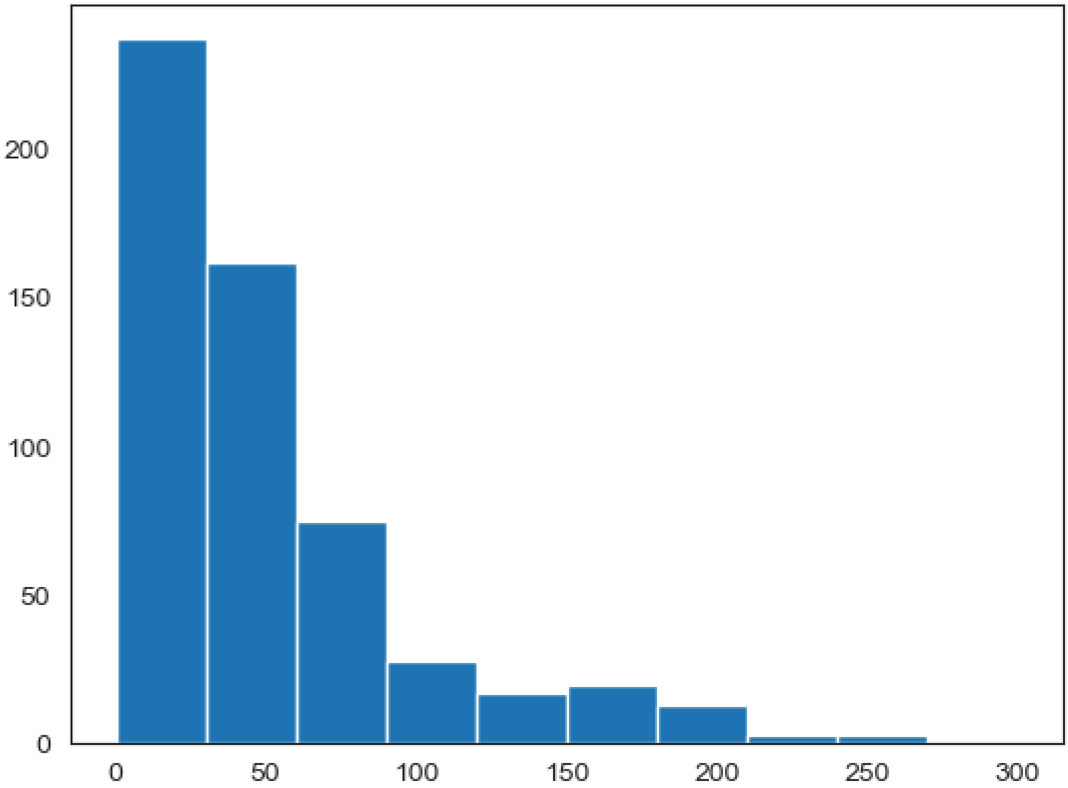
In [114…

```
v5_a = plt.hist(movies.CriticRating, bins = 20)
plt.show() # this is for a different column
```



In [116…

```
v5_v = plt.hist(movies.BudgetMillions)
plt.show() # this for a different column
```

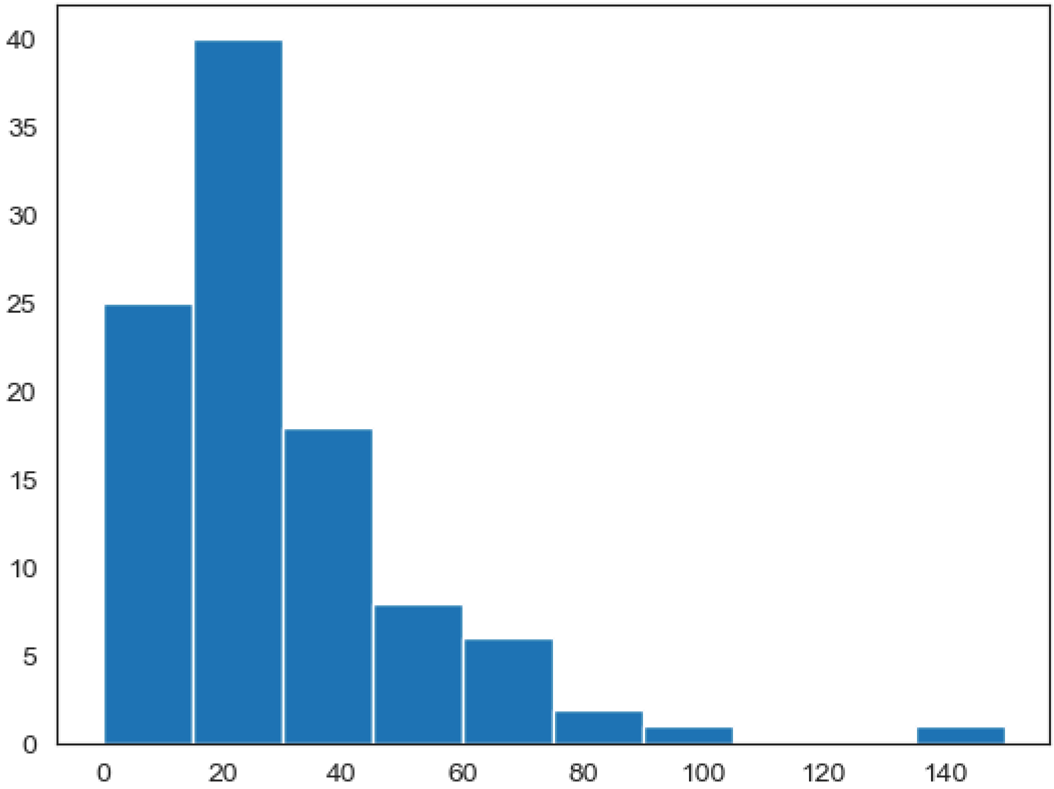


In [118…

```
# WE ARE PRINTING THE HIST PLOT FOR A PARTICULAR CATEGORY INSIDE 'GENRE' COLUMN
```

In [122…

```
v5_c = plt.hist(movies[movies.Genre == 'Drama'].BudgetMillions)
plt.show()
```

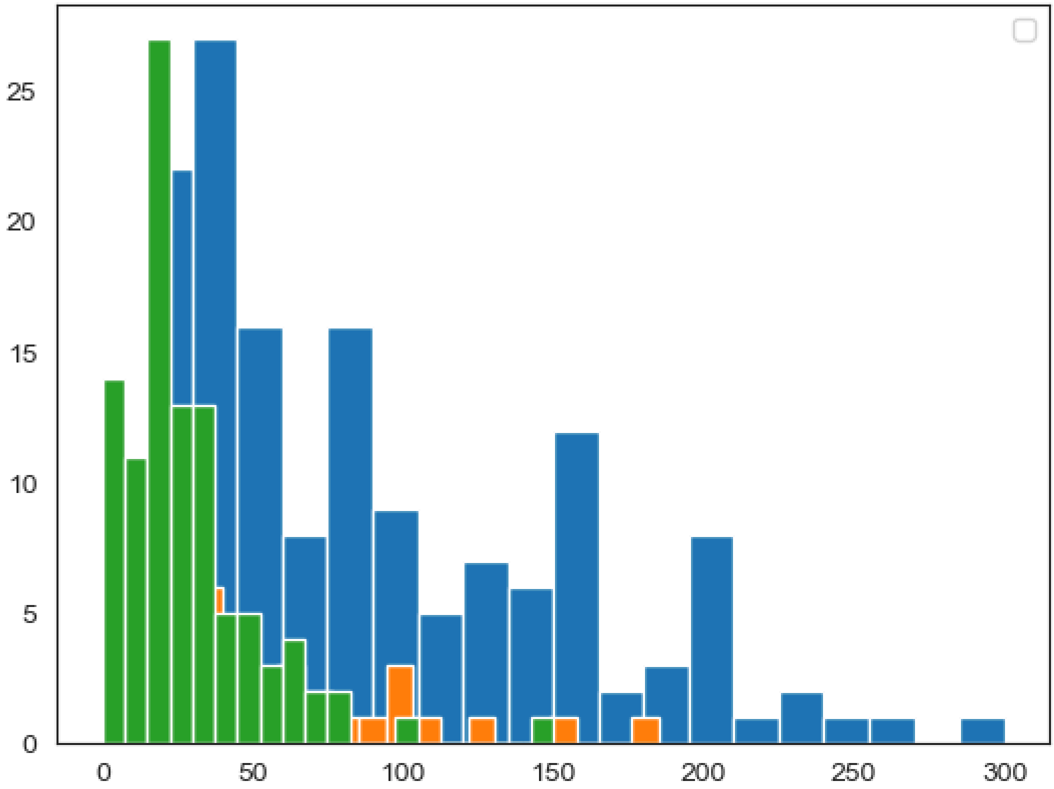


In [124…

```
# WE ARE PRINTING THE HIST PLOT FOR CATEGORIES INSIDE 'GENRE' COLUMN WITH ITS BU
```

In [126…

```
v5_d = plt.hist(movies[movies.Genre == 'Action'].BudgetMillions, bins = 20)
v5_e = plt.hist(movies[movies.Genre == 'Thriller'].BudgetMillions, bins = 20)
v5_f = plt.hist(movies[movies.Genre == 'Drama'].BudgetMillions, bins = 20)
plt.legend()
plt.show()
```

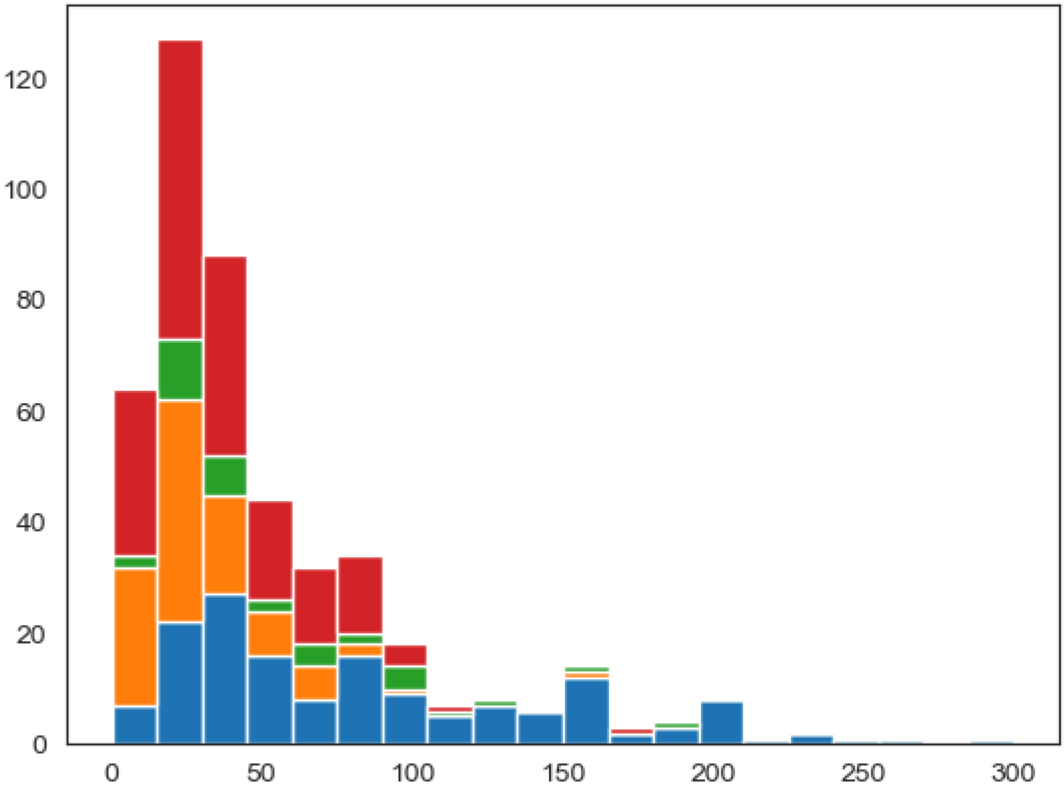


In [128…

```
# WE ARE PRINTING THE HIST PLOT FOR CATEGORIES INSIDE 'GENRE' COLUMN WITH ITS BU
```

In [134…

```
v5_g = plt.hist([movies[movies.Genre == 'Action'].BudgetMillions,\
          movies[movies.Genre == 'Drama'].BudgetMillions, \
          movies[movies.Genre == 'Thriller'].BudgetMillions, \
          movies[movies.Genre == 'Comedy'].BudgetMillions],
         bins = 20, stacked = True)
plt.show()
```

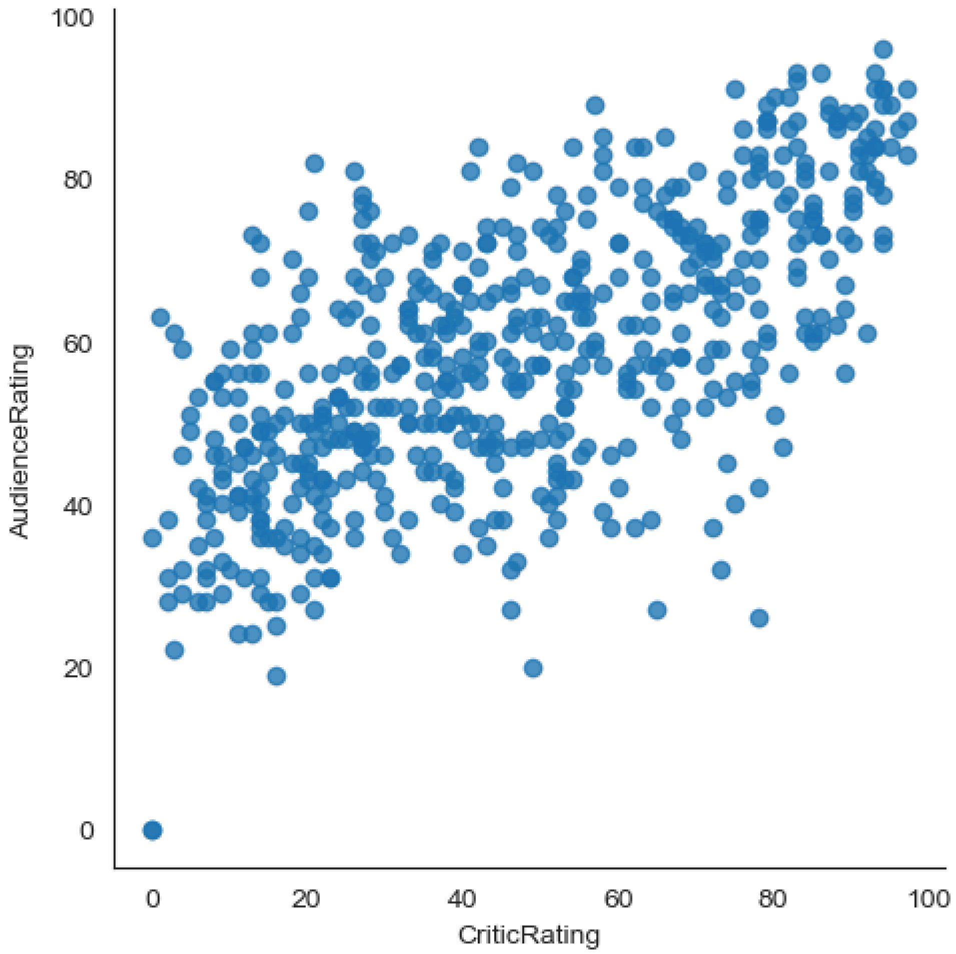


In [136…

```
# lmplot is a regression line plot
```

In [138…

```
v6 = sns.lmplot(data = movies, x = 'CriticRating', y = 'AudienceRating', fit_reg
plt.show()
```

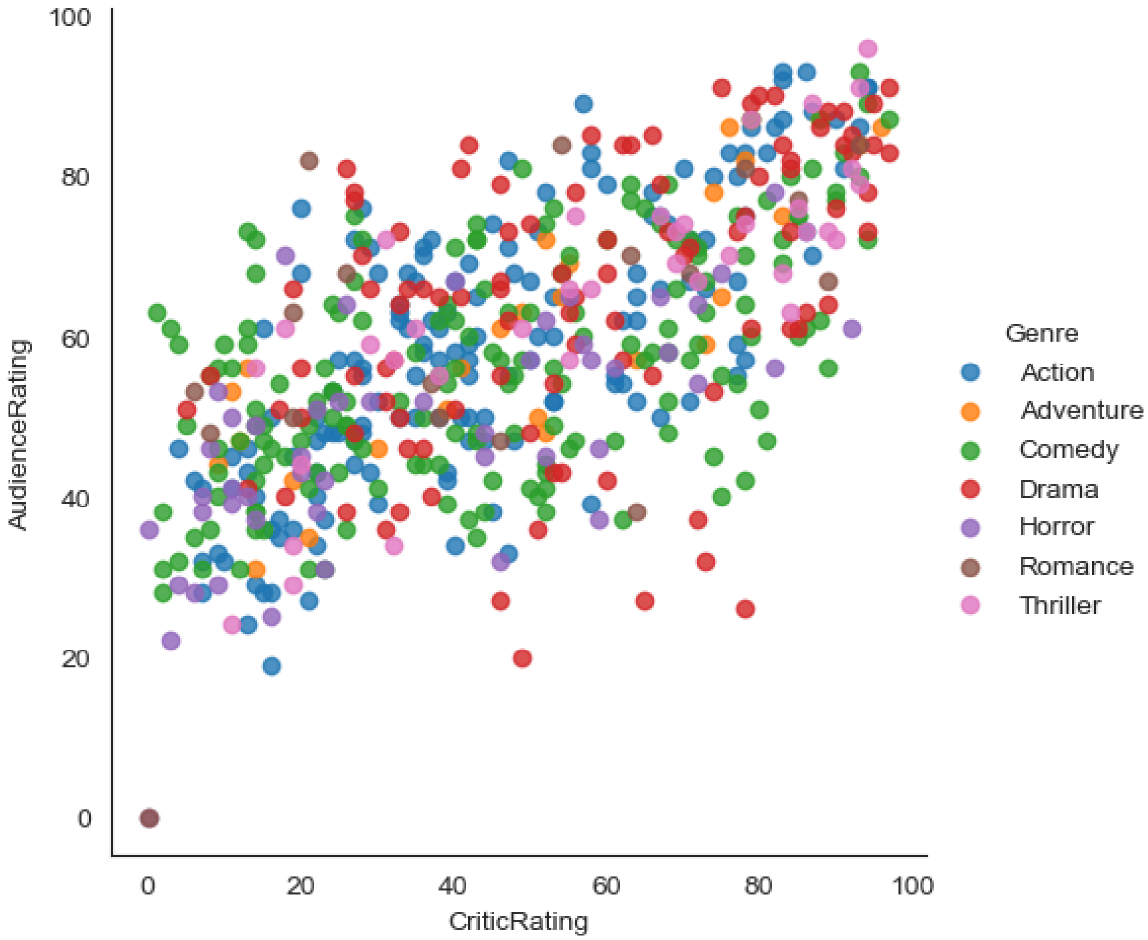


In [140…

```
# lmplot is used to plot the regression line
# hue = 'Genre', colors the points by the Genre column, which is assumed to be c
```

In [142…

```
v6_a = sns.lmplot(data = movies, x = 'CriticRating', y = 'AudienceRating', hue =
plt.show()
```

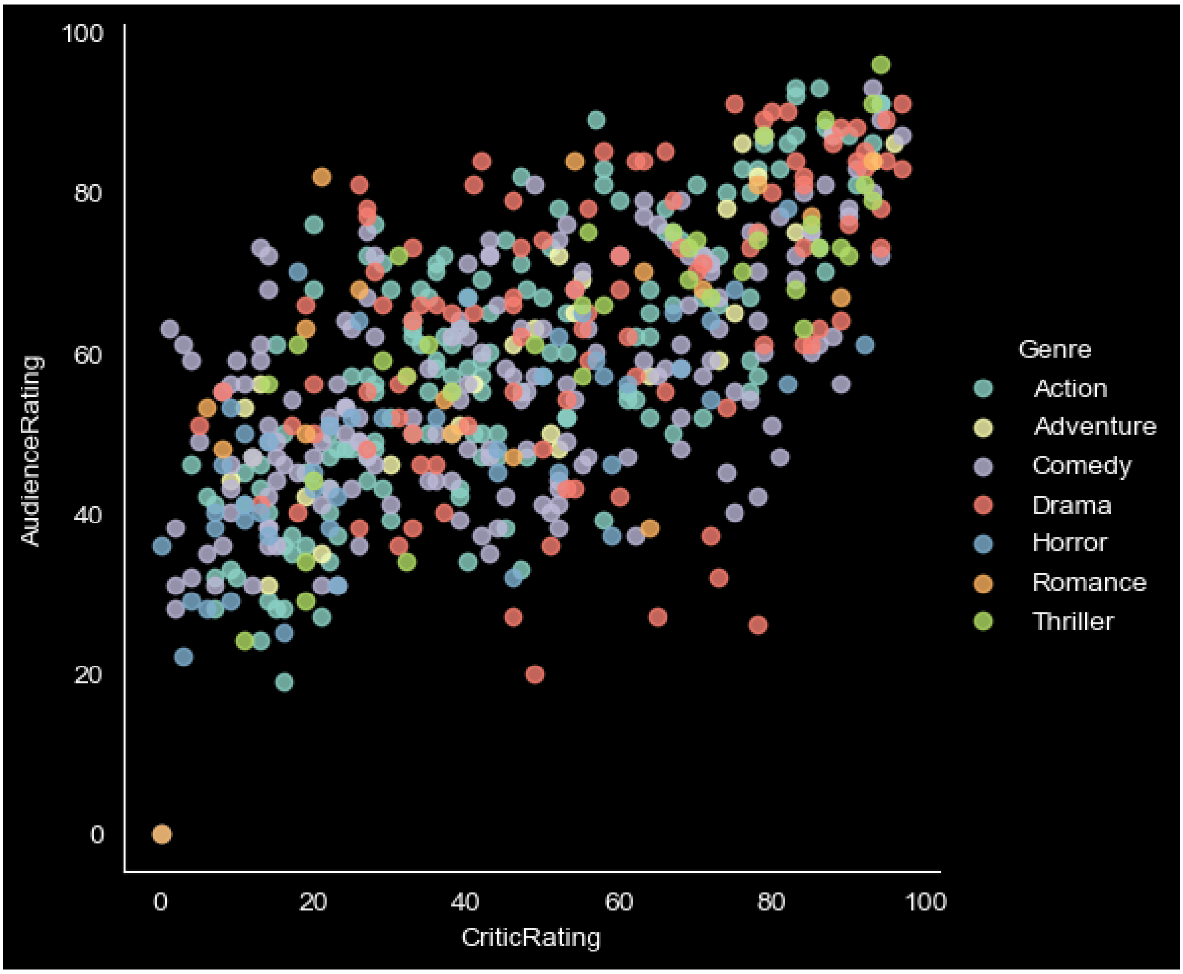


In [144…

```
# WE ARE CHANGING THE BACKGROUND GRID STYLE, THE ABOVE CODE IS THE SAME
```

In [146…

```
plt.style.use('dark_background')
v6_b = sns.lmplot(data = movies, x = 'CriticRating', y = 'AudienceRating', fit_r
plt.show()
```

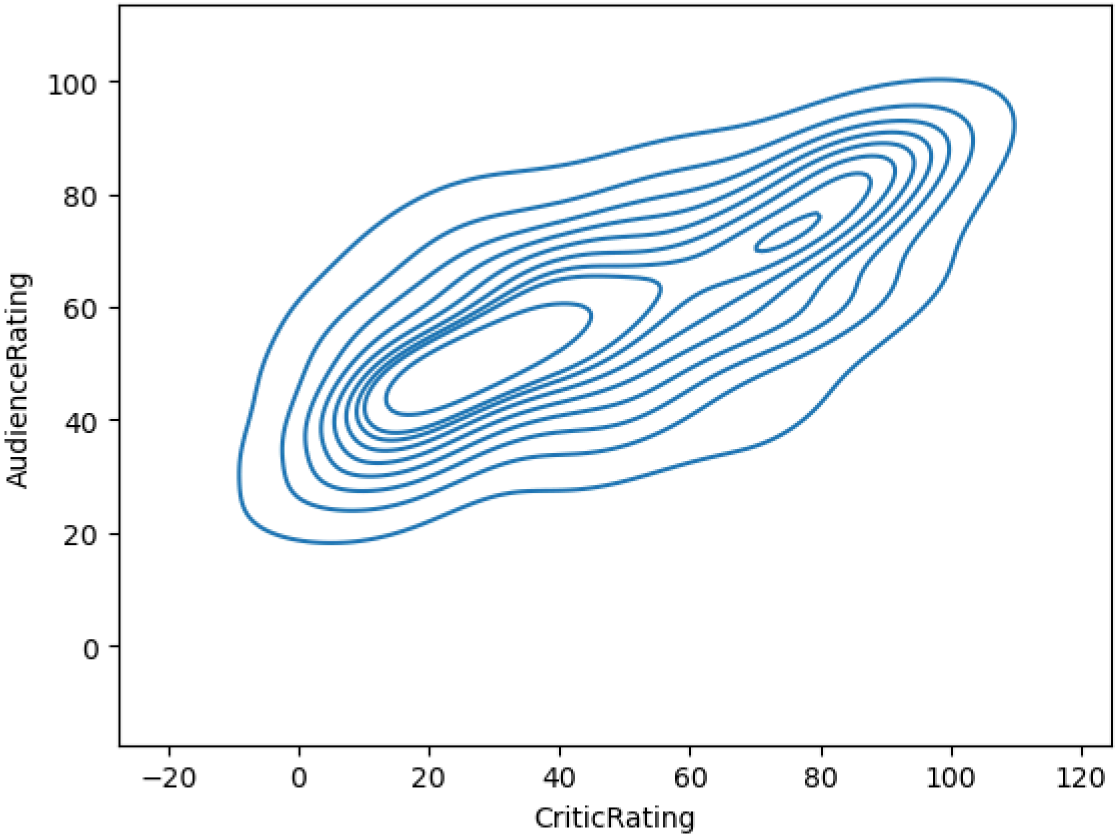


In [148…

```
# kdeplot(), creates a 2D density plot to show the probability density of two va
```

In [152…

```
plt.style.use('default')
v7 = sns.kdeplot(x = 'CriticRating', y = 'AudienceRating', data = movies)
plt.show()
```

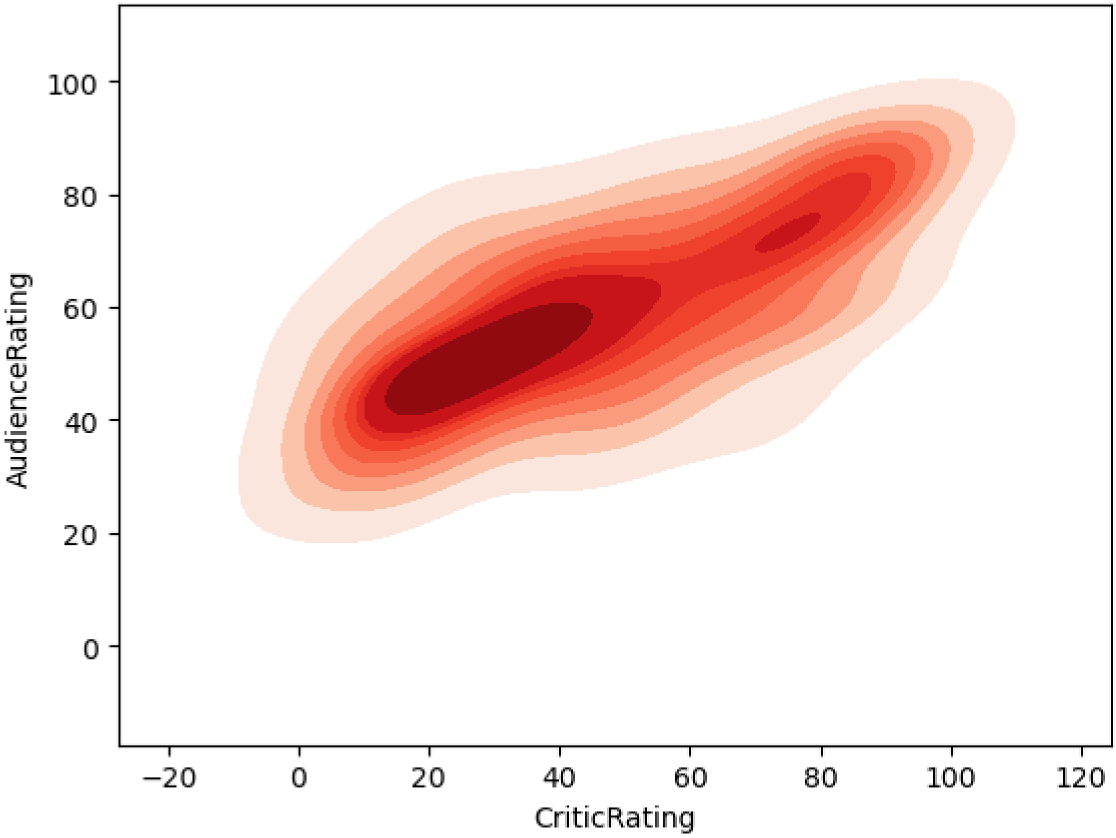


In [154…

```
# WE ARE ADDING SOME STYLING TO THE PLOT
```

In [156…

```
v7_a = sns.kdeplot(data=movies, x='CriticRating', y='AudienceRating', shade=True
plt.show()
```

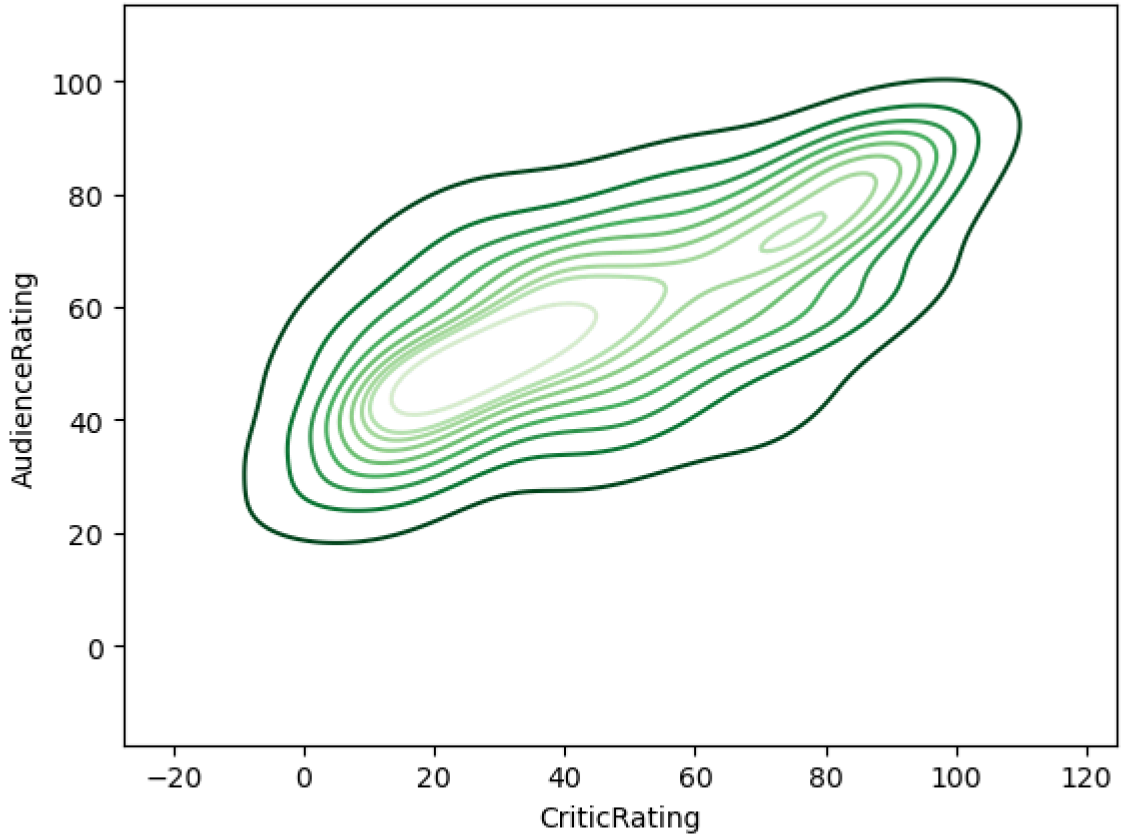


In [158…

```
# WE ARE ADDING SOME STYLING TO THE PLOT
```

In [160…

```
v7_b = sns.kdeplot(data = movies, x= 'CriticRating', y= 'AudienceRating',shade_l
plt.show()
```

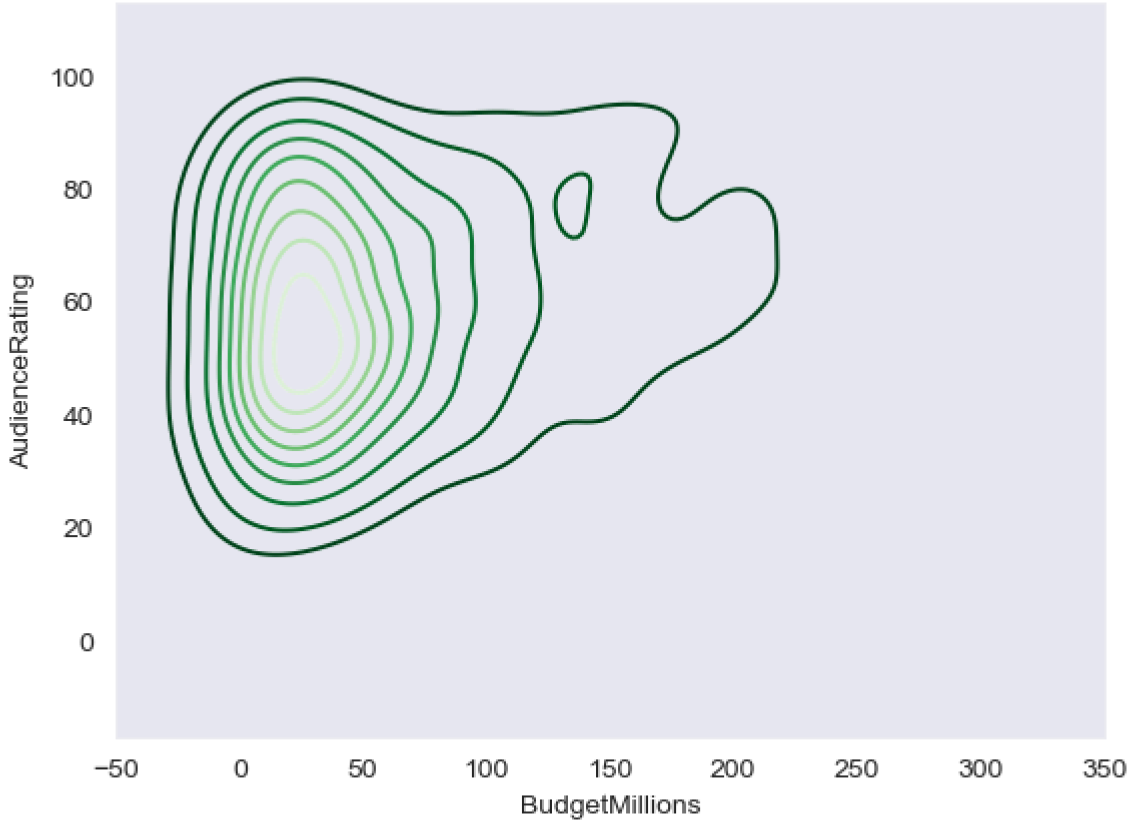


In [162…

```
# WE ARE ADDING SOME STYLING TO THE GRID
```

In [164…

```
sns.set_style('dark')
v7_c = sns.kdeplot(data = movies, x = 'BudgetMillions',y ='AudienceRating',shade
plt.show()
```

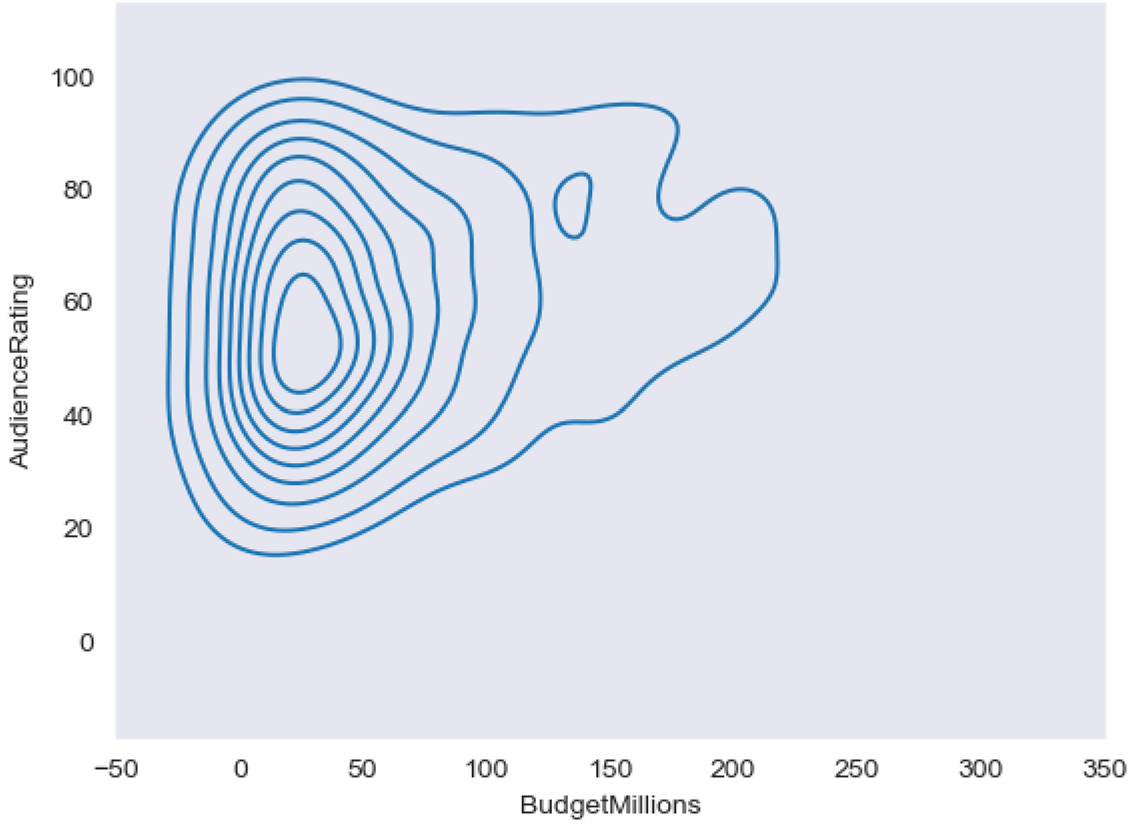


In [168…

```
# WE ARE ADDING NO STYLING TO THE PLOT
```

In [170…

```
sns.set_style('dark')
v7_d = sns.kdeplot(data = movies, x = 'BudgetMillions',y ='AudienceRating')
plt.show()
```

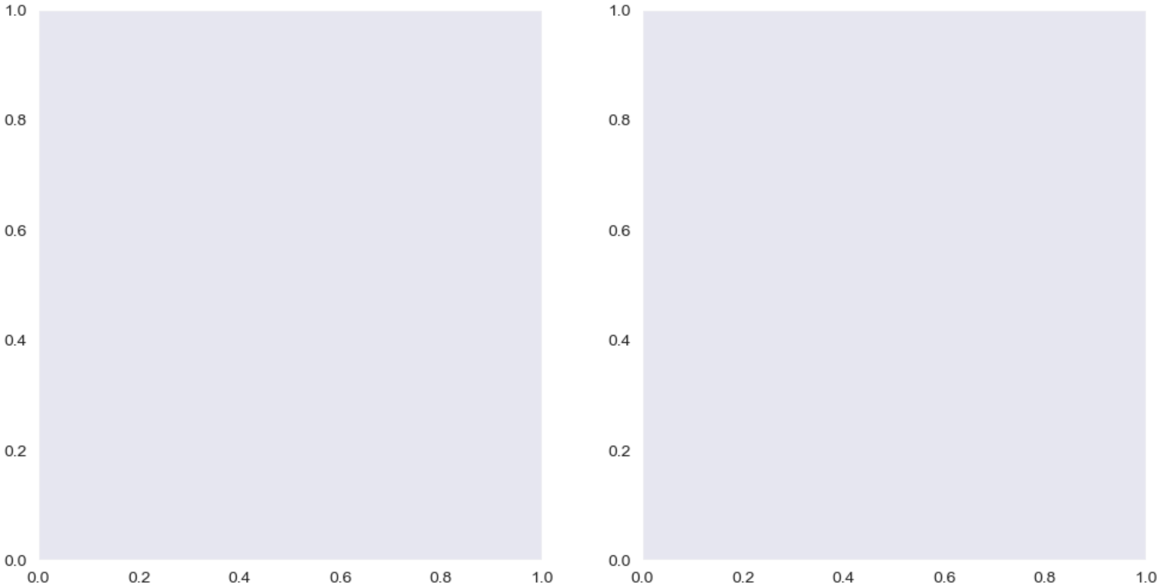


In [172...

```
# subplots(), creates a figure with a grid of subplots.
# 1,2 means 1 row and 2 columns
```

In [174...

```
f, ax = plt.subplots(1,2, figsize = (12,6))
plt.show()
```

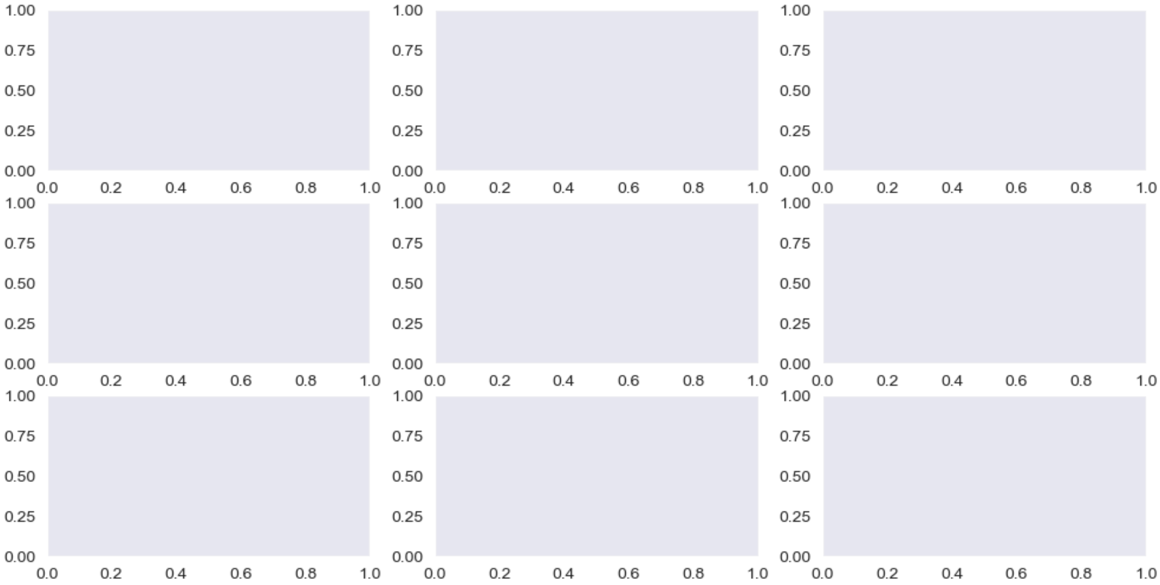


In [176...

```
# subplots(), creates a figure with a grid of subplots.
# 3,3 means 3 rows and 3 columns
```

In [178...

```
f, ax = plt.subplots(3,3, figsize = (12,6))
plt.show()
```

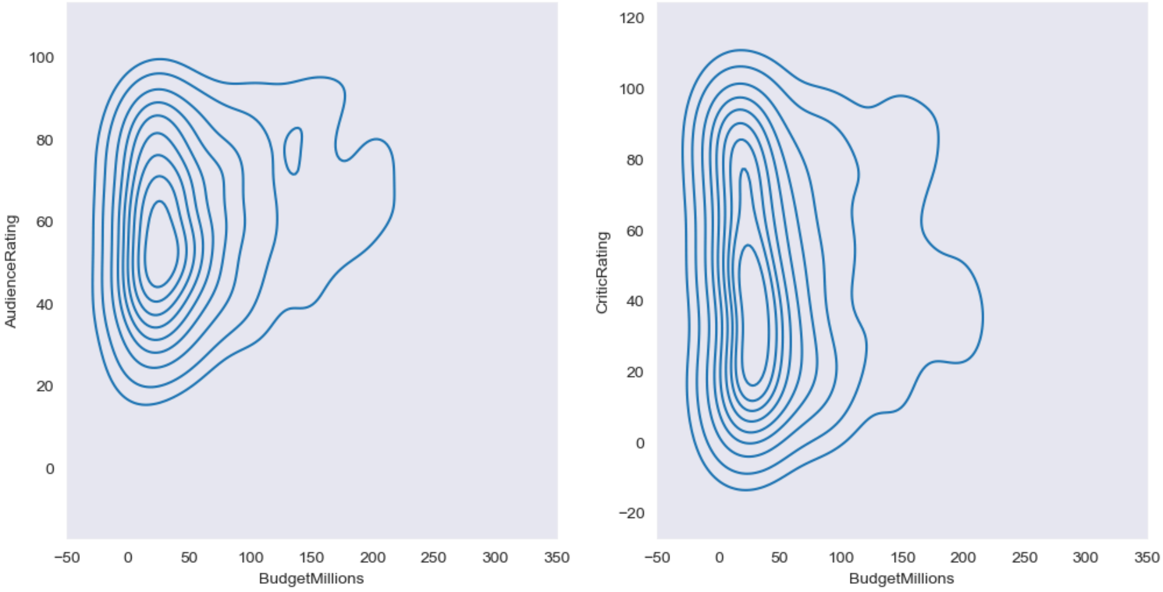


In [180...

```
# This code creates a figure with two subplots side by side, each displaying a K
```

In [182...

```
f, axes = plt.subplots(1,2, figsize = (12, 6))
k1 = sns.kdeplot(data = movies, x = 'BudgetMillions', y = 'AudienceRating', ax =
k2 = sns.kdeplot(data = movies, x = 'BudgetMillions', y = 'CriticRating', ax = a
plt.show()
```

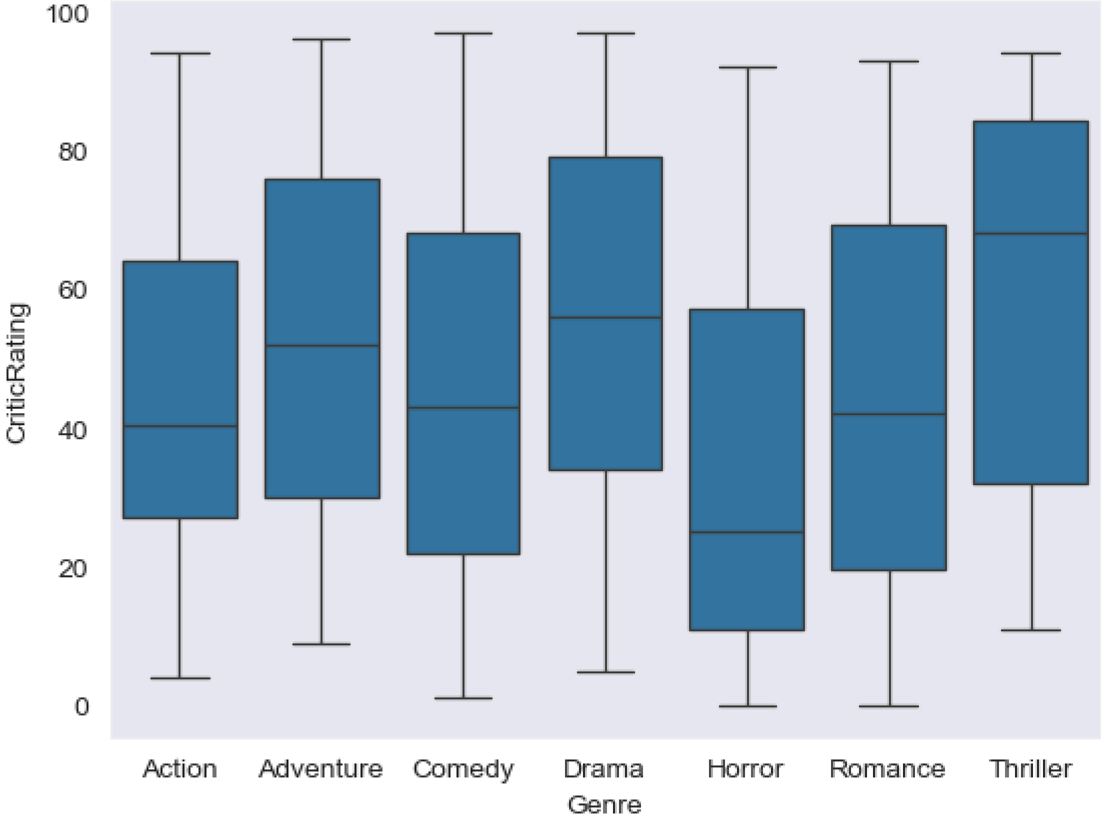


In [184…

```
# boxplot(), gives the numerical ditribution of the columns
# like max, min, average
```

In [186…

```
v8 = sns.boxplot(data = movies, x = 'Genre', y = 'CriticRating')
plt.show()
```

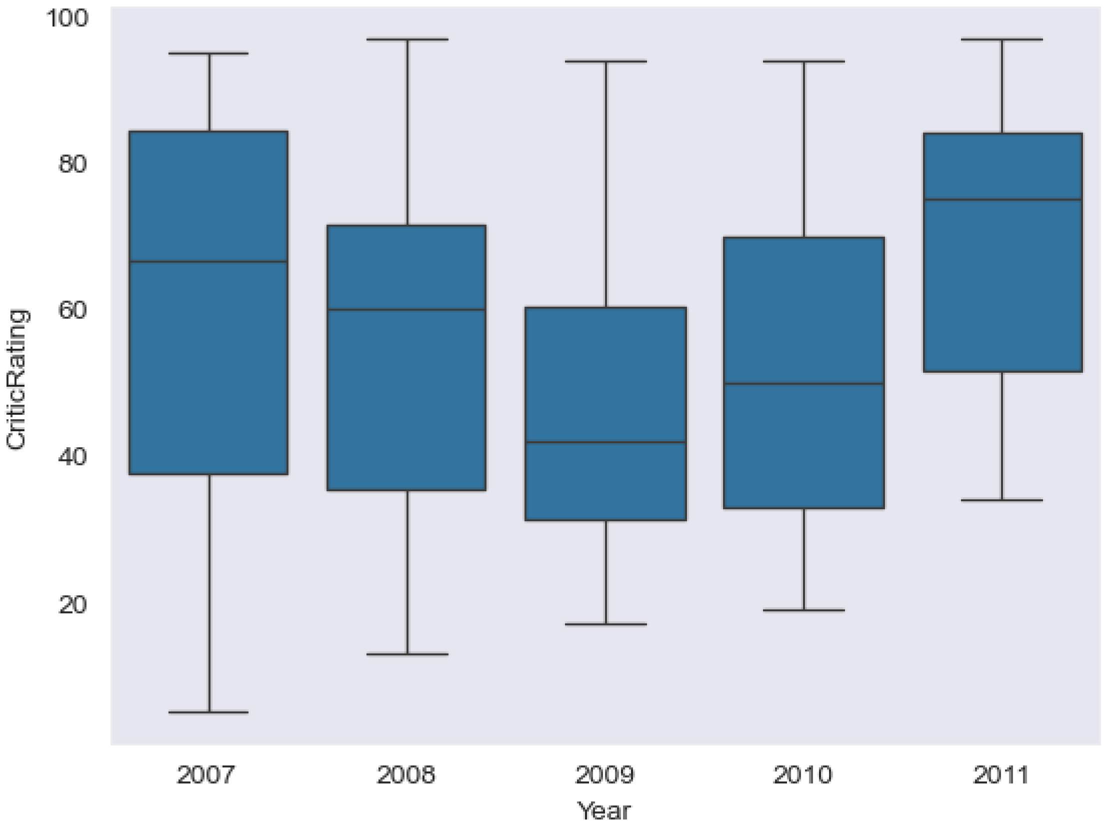


In [194…

```
# this code gives the boxplot() of a particular 'Genre' category
```

In [192…

```
v8_b = sns.boxplot(data = movies[movies.Genre == 'Drama'], x = 'Year', y = 'Crit
plt.show()
```

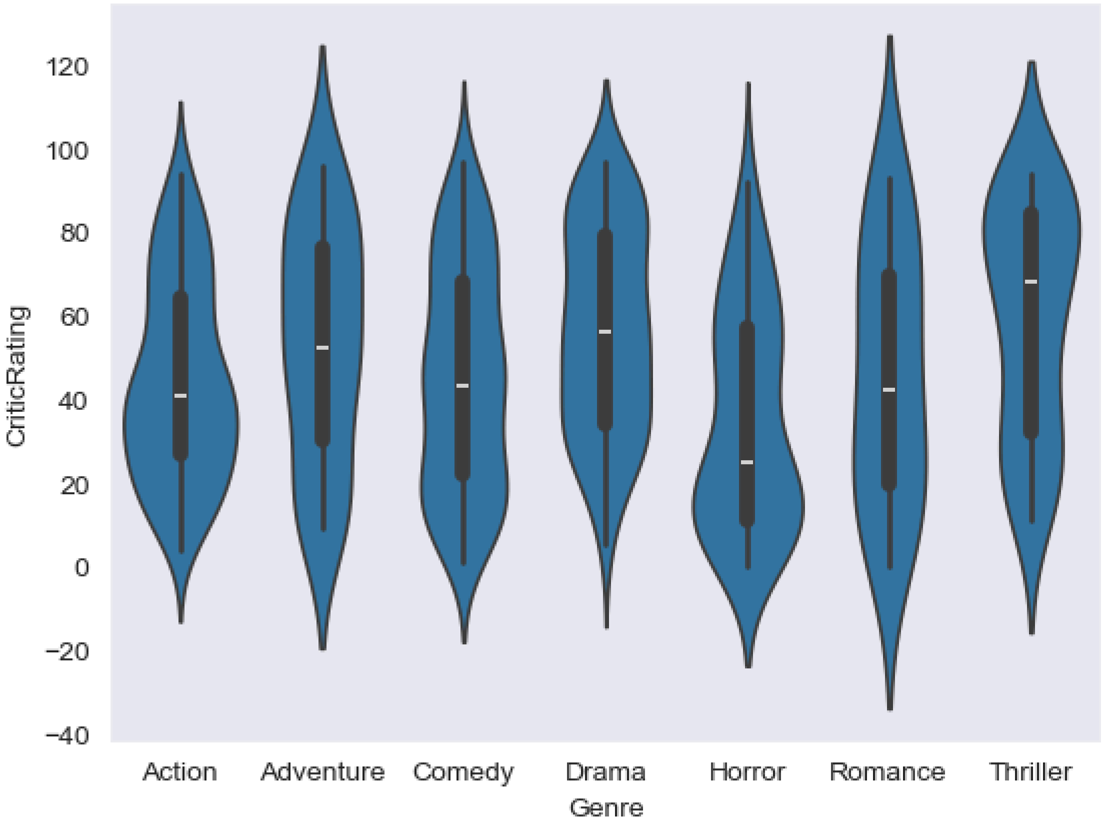


In [188…

```
# violinplot(), this is same as a boxplot(), it is just that the shape changes
```

In [190…

```
v9 = sns.violinplot(data = movies, x = 'Genre', y = 'CriticRating')
plt.show()
```

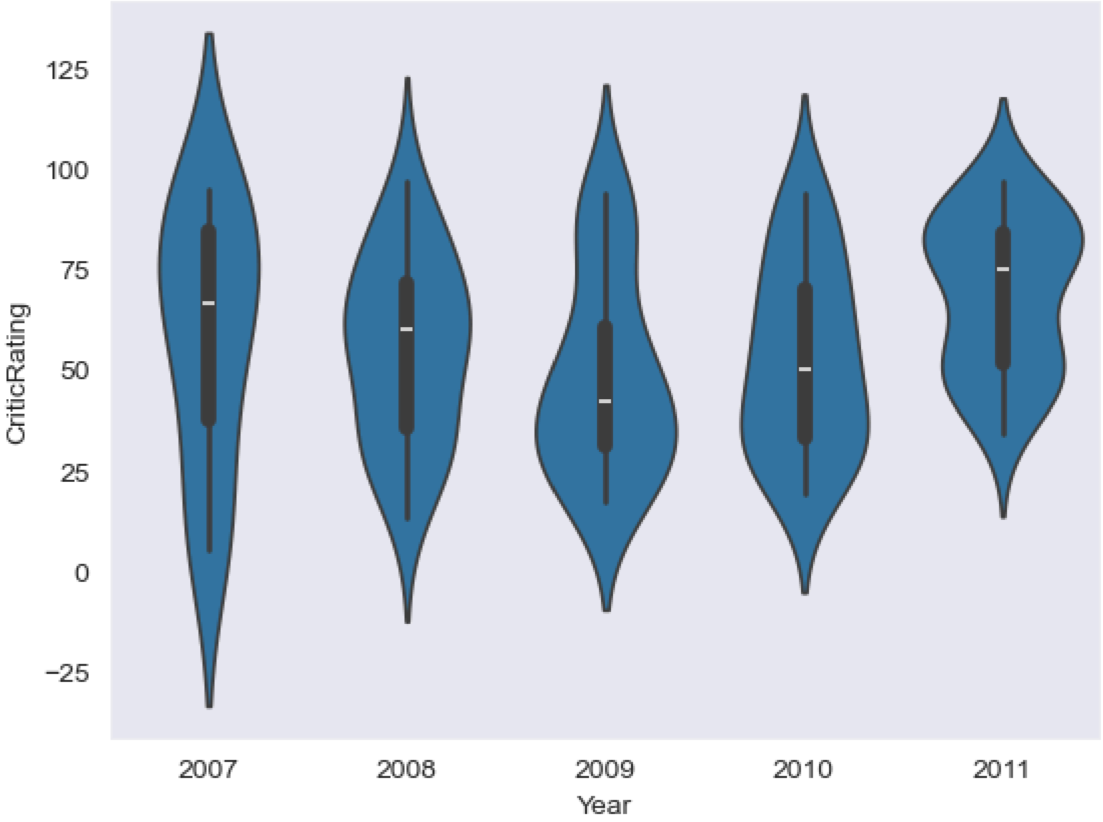


In [196…

```
# this code gives the violinplot() of a particular 'Genre' category
```

In [198…

```
v9_a = sns.violinplot(data = movies[movies.Genre == 'Drama'], x = 'Year', y = 'C
plt.show()
```

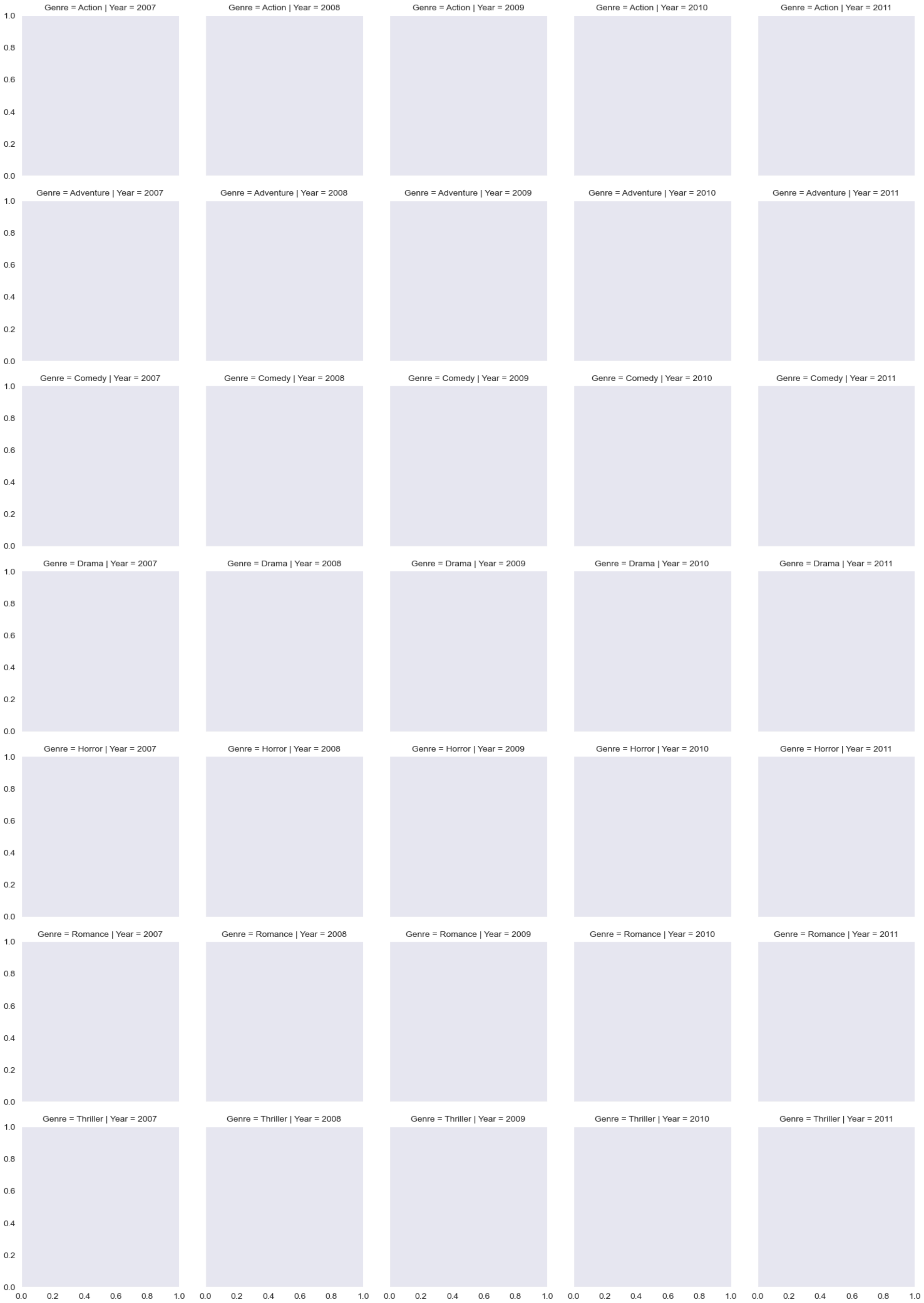


In [200…

```
# facetgrid(), creates a grid layout for multiple subplots
# the rows will be the dataset column values
# the columns will be the dataset column values
```

In [202…

```
v10 = sns.FacetGrid(movies, row = 'Genre', col = 'Year', hue = 'Genre')
plt.show()
```

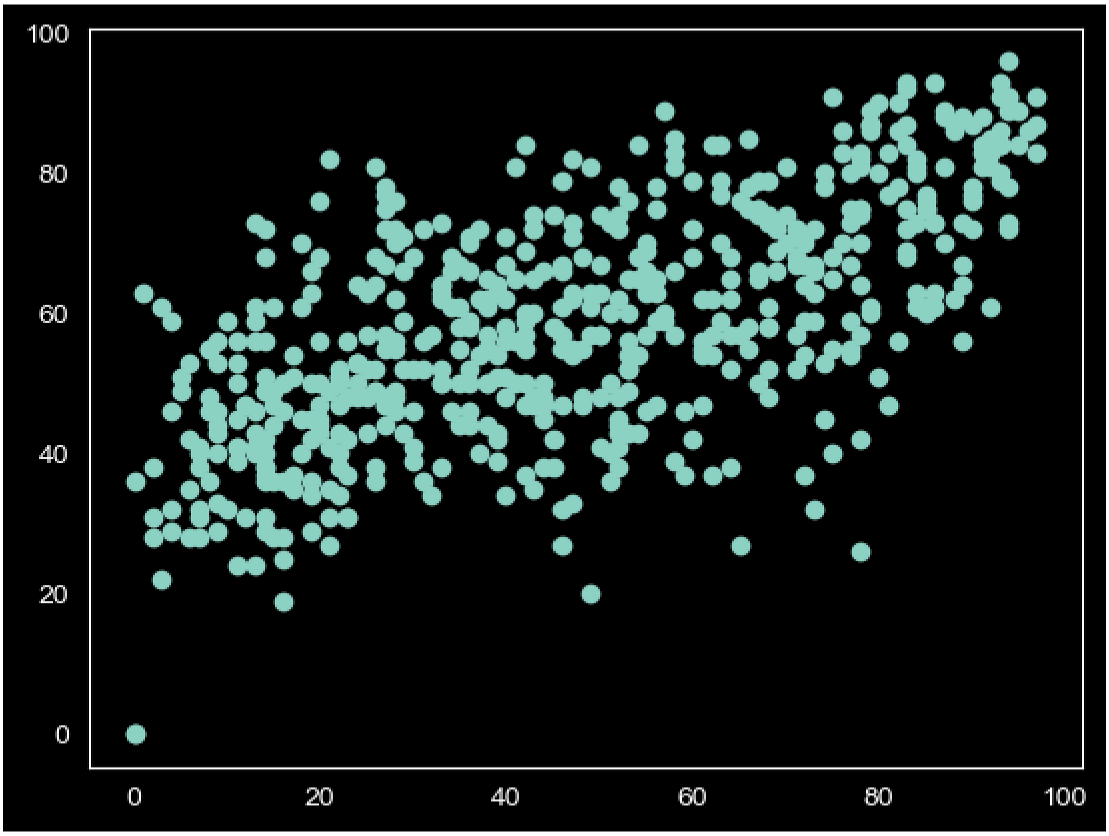


In [204…

```
# scatter(), creates a scatter plot of the column values
```

In [206…

```
plt.style.use('dark_background')
plt.scatter(movies.CriticRating,movies.AudienceRating)
plt.show()
```

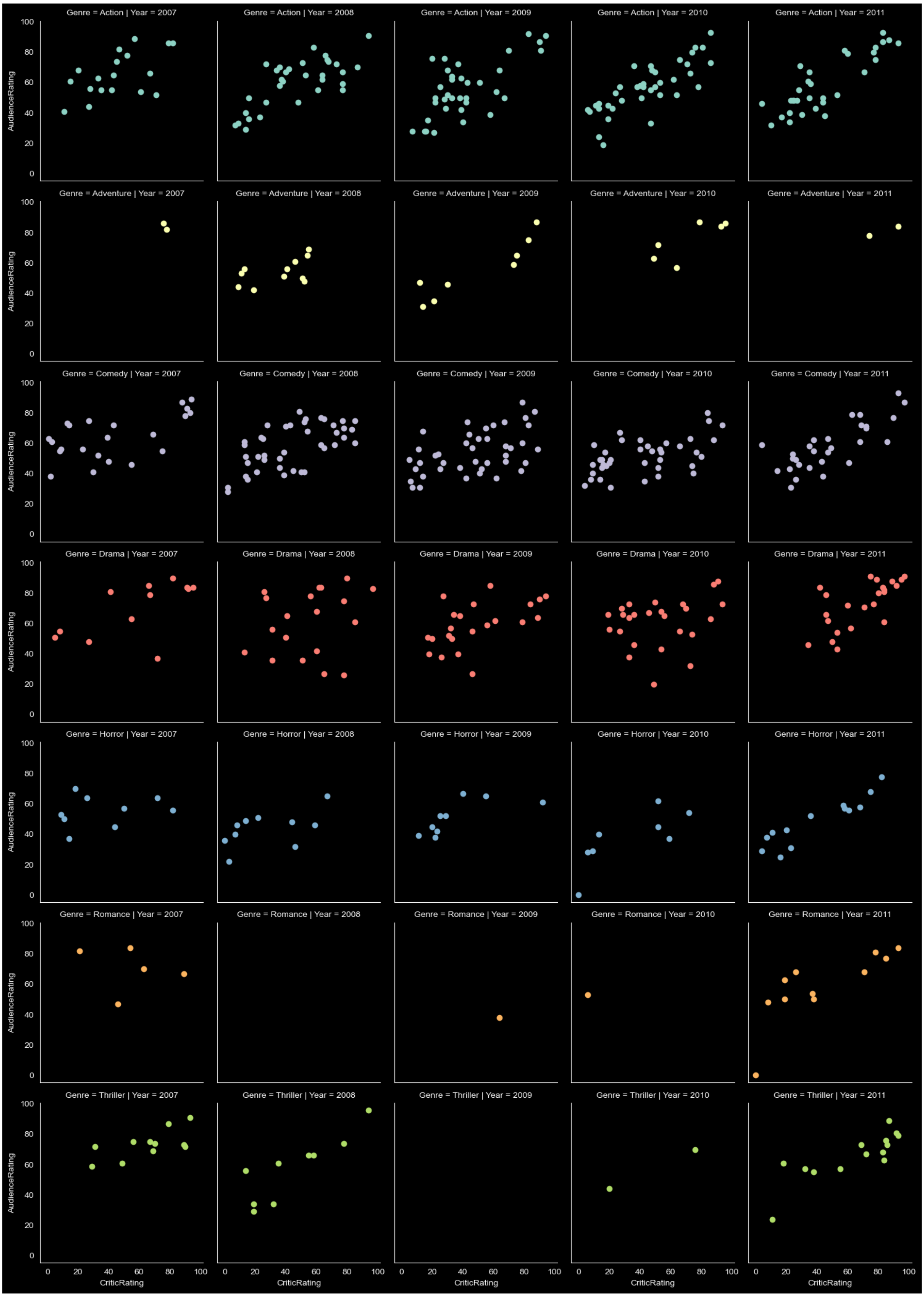


In [208…

```
# FacetGrid(), Creates a grid of subplots
# map(), Maps a plotting function (here, plt.scatter) to the grid
```

In [216…

```
plt.style.use('dark_background')
v11 = sns.FacetGrid(movies, row = 'Genre', col = 'Year', hue = 'Genre')
v11_a = v11.map(plt.scatter, 'CriticRating', 'AudienceRating')
plt.show()
```

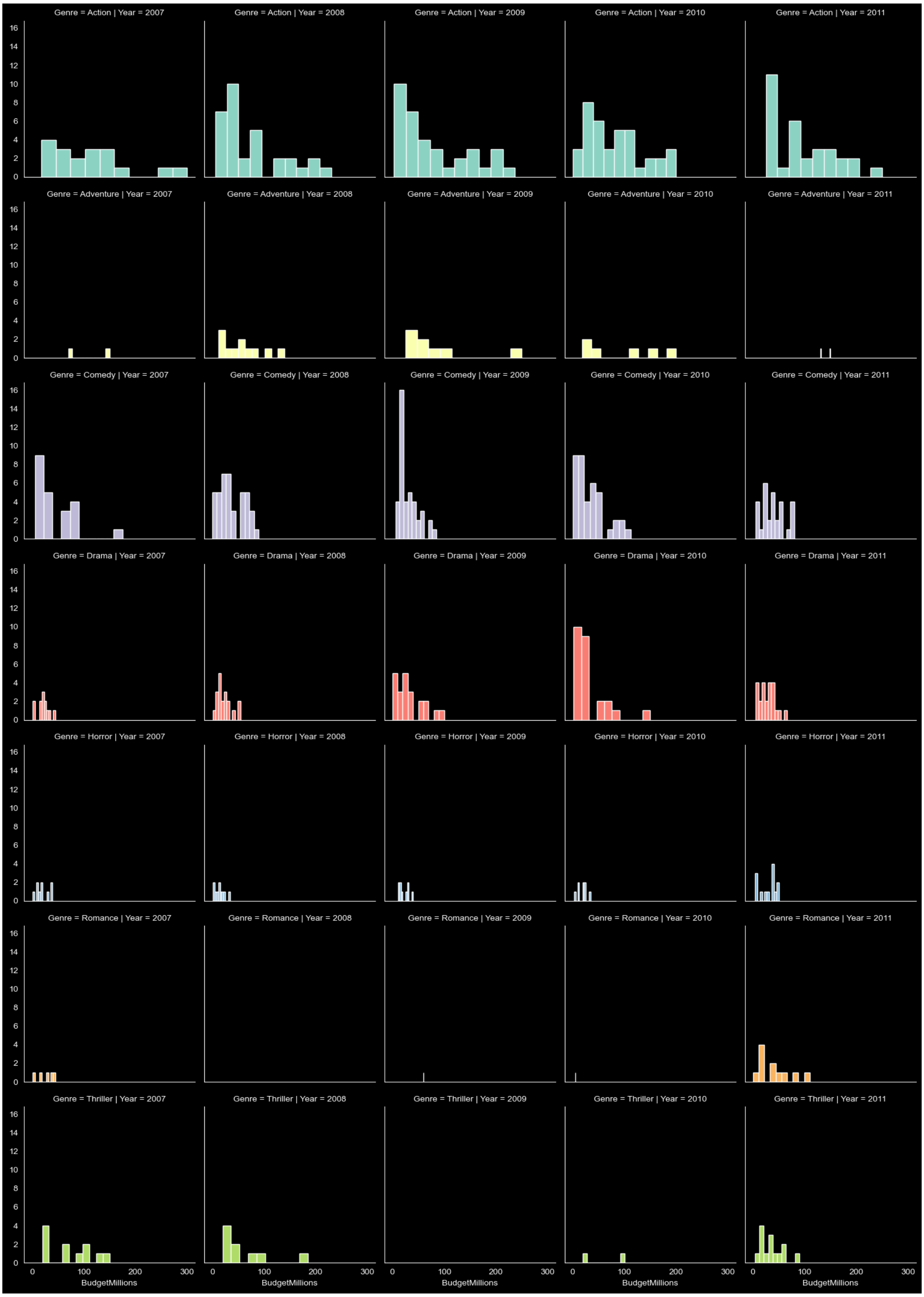


In [217…

```
# FacetGrid(), Creates a grid of subplots
# map(), Maps a plotting function (here, plt.hist) to the grid
```

In [218…

```
v11_b = sns.FacetGrid(movies, row = 'Genre', col = 'Year', hue = 'Genre')
v11_c = v11_b.map(plt.hist, 'BudgetMillions')
plt.show()
```

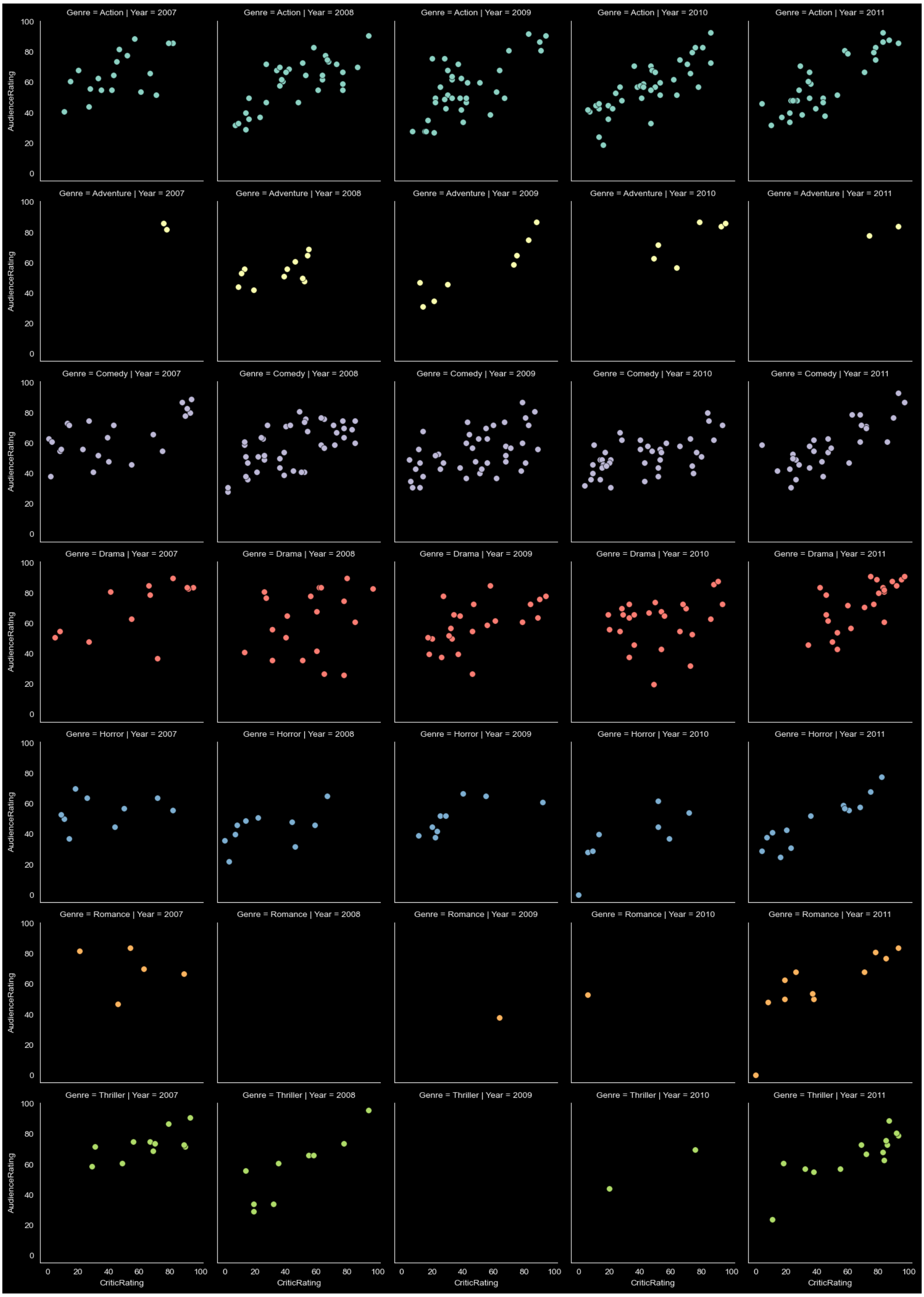


In [222…

```
# FacetGrid(), Creates a grid of subplots
# map(), Maps a plotting function (here, plt.scatter) to the grid
# kws, A dictionary of additional arguments passed to plt.scatter
```

In [224…

```
v11_d =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
kws = dict(s=50, linewidth=0.5,edgecolor='black')
v11_d = v11_d.map(plt.scatter, 'CriticRating', 'AudienceRating',**kws )
plt.show()
```



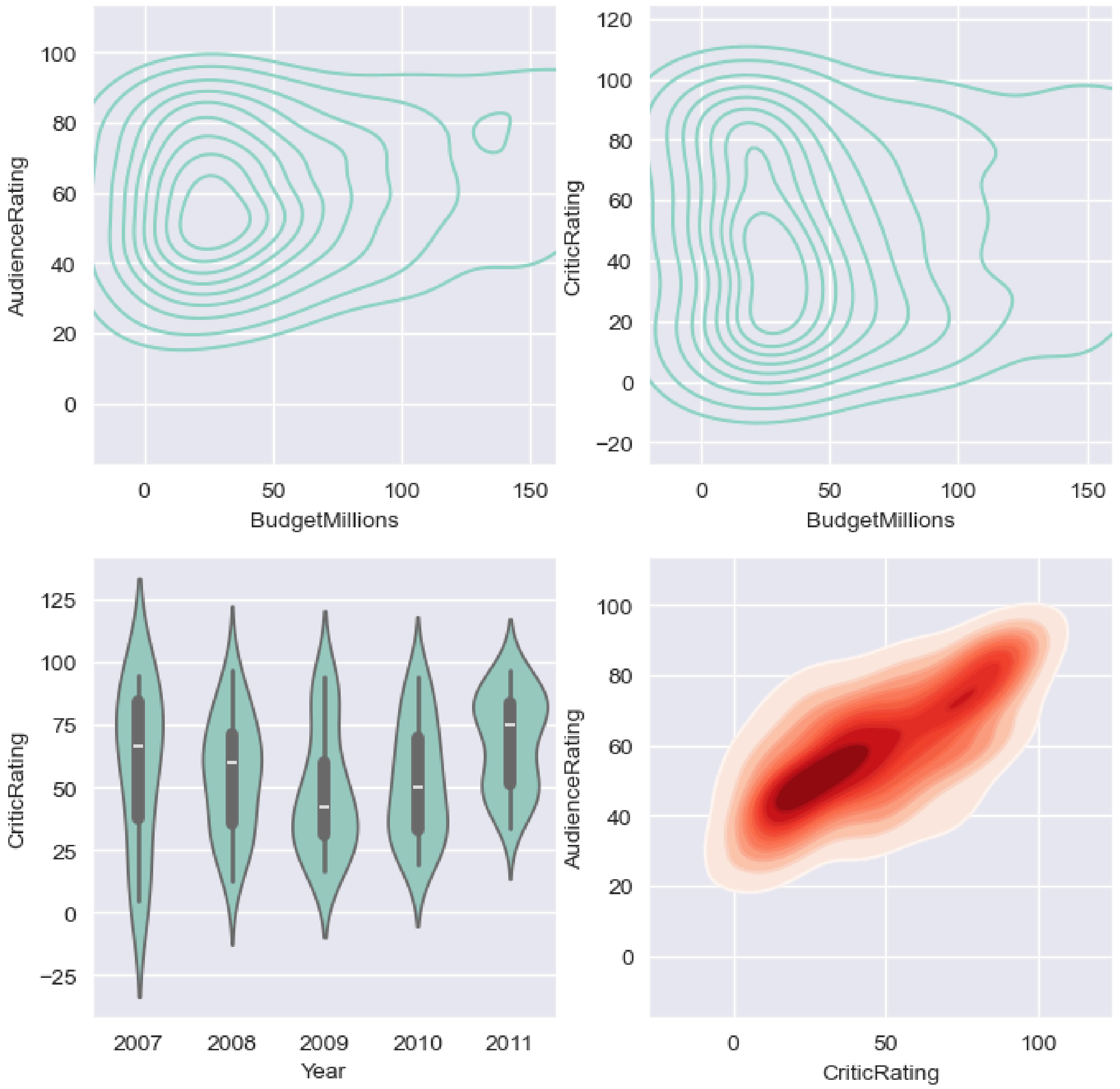
In [226…

```
# WE ARE PLOTTING THE SUBPLOTS(), KDEPLOT(), VIOLINPLOT()
```

In [228…

```
sns.set_style('darkgrid')
f, axes = plt.subplots (2,2, figsize = (8,8))

v12 = sns.kdeplot(x = movies.BudgetMillions,y = movies.AudienceRating,ax=axes[0,
v12_a = sns.kdeplot(x = movies.BudgetMillions,y = movies.CriticRating,ax = axes[

v12.set(xlim=(-20,160))
v12_a.set(xlim=(-20,160))
```

```
v13 = sns.violinplot(data=movies[movies.Genre=='Drama'], x='Year', y = 'CriticRa

v14 = sns.kdeplot(x = movies.CriticRating,y=movies.AudienceRating,shade = True,s

v14_a = sns.kdeplot(x = movies.CriticRating,y = movies.AudienceRating,cmap='Reds

plt.show()
```



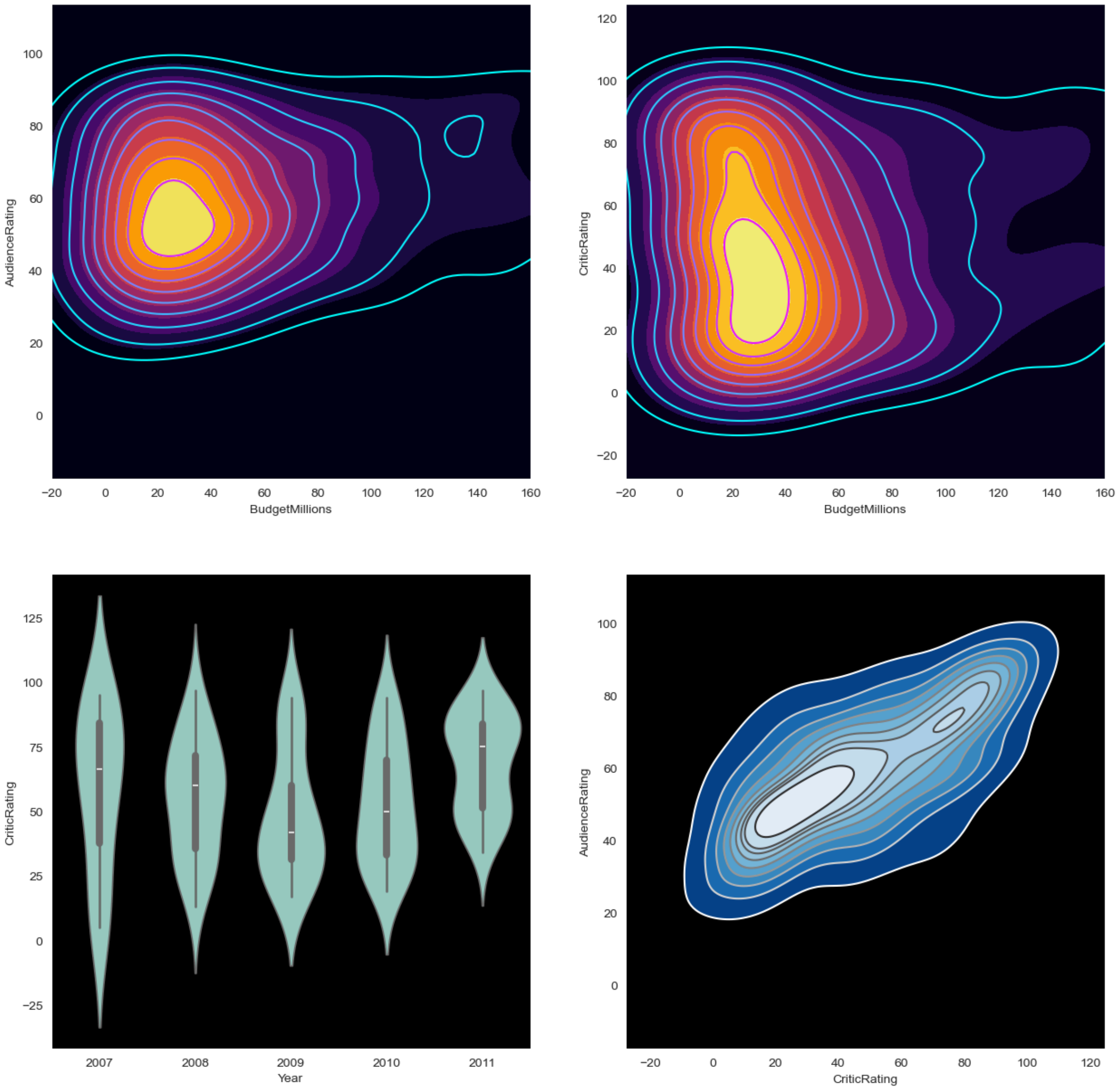
In [230…

```
# WE ARE PLOTTING THE SAME OUTPUT AS ABOVE, BUT WOTH SOME STYLE CHANGES
```

In [234…

```
sns.set_style('dark',{'axes.facecolor':'black'})
f, axes = plt.subplots (2,2, figsize = (15,15))

v12 = sns.kdeplot(x = movies.BudgetMillions,y = movies.AudienceRating, \
                 shade = True, shade_lowest=True,cmap = 'inferno', \
                 ax = axes[0,0])
v12_a = sns.kdeplot(x= movies.BudgetMillions,y =  movies.AudienceRating, \
                 cmap = 'cool',ax = axes[0,0])

v13 = sns.kdeplot(x= movies.BudgetMillions,y=movies.CriticRating,\
                 shade=True, shade_lowest=True, cmap='inferno',\
                 ax = axes[0,1])
v13_b = sns.kdeplot(x=movies.BudgetMillions,y=movies.CriticRating,\
                  cmap = 'cool', ax = axes[0,1])
```

```
v14 = sns.violinplot(data=movies[movies.Genre=='Drama'], \
                   x='Year', y = 'CriticRating', ax=axes[1,0])

v15 = sns.kdeplot(x = movies.CriticRating,y =movies.AudienceRating, \
                 shade = True,shade_lowest=False,cmap='Blues_r', \
                 ax=axes[1,1])

v15_b = sns.kdeplot(x = movies.CriticRating,y = movies.AudienceRating, \
                  cmap='gist_gray_r',ax = axes[1,1])


v12.set(xlim=(-20,160))
v13.set(xlim=(-20,160))

plt.show()
```


AudienceRating
BudgetMillions
CriticRating
BudgetMillions
CriticRating
Year
AudienceRating
CriticRating