In [1]:
```
# MOVIE RATING ANALYTICS ( ADVANCED VISULIZATION)

import pandas as pd
import os
```

In [3]:
```
os.getcwd() # if you want to change the working directory
```

Out[3]:
```
'C:\\Users\\swati\\OneDrive\\Documents\\Data_Analyst\\nov'
```

In [17]:
```
movie = pd.read_csv(r"C:\Users\swati\Downloads\Movie-Rating.csv")
movie
```

Out[17]:

	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 [19]:
```
len(movie)
```

Out[19]:
```
559
```

In [21]:
```
movie.head()
```

Out[21]:

	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 [23]:
```
movie.tail()
```

Out[23]:

	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 [25]:
```
movie.columns
```

Out[25]:
```
Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
       'Budget (million $)', 'Year of release'],
      dtype='object')
```

In [72]:
```
movie.columns = [ 'film', 'Genre','CriticRating','AudienceRating','BudgetMillion
```

In [74]:
```
movie.head() # Removied space & % removied noise characters
```

Out[74]:

	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

In [76]:
```
movie.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 [78]:
```
movie.describe()
# if you look at the year the data type is int but when you looknat the mean val
# we have to change to categroy type
# also from object datatype we will convert to category datatypes
```

Out[78]:

	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

In [80]:
```
movie['film']
# movie9'Audience Ratings %']
```

Out[80]:
```
0       (500) Days of Summer 
1                 10,000 B.C.
2                  12 Rounds 
3                   127 Hours
4                  17 Again 
                ...          
554             Your Highness
555           Youth in Revolt
556                    Zodiac
557               Zombieland 
558                 Zookeeper
Name: film, Length: 559, dtype: category
Categories (559, object): ['(500) Days of Summer ', '10,000 B.C.', '12 Rounds
', '127 Hours', ..., 'Youth in Revolt', 'Zodiac', 'Zombieland ', 'Zookeeper']
```

In [82]:
```
movie.film
```

Out[82]:
```
0      (500) Days of Summer 
1                10,000 B.C.
2                12 Rounds 
3                 127 Hours
4                 17 Again 
                   ...
554            Your Highness
555          Youth in Revolt
556                   Zodiac
557              Zombieland 
558                Zookeeper
Name: film, Length: 559, dtype: category
Categories (559, object): ['(500) Days of Summer ', '10,000 B.C.', '12 Rounds
', '127 Hours', ..., 'Youth in Revolt', 'Zodiac', 'Zombieland ', 'Zookeeper']
```

In [84]:
```
movie.film = movie.film.astype('category')
```

In [86]:
```
movie.film
```

Out[86]:
```
0      (500) Days of Summer 
1                10,000 B.C.
2                12 Rounds 
3                 127 Hours
4                 17 Again 
                   ...
554            Your Highness
555          Youth in Revolt
556                   Zodiac
557              Zombieland 
558                Zookeeper
Name: film, Length: 559, dtype: category
Categories (559, object): ['(500) Days of Summer ', '10,000 B.C.', '12 Rounds
', '127 Hours', ..., 'Youth in Revolt', 'Zodiac', 'Zombieland ', 'Zookeeper']
```

In [88]:
```
movie.head()
```

Out[88]:

	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

In [90]:
```
movie.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 [92]:
```
movie.Genre = movie.Genre.astype('category')
movie.Year = movie.Year.astype('category')
```

In [94]:
```
movie.Genre
```

Out[94]:
```
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 [96]:
```
movie.Year # is it real no. year you can take average,min,max but out come have
```

Out[96]:
```
0      2009
1      2008
2      2009
3      2010
4      2009
       ... 
554    2011
555    2009
556    2007
557    2009
558    2011
Name: Year, Length: 559, dtype: category
Categories (5, int64): [2007, 2008, 2009, 2010, 2011]
```

In [98]:
```
movie.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 [100…

```
movie.Genre.cat.categories
```

Out[100…

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

In [102…

```
movie.describe()
# now when you see the describe you will get only integer mean, standard deviati
```

Out[102…

	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

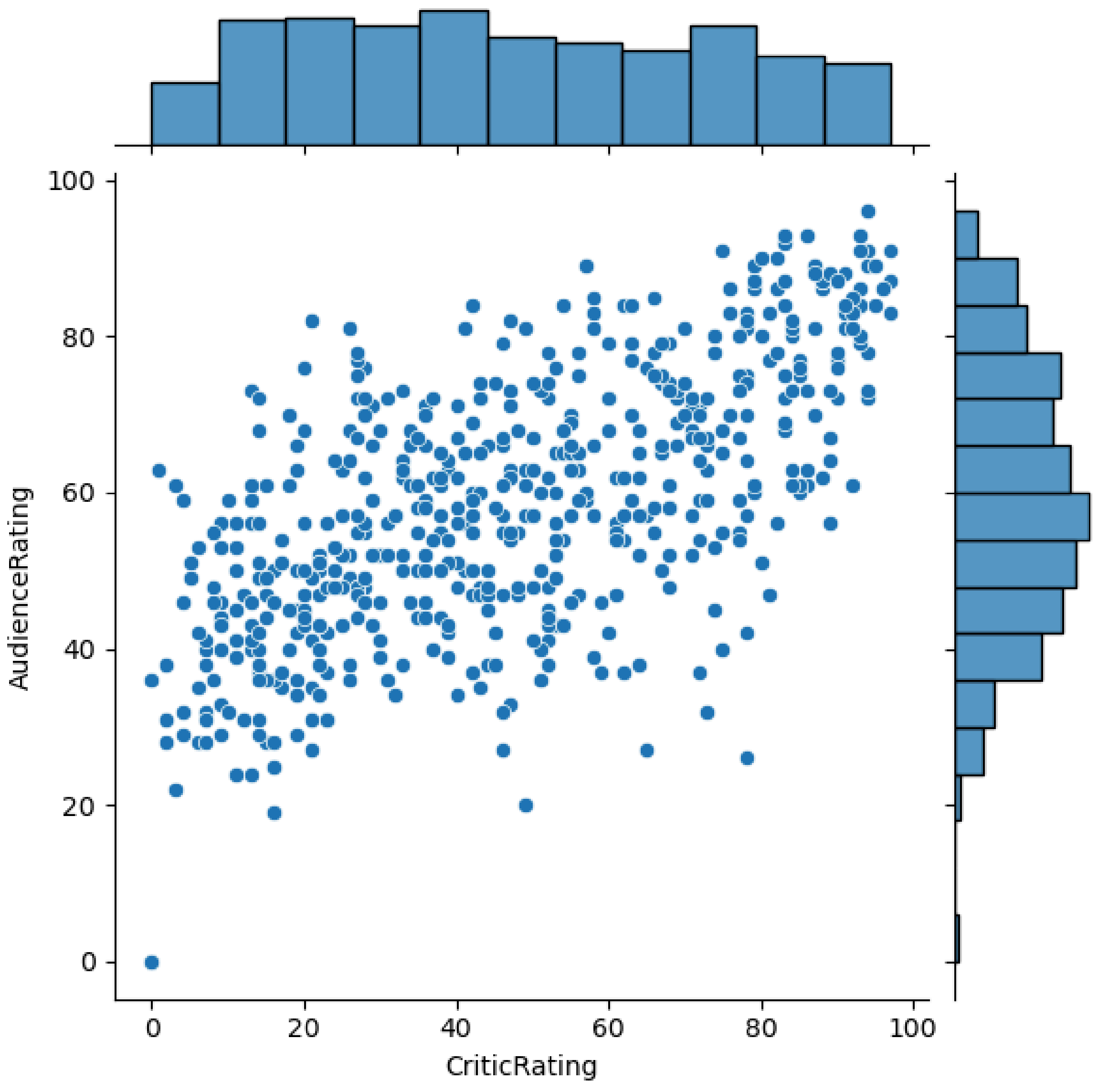
In [104…

```
# hpe to working with joint plots

from matplotlib import pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

- basically joint plot is a scatter plot & it find the relation b/w audience & critics
- also if you look up you can find the uniform distribution(critics) and normal distribution (audience)

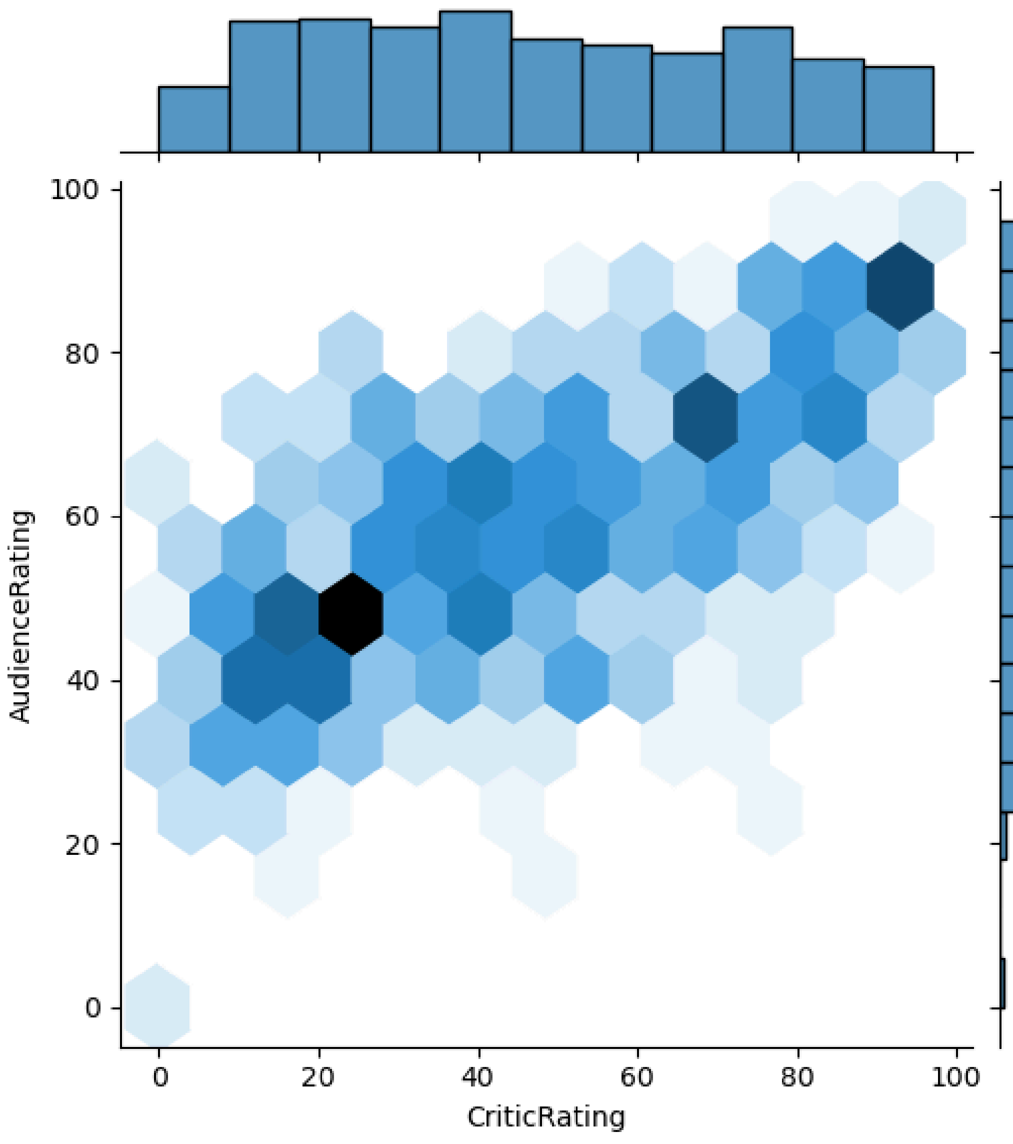
In [107…

```
j = sns.jointplot ( data = movie, x = 'CriticRating', y = 'AudienceRating')
# Audience rating is more dominant then critics rating
# Based on this we find out as most people are most likihood to watch audience r
# let me explain the  - if you filter audience rating & critic rating. critic ra
```



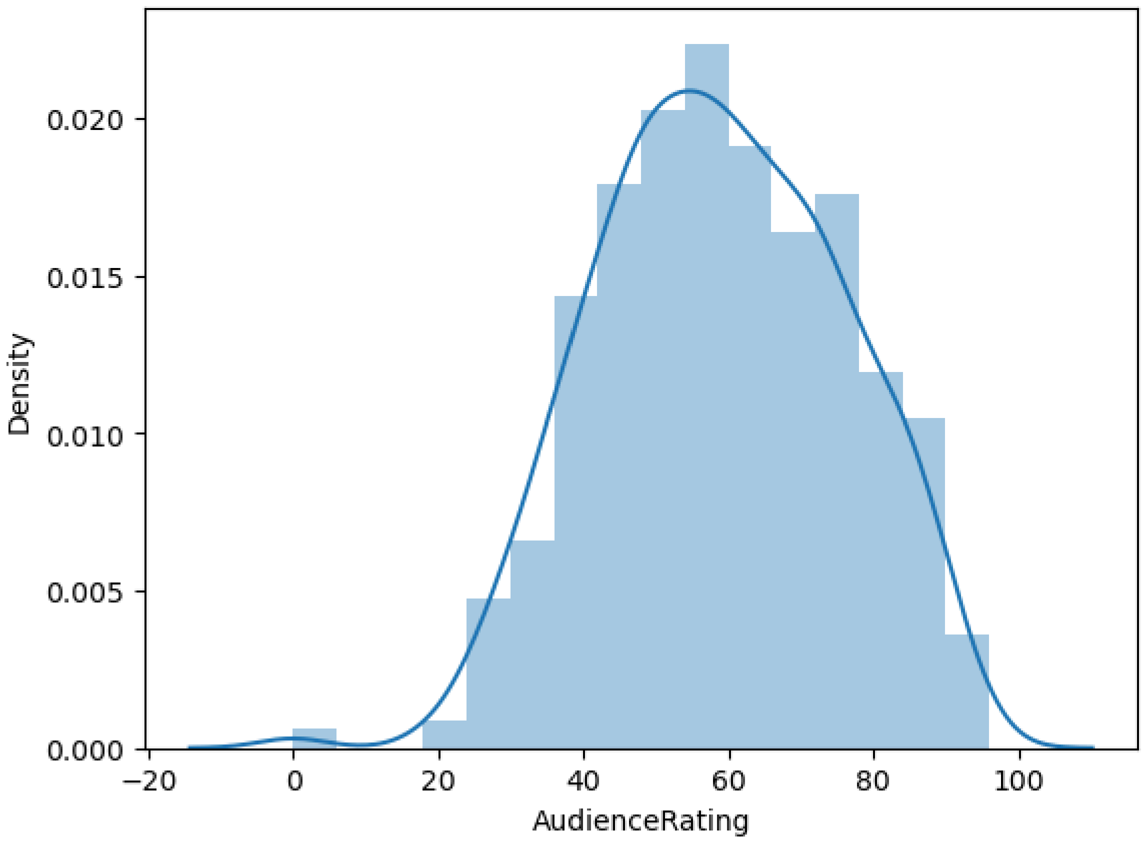
In [111…

```
j = sns.jointplot( data = movie, x = 'CriticRating', y = 'AudienceRating', kind=
```



In [113…

```
# Histograms

# <<< chat 1

m1 = sns.distplot( movie.AudienceRating)

#y - axis generated by seaborn automatically thet is the powerfull of seaborn ga
```

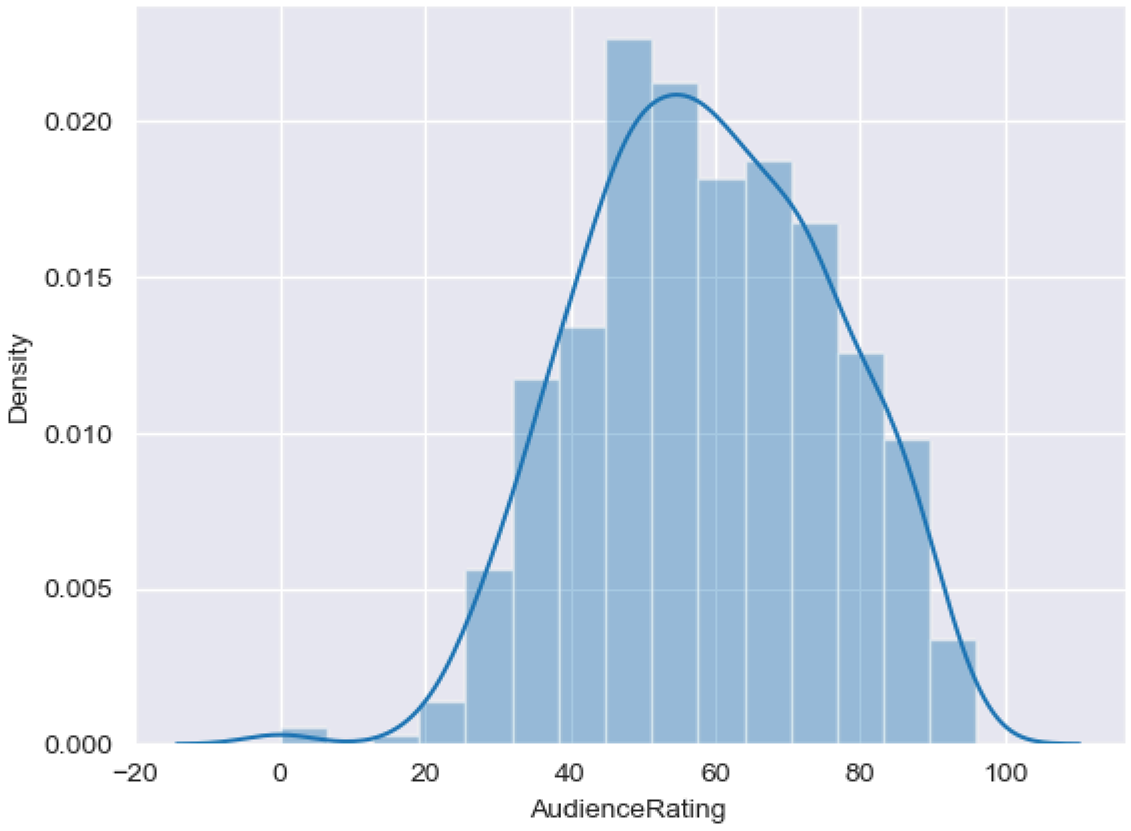


In [115…

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

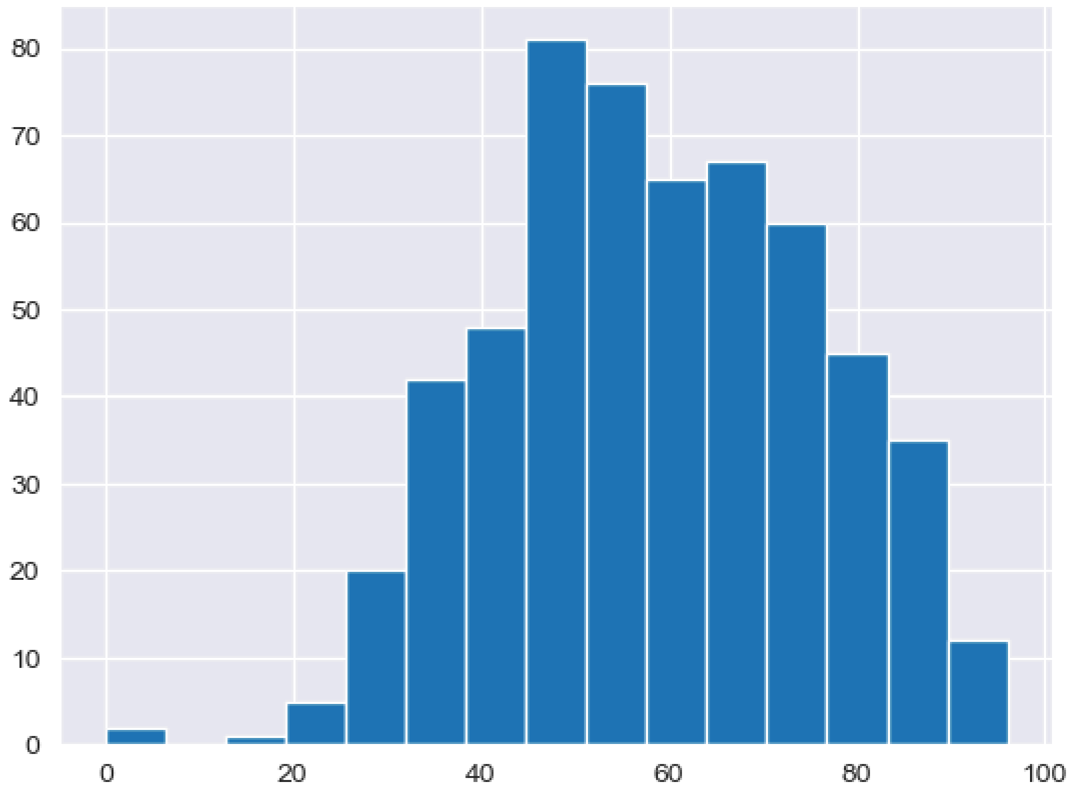
In [119…

```
m2 = sns.distplot(movie.AudienceRating, bins = 15)
```



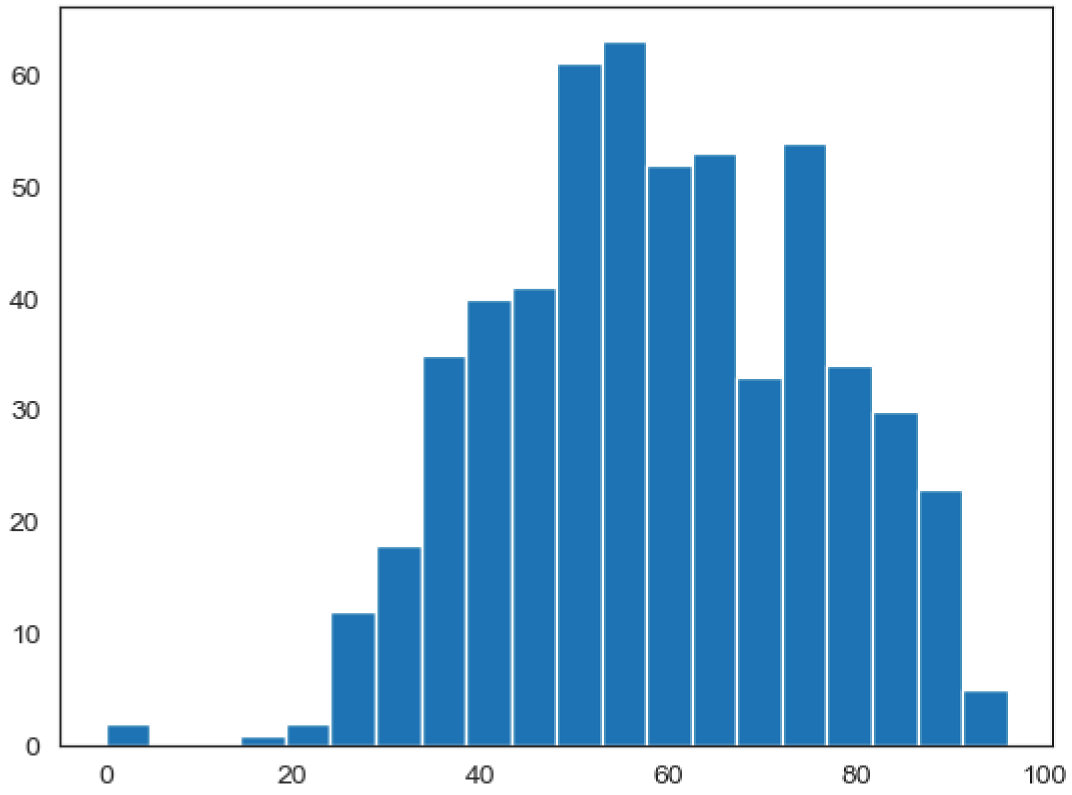
In [121…

```
# sns.set _ style('darkgrid')
n1 = plt.hist(movie.AudienceRating, bins=15)
```



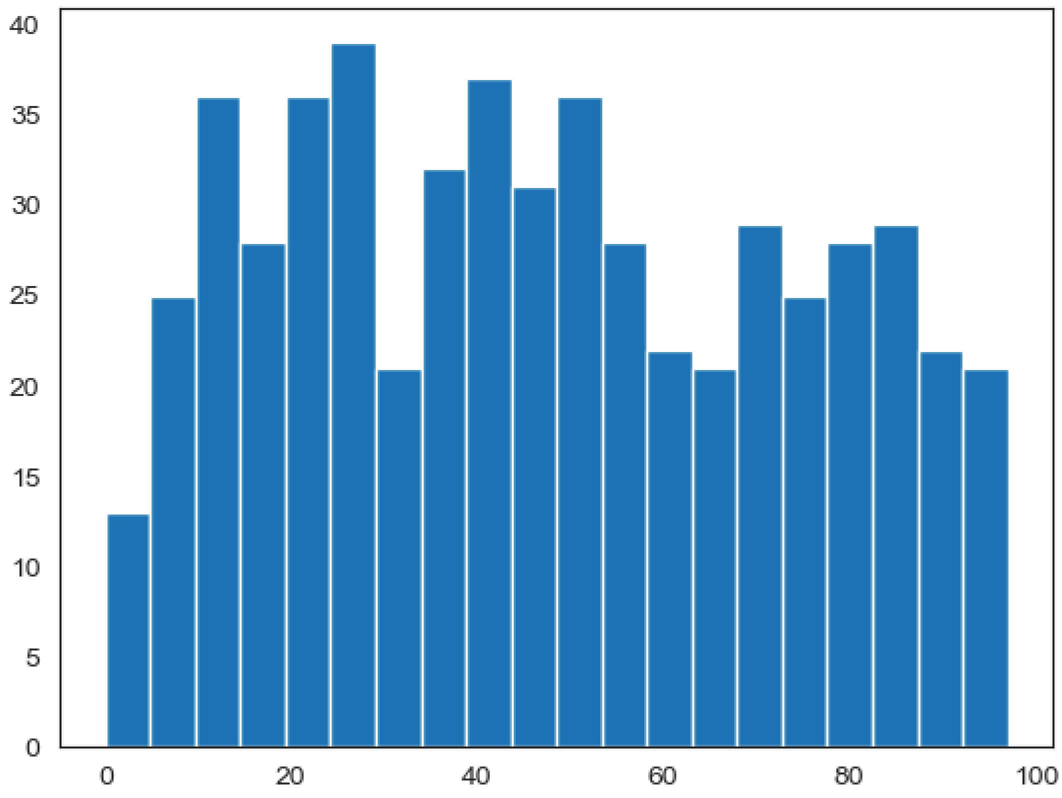
In [123…

```
sns.set_style('white') # normal distribution & called as bell curve
n1 = plt.hist(movie.AudienceRating, bins=20)
```



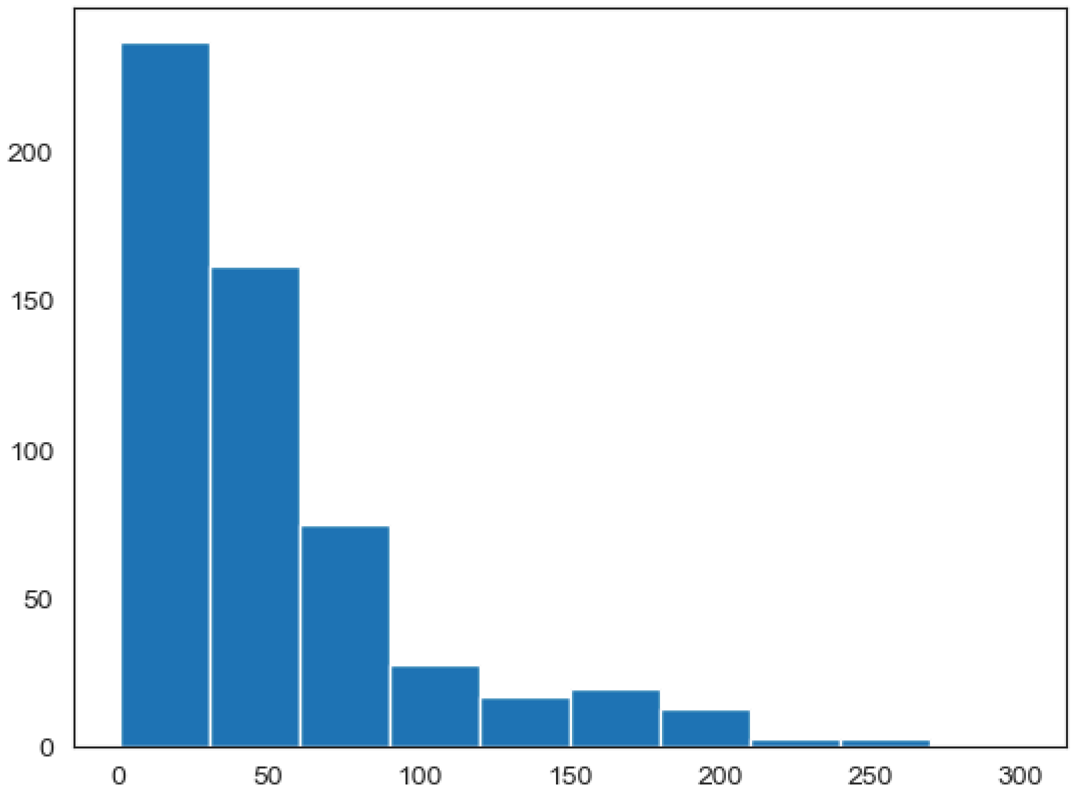
In [125…

```
n1 = plt.hist(movie.CriticRating, bins=20) # uniform distribution
```



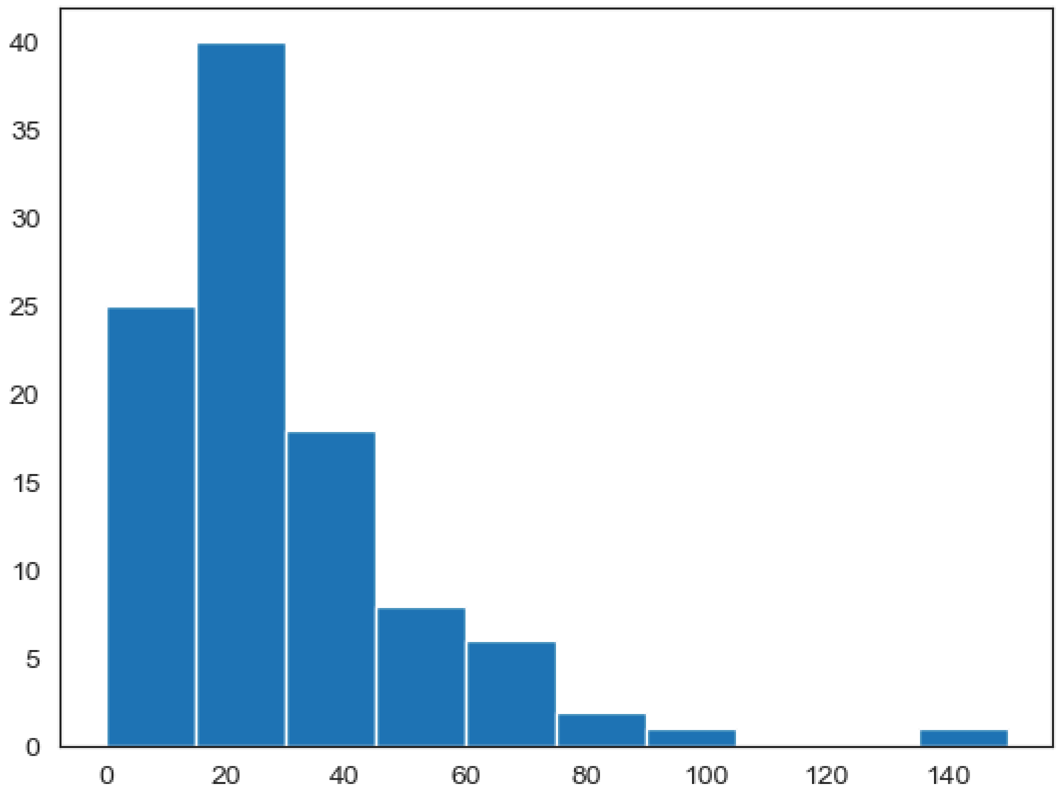
In [127…
```
# <<< chat - 2

# Creating stacked histograms & this is bit tough to understand
```

In [131…
```
#h1 = plt.hist(movie.BudgetMillions)

plt.hist(movie.BudgetMillions)
plt.show()
```



In [137…

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



In [139…

```
movie.head()
```

Out[139…

	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

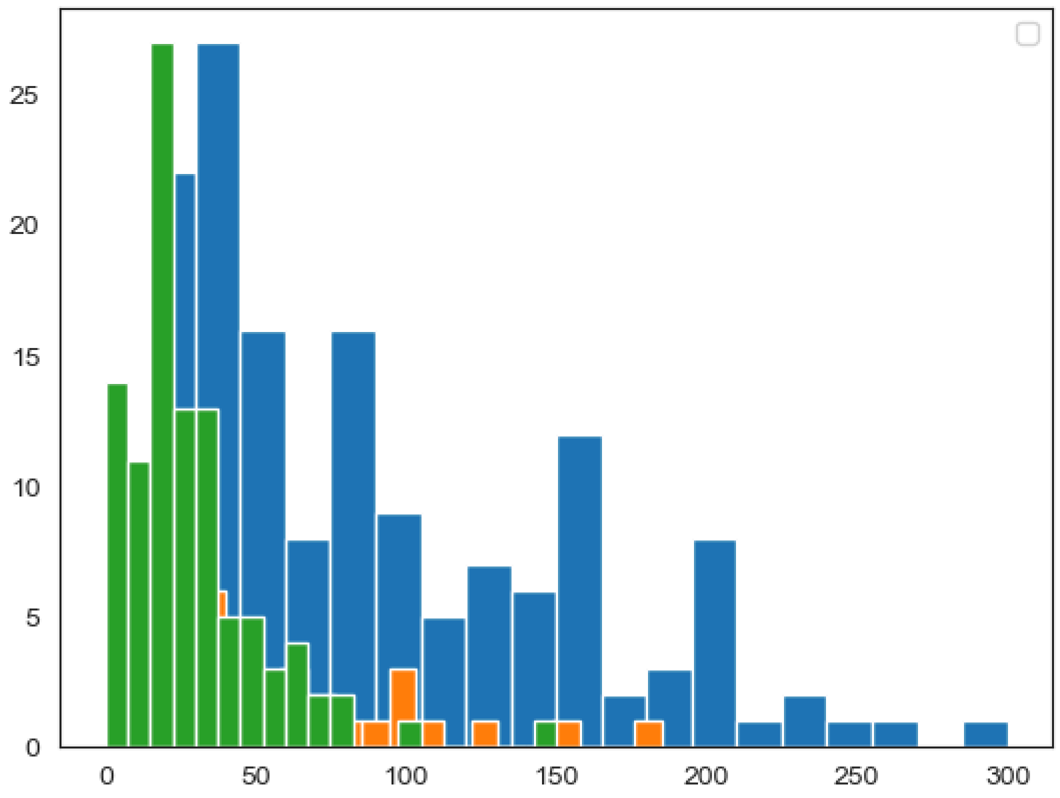
In [ ]:

```
# movie.Genre.unique()
```

In [141…

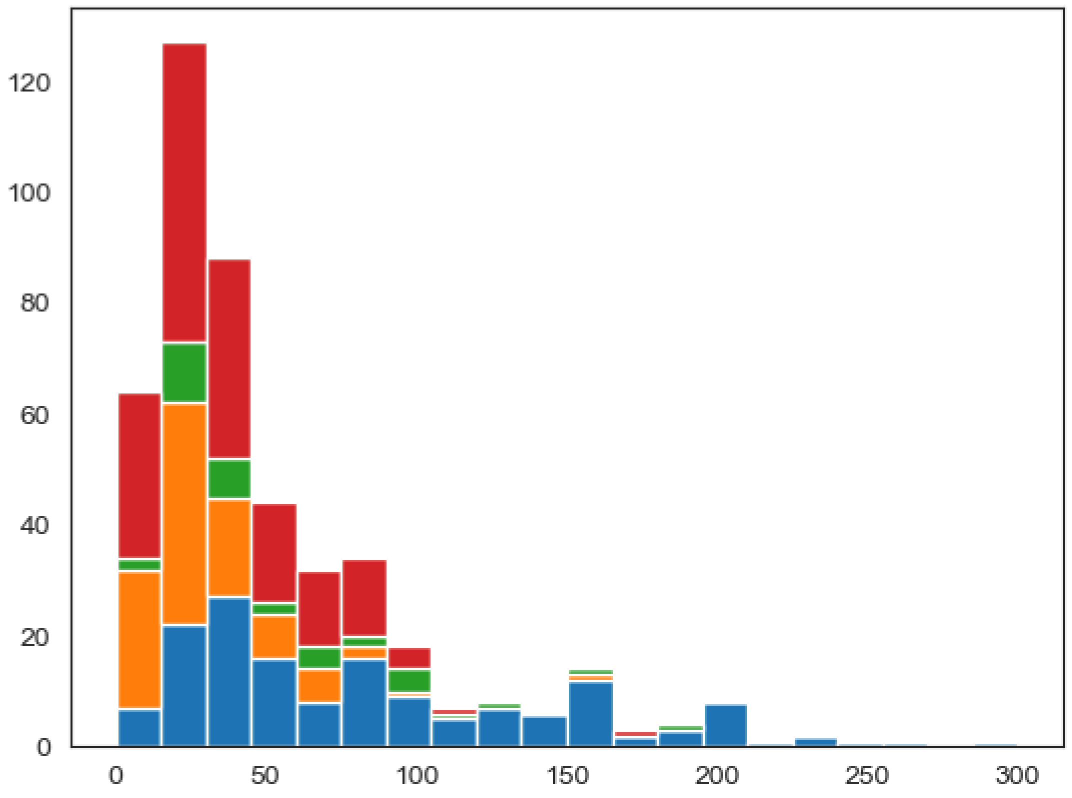
```
# Below plot are stacked histogram becuase overlaped

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

```
No artists with labels found to put in legend.  Note that artists whose label sta
rt with an underscore are ignored when legend() is called with no argument.
```



In [155…

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



In [157…

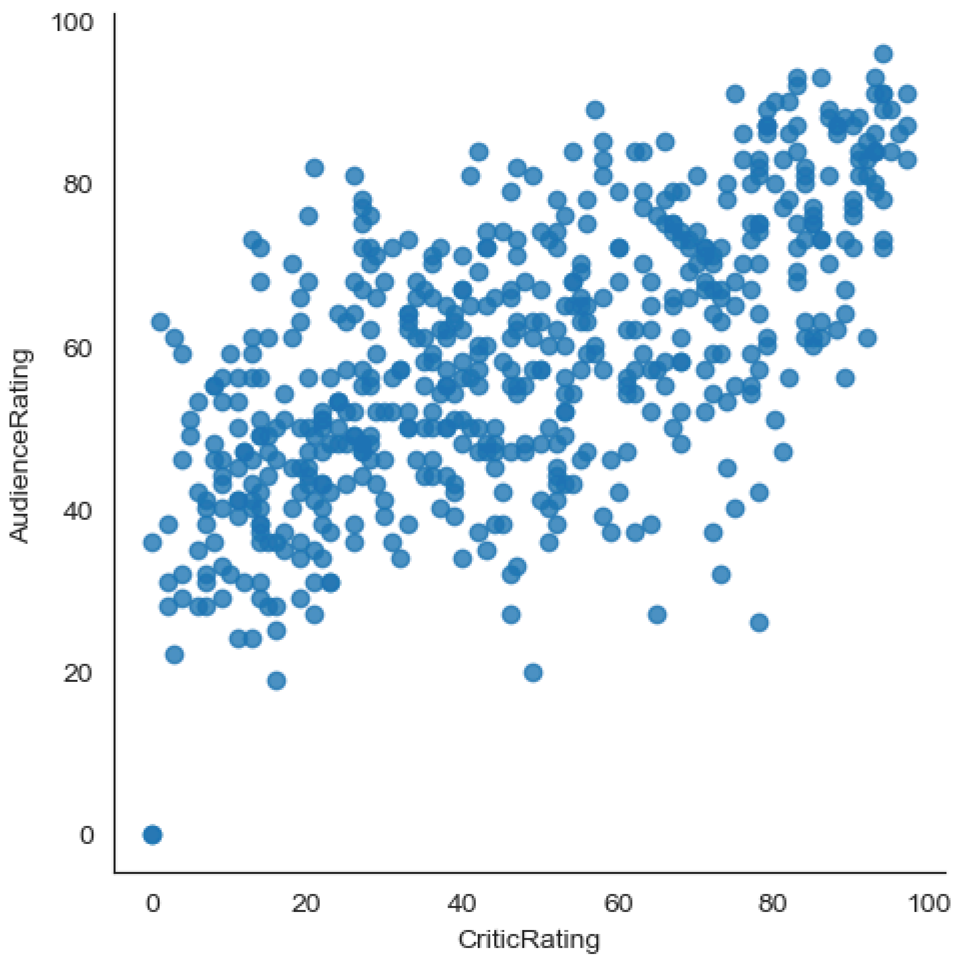
```
# if you have 100 categories you cannot copy & paste all the things
```

```
for gen in movie.Genre.cat.categories:
    print(gen)
```

```
Action
Adventure
Comedy
Drama
Horror
Romance
Thriller
```

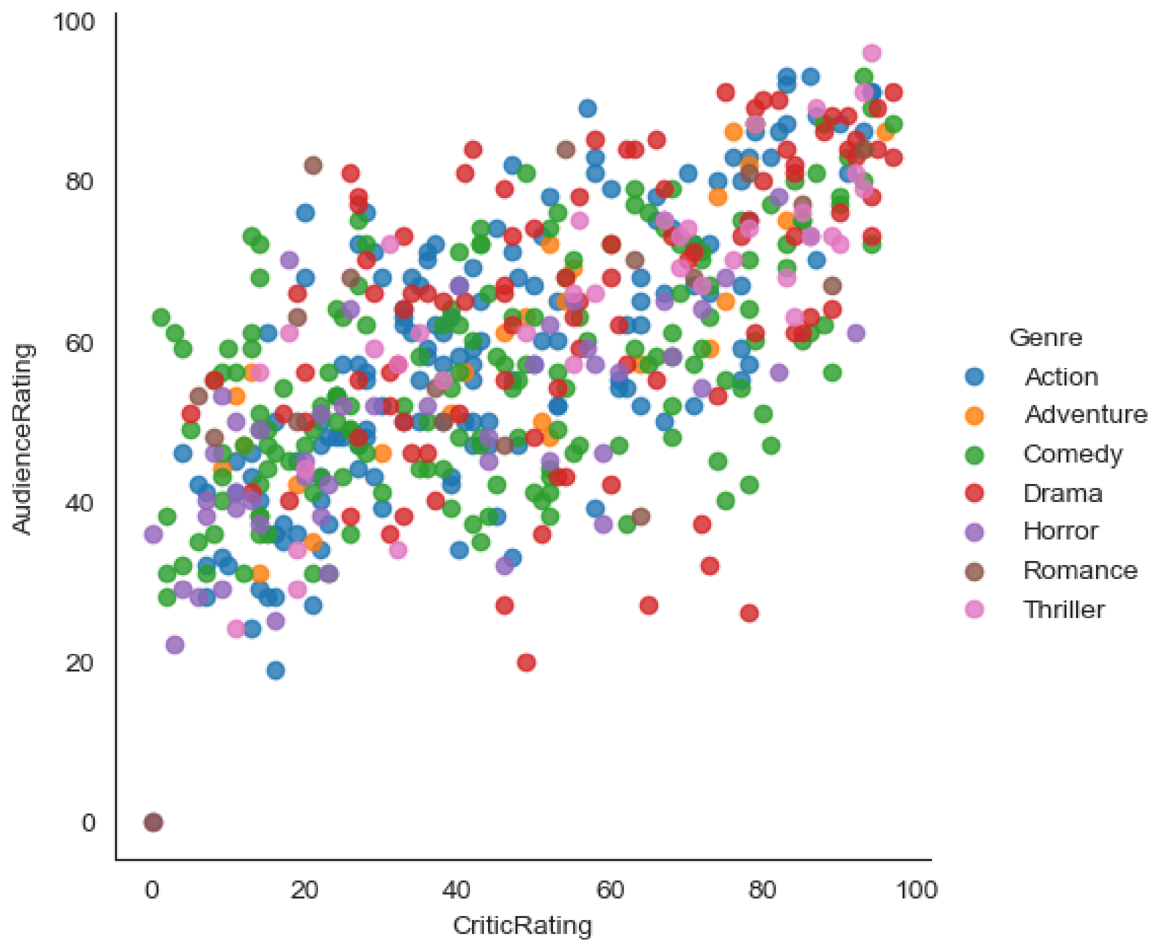
In [159…

```
vis1 = sns.lmplot(data=movie, x = 'CriticRating', y = 'AudienceRating',\
                  fit_reg=False)
```



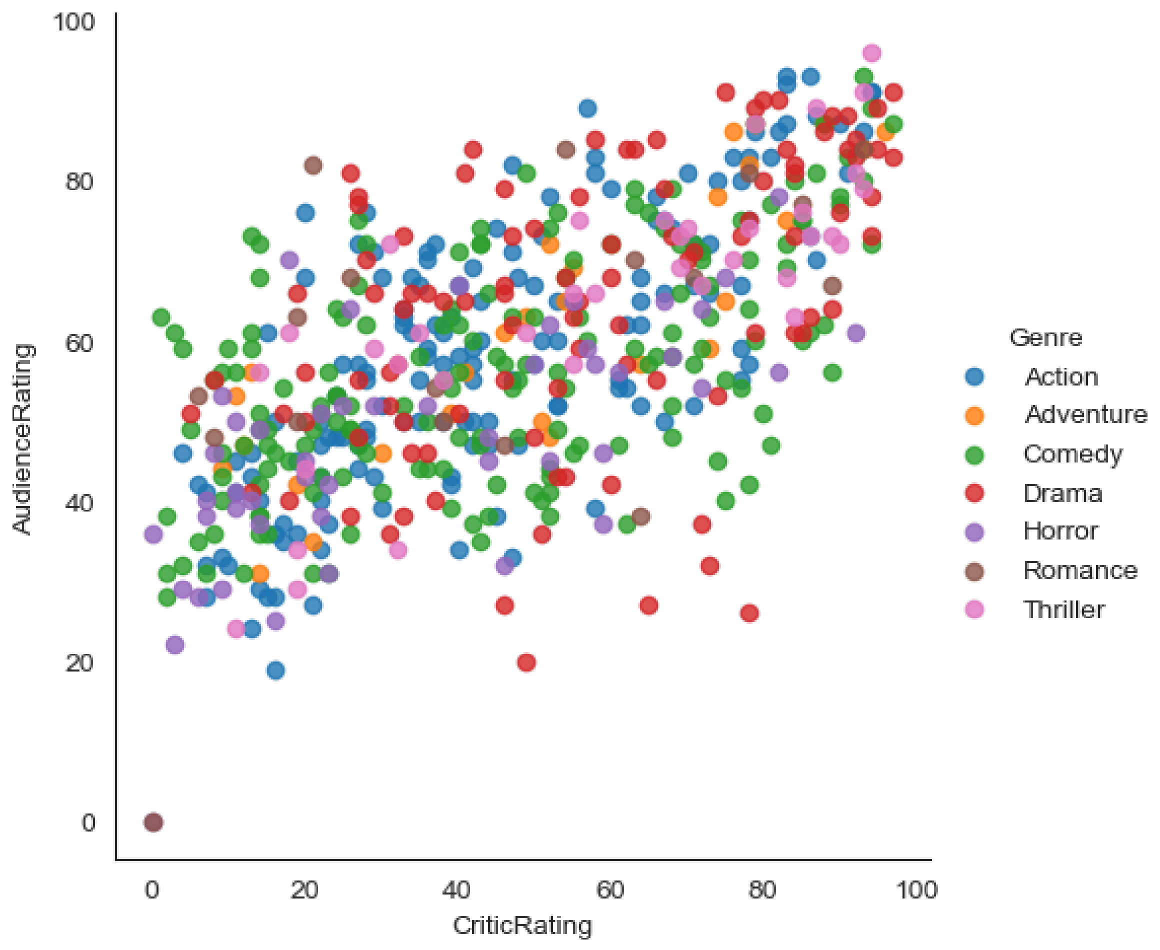
In [165…

```
vis1 = sns.lmplot(data=movie, x = 'CriticRating', y = 'AudienceRating',\
                  fit_reg=False, hue = 'Genre')
```



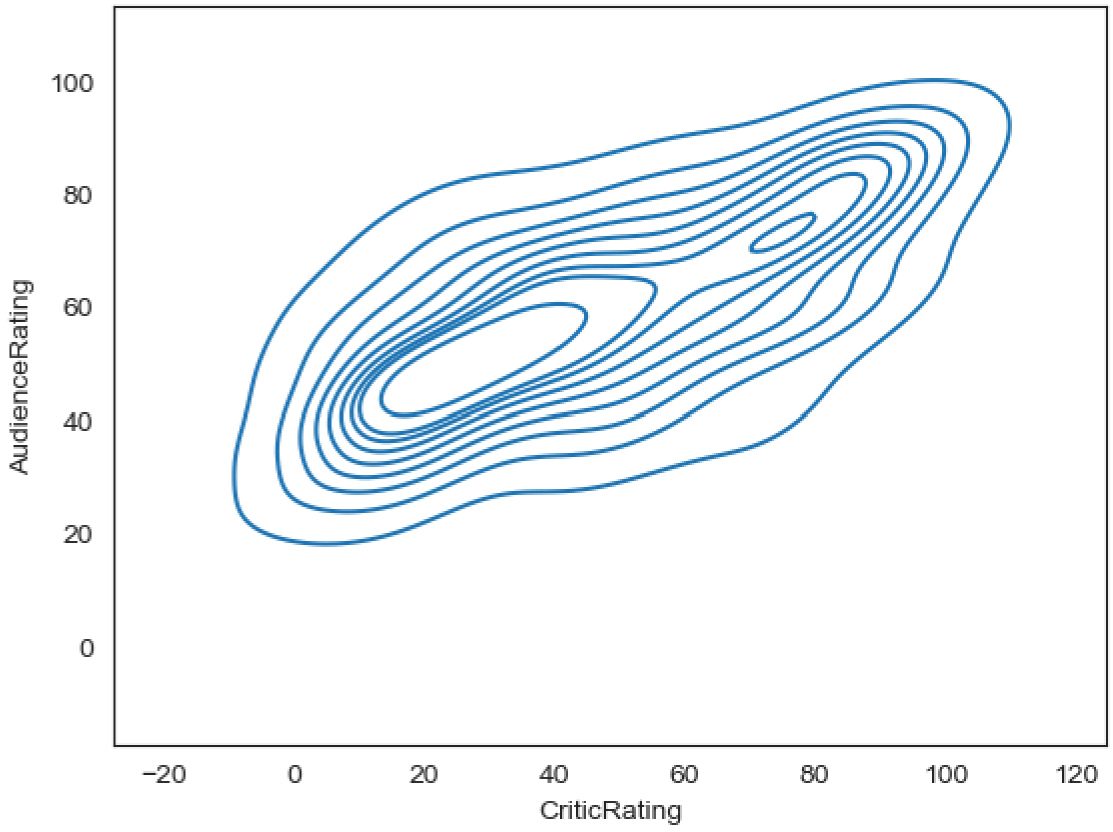
In [175…

```
vis1 = sns.lmplot(data=movie, x='CriticRating', y='AudienceRating',\
                  fit_reg=False, hue = 'Genre',aspect=1)
```



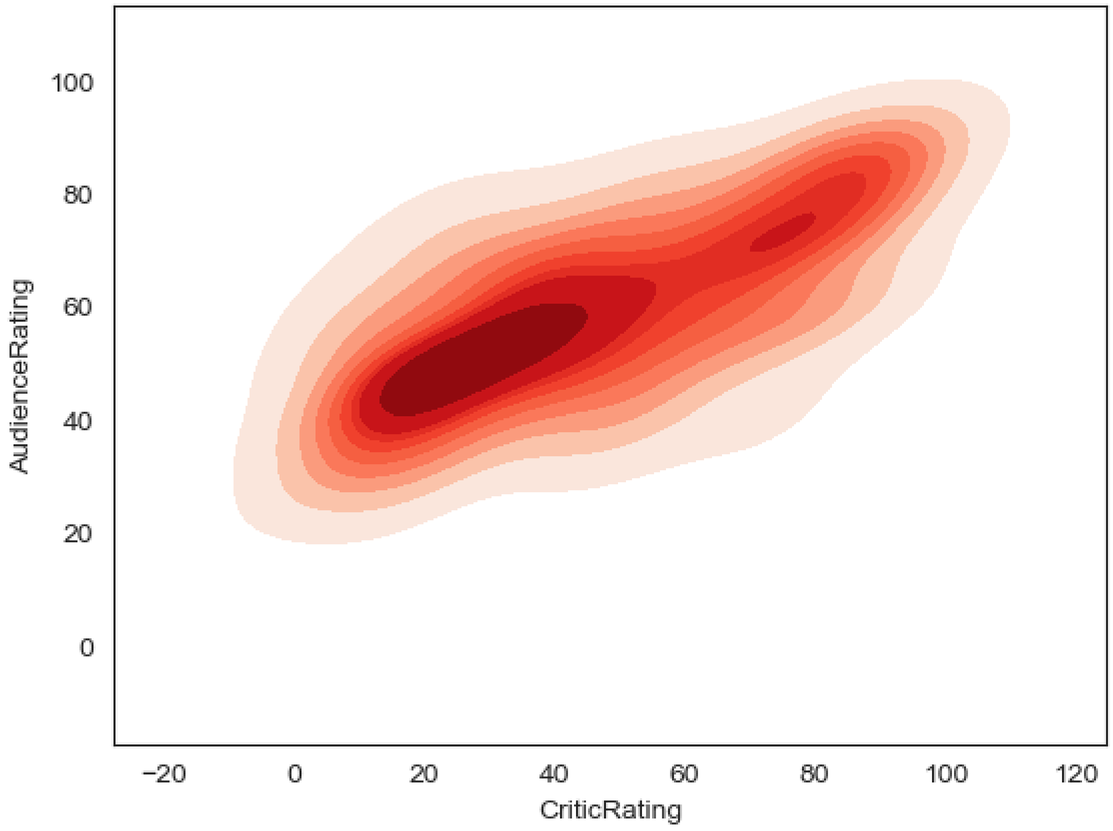
In [177…
```
# kernal Density Estimate plot ( KDE PLOT)
# how can i visulize audience rating & critics rating. using scatterplot
```

In [187…
```
k1 = sns.kdeplot(x = movie.CriticRating,y = movie.AudienceRating)

# where do u find more density and how density is distibuted across from the the
# center point is kernal this is calld KDE & insteade of dots it visualize like
# we can able to clearly see the spread at the audience ratings
```



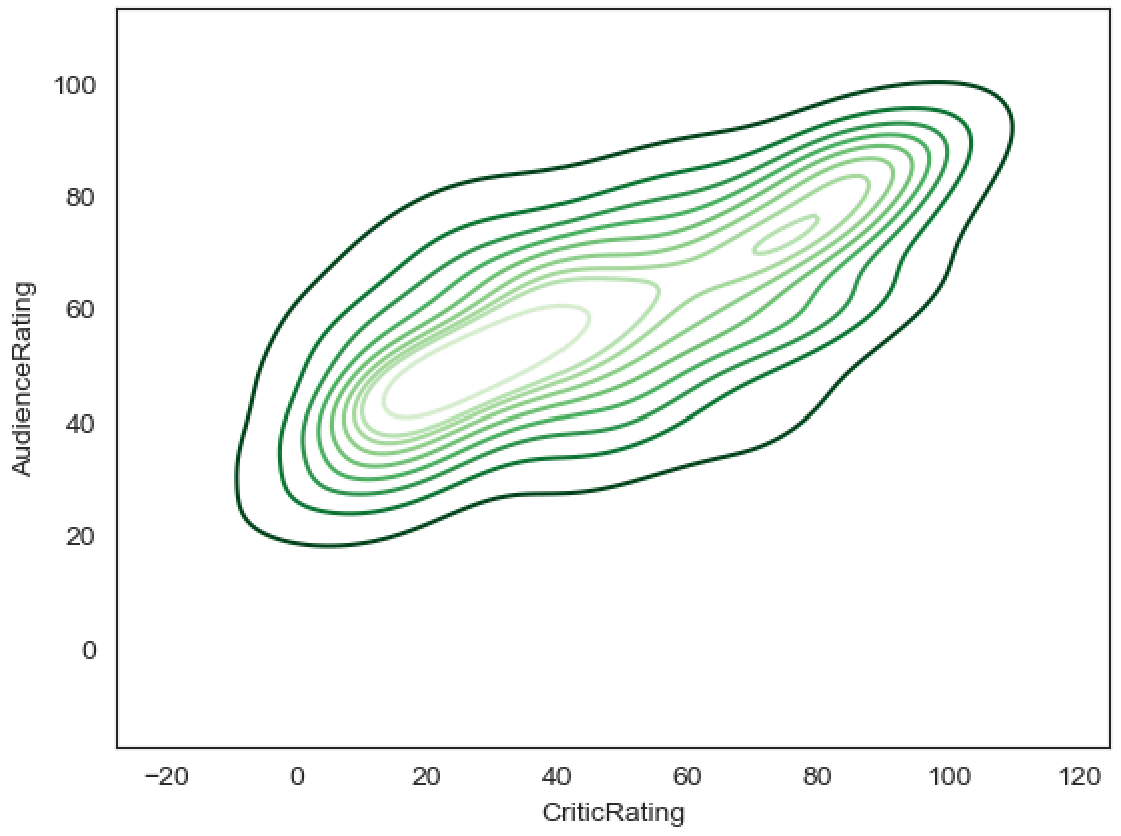
In [199…

```
k1 = sns.kdeplot(x = movie.CriticRating,y = movie.AudienceRating,shade = True,sh
```



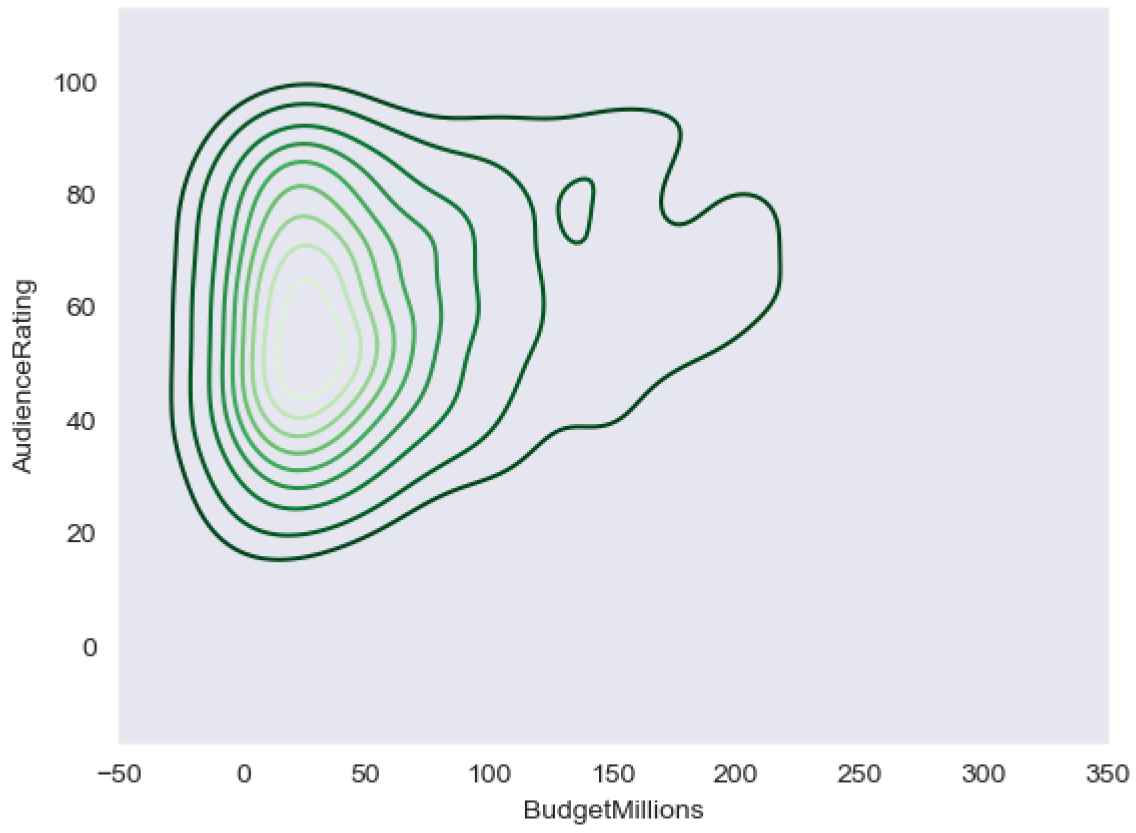
In [201…

```
k2 = sns.kdeplot(x = movie.CriticRating,y = movie.AudienceRating,shade_lowest=Fa
```



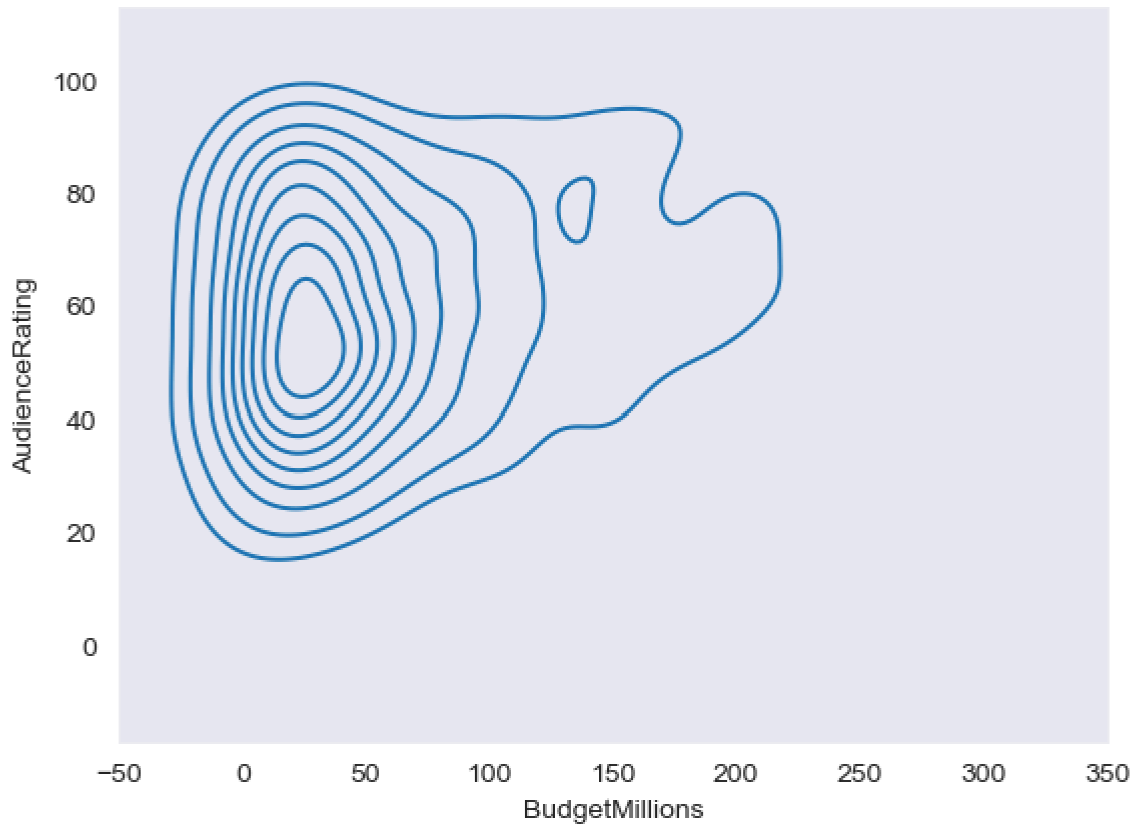
In [205…

```
sns.set_style('dark')
k1 = sns.kdeplot(x = movie.BudgetMillions,y = movie.AudienceRating,shade_lowest=
```

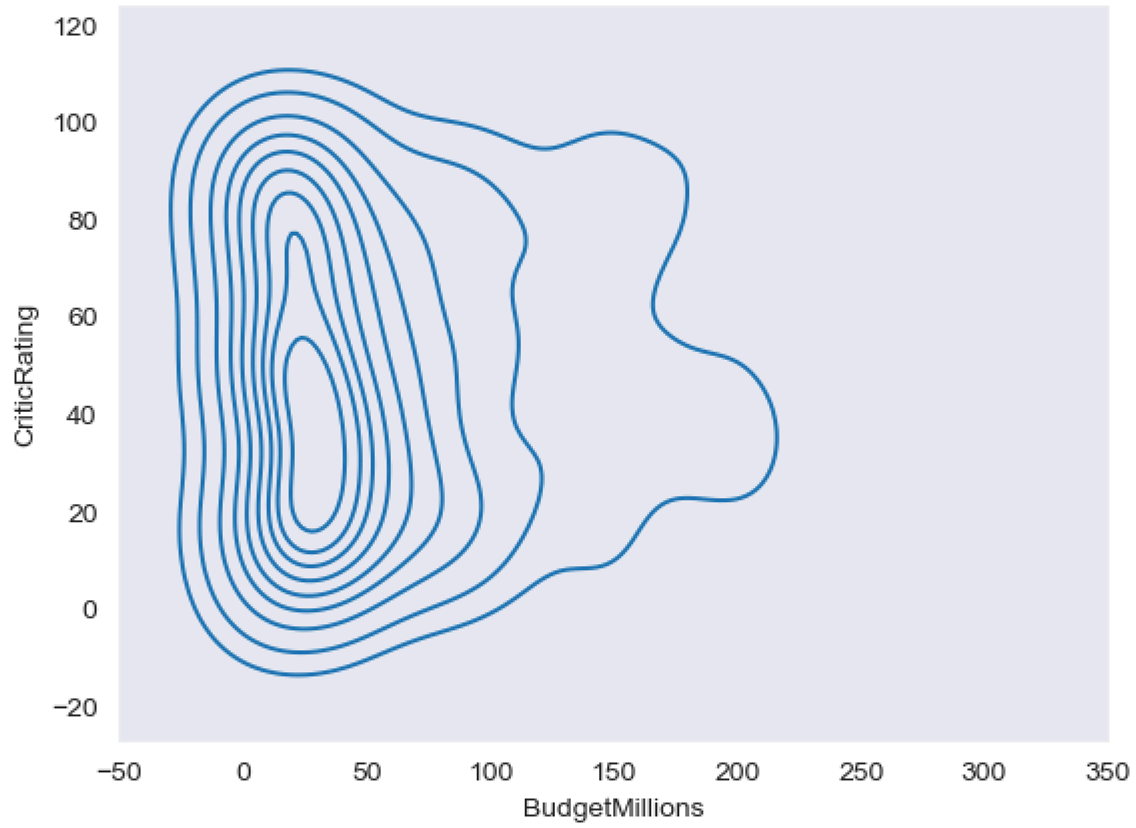


In [207…

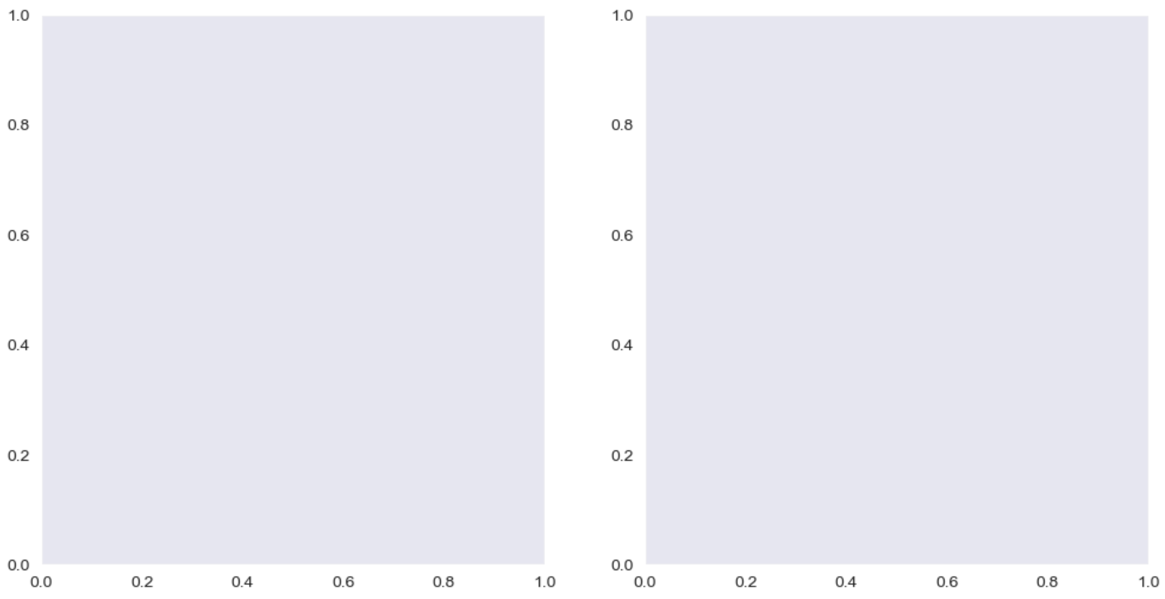
```
sns.set_style('dark')
k1 = sns.kdeplot(x = movie.BudgetMillions,y = movie.AudienceRating)
```



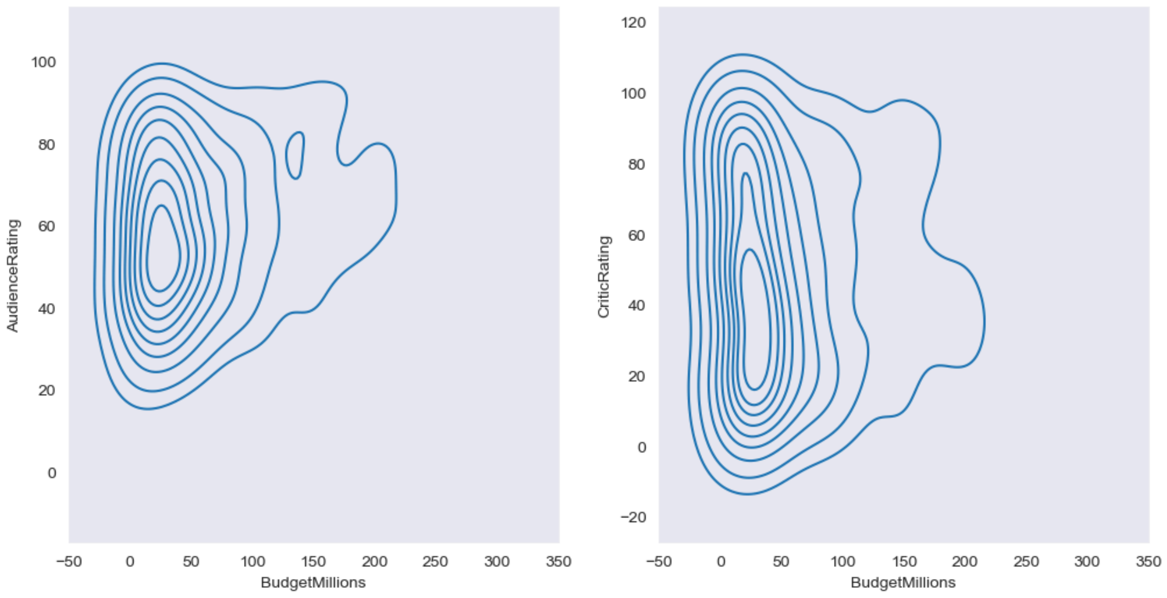
In [211…
```
k2 = sns.kdeplot(x = movie.BudgetMillions,y = movie.CriticRating)
```



In [213…
```
#subplots

f, ax = plt.subplots(1,2, figsize =(12,6))
#f, ax = plt.subplots(3,3, figsize =(12,6))
```



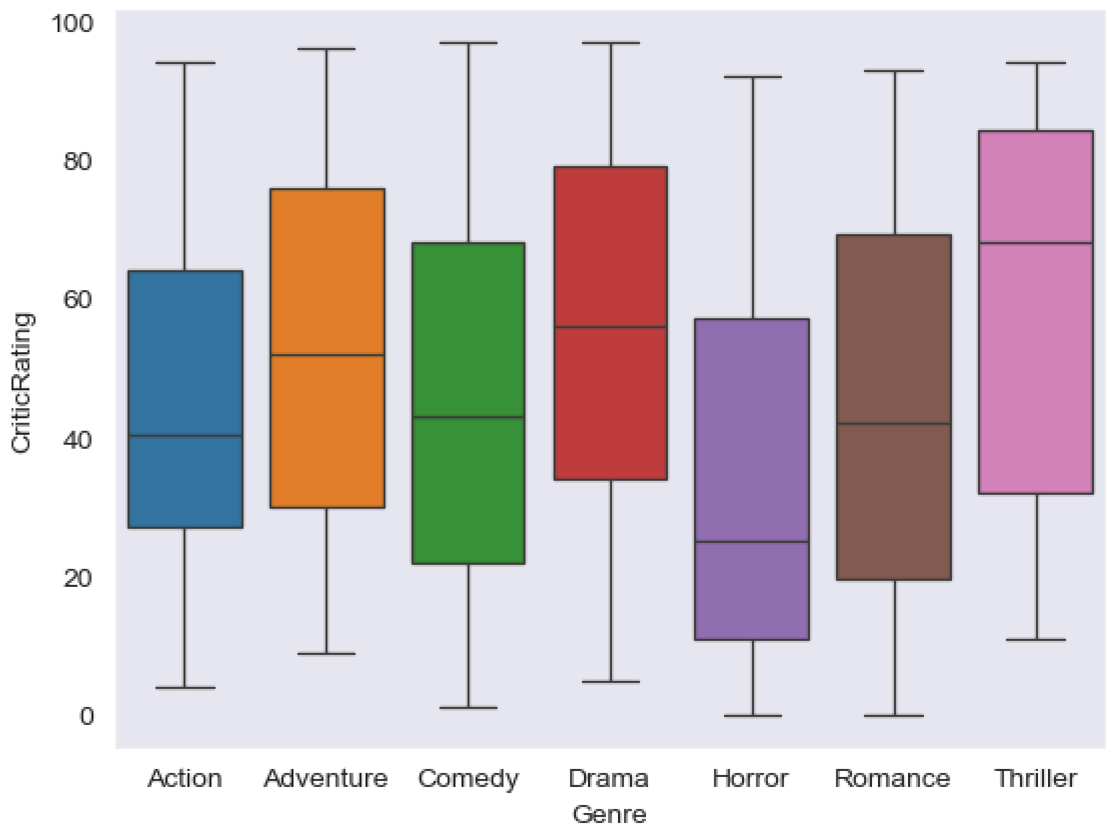
In [217…
```
f, axes = plt.subplots(1,2, figsize =(12,6))

k1 = sns.kdeplot(x = movie.BudgetMillions,y = movie.AudienceRating,ax=axes[0])
k2 = sns.kdeplot(x = movie.BudgetMillions,y = movie.CriticRating,ax = axes[1])
```



In [219…
```
axes
```

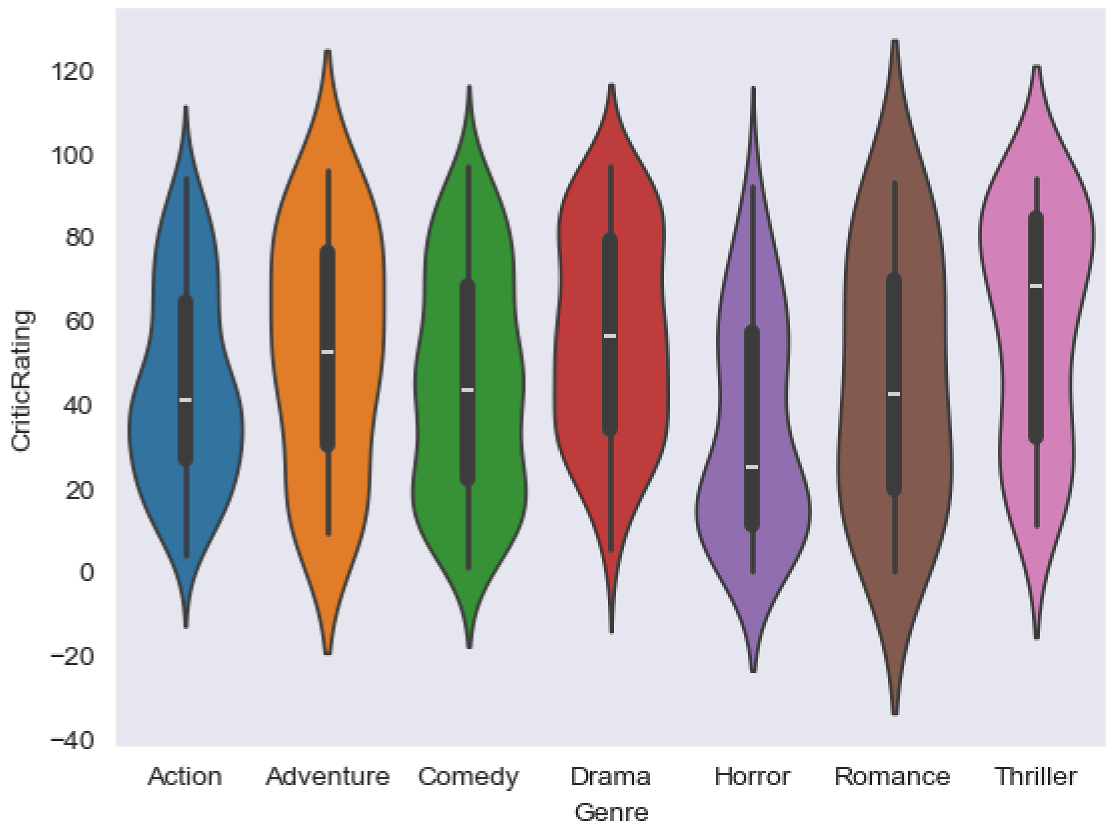
Out[219…
```
array([<Axes: xlabel='BudgetMillions', ylabel='AudienceRating'>,
       <Axes: xlabel='BudgetMillions', ylabel='CriticRating'>],
      dtype=object)
```

In [223…
```
#Box plots -

w = sns.boxplot(data=movie, x='Genre', y = 'CriticRating', hue = 'Genre')
```



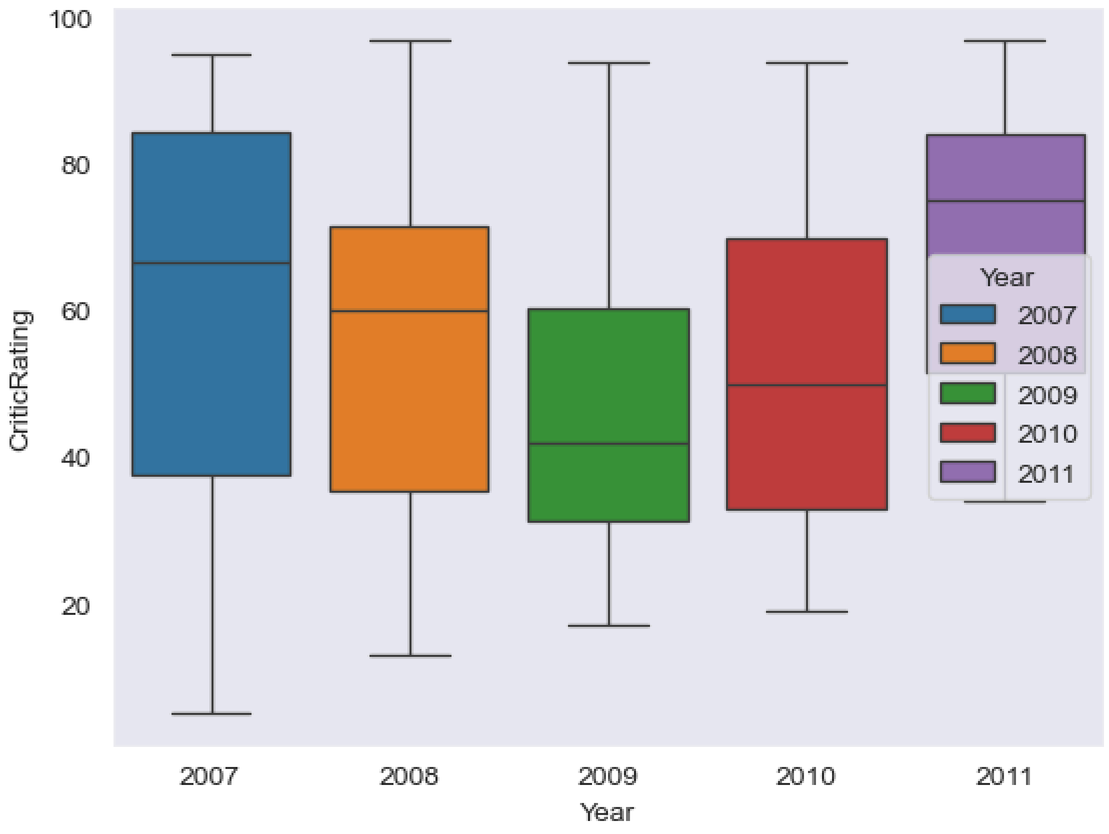
In [225…

```
#violin plot

z = sns.violinplot(data=movie, x='Genre', y = 'CriticRating', hue = 'Genre')
```



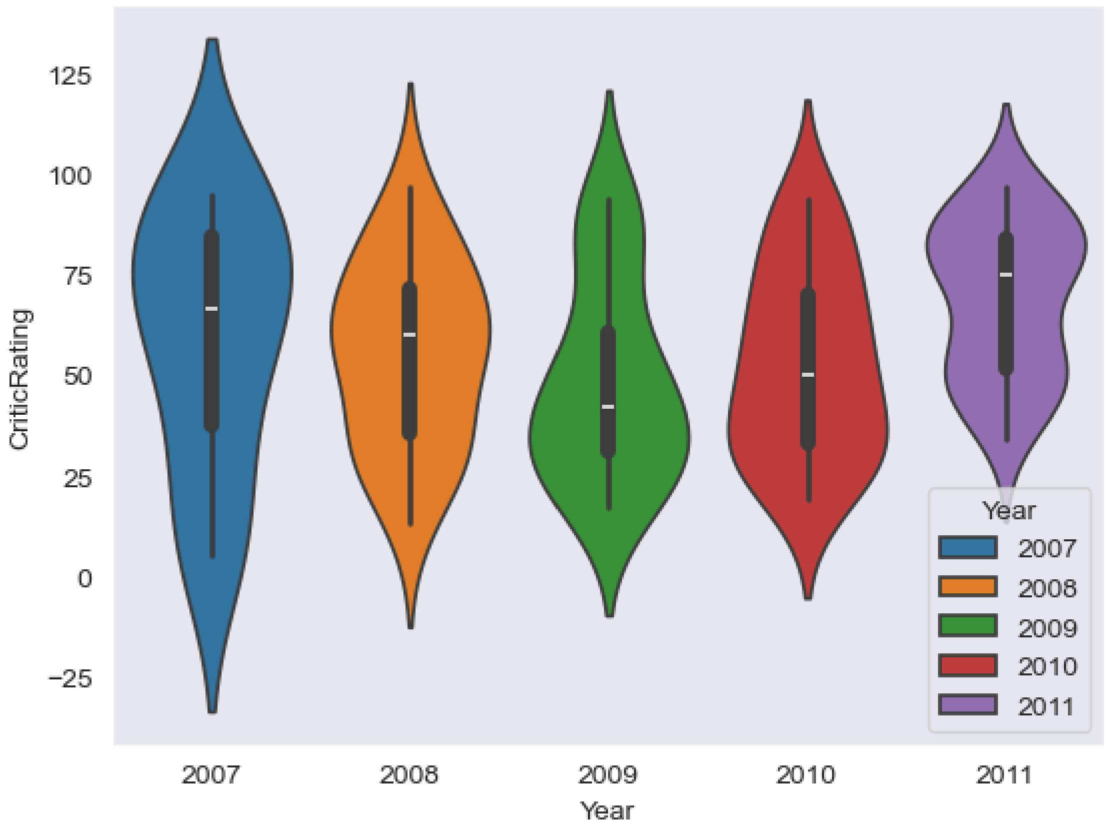
In [235…

```
w1 = sns.boxplot(data=movie[movie.Genre == 'Drama'], x='Year', y = 'CriticRating
```



In [237…

```
z = sns.violinplot(data=movie[movie.Genre == 'Drama'], x='Year', y = 'CriticRati
```



In [ ]:

```
# Createing a Facet grid
```

In [239…

```
g =sns.FacetGrid (movie, row = 'Genre', col = 'Year', hue = 'Genre') #kind of su
```

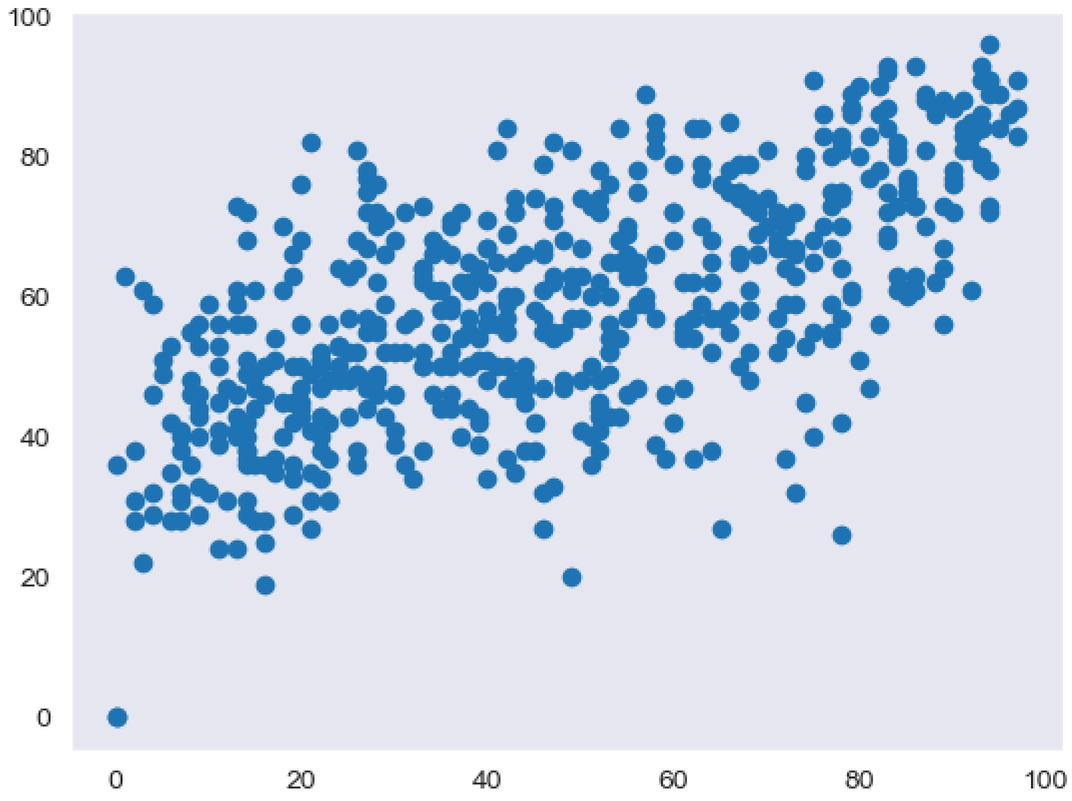
Genre = Action | Year = 2007
Genre = Action | Year = 2008
Genre = Action | Year = 2009
Genre = Action | Year = 2010
Genre = Action | Year = 2011
Genre = Adventure | Year = 2007
Genre = Adventure | Year = 2008
Genre = Adventure | Year = 2009
Genre = Adventure | Year = 2010
Genre = Adventure | Year = 2011
Genre = Comedy | Year = 2007
Genre = Comedy | Year = 2008
Genre = Comedy | Year = 2009
Genre = Comedy | Year = 2010
Genre = Comedy | Year = 2011
Genre = Drama | Year = 2007
Genre = Drama | Year = 2008
Genre = Drama | Year = 2009
Genre = Drama | Year = 2010
Genre = Drama | Year = 2011
Genre = Horror | Year = 2007
Genre = Horror | Year = 2008
Genre = Horror | Year = 2009
Genre = Horror | Year = 2010
Genre = Horror | Year = 2011
Genre = Romance | Year = 2007
Genre = Romance | Year = 2008
Genre = Romance | Year = 2009
Genre = Romance | Year = 2010
Genre = Romance | Year = 2011
Genre = Thriller | Year = 2007
Genre = Thriller | Year = 2008
Genre = Thriller | Year = 2009
Genre = Thriller | Year = 2010
Genre = Thriller | Year = 2011
0.0
0.2
0.4
0.6
0.8
1.0

In [241…

```
plt.scatter(movie.CriticRating,movie.AudienceRating)
```

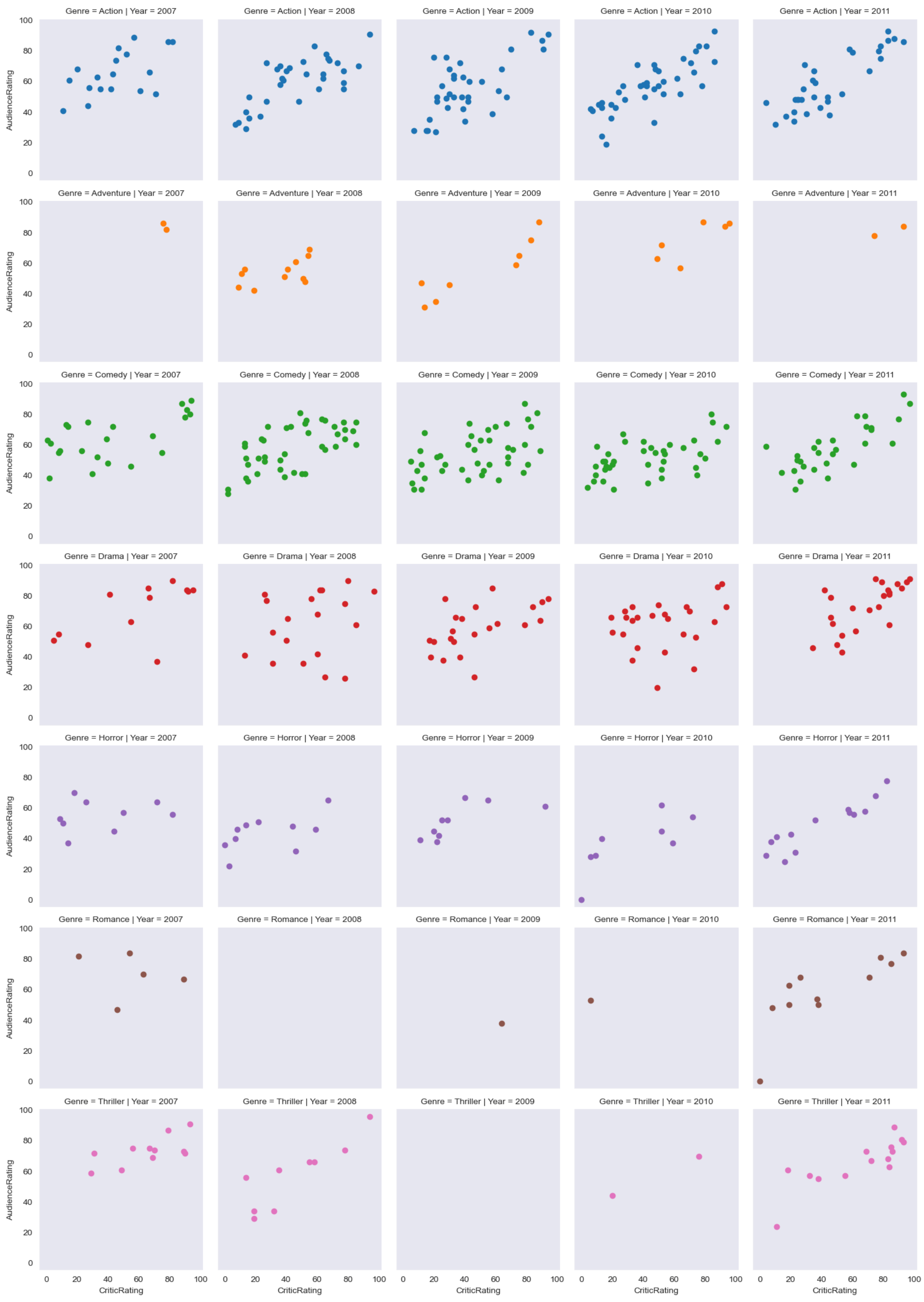
Out[241…

```
<matplotlib.collections.PathCollection at 0x21620a6bcb0>
```



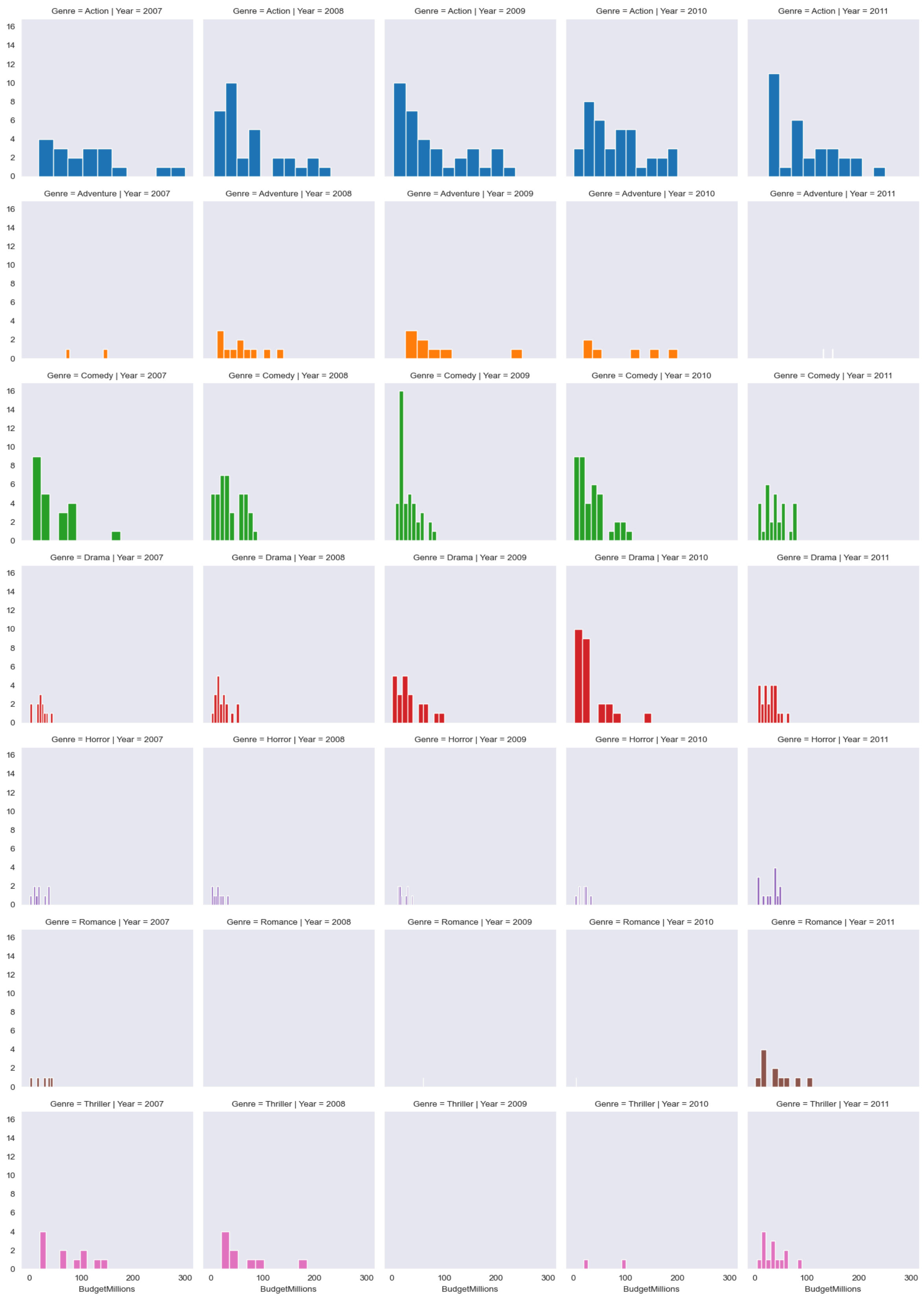
In [243…

```
g =sns.FacetGrid (movie, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.scatter, 'CriticRating', 'AudienceRating' ) #scatterplots are mapp
```



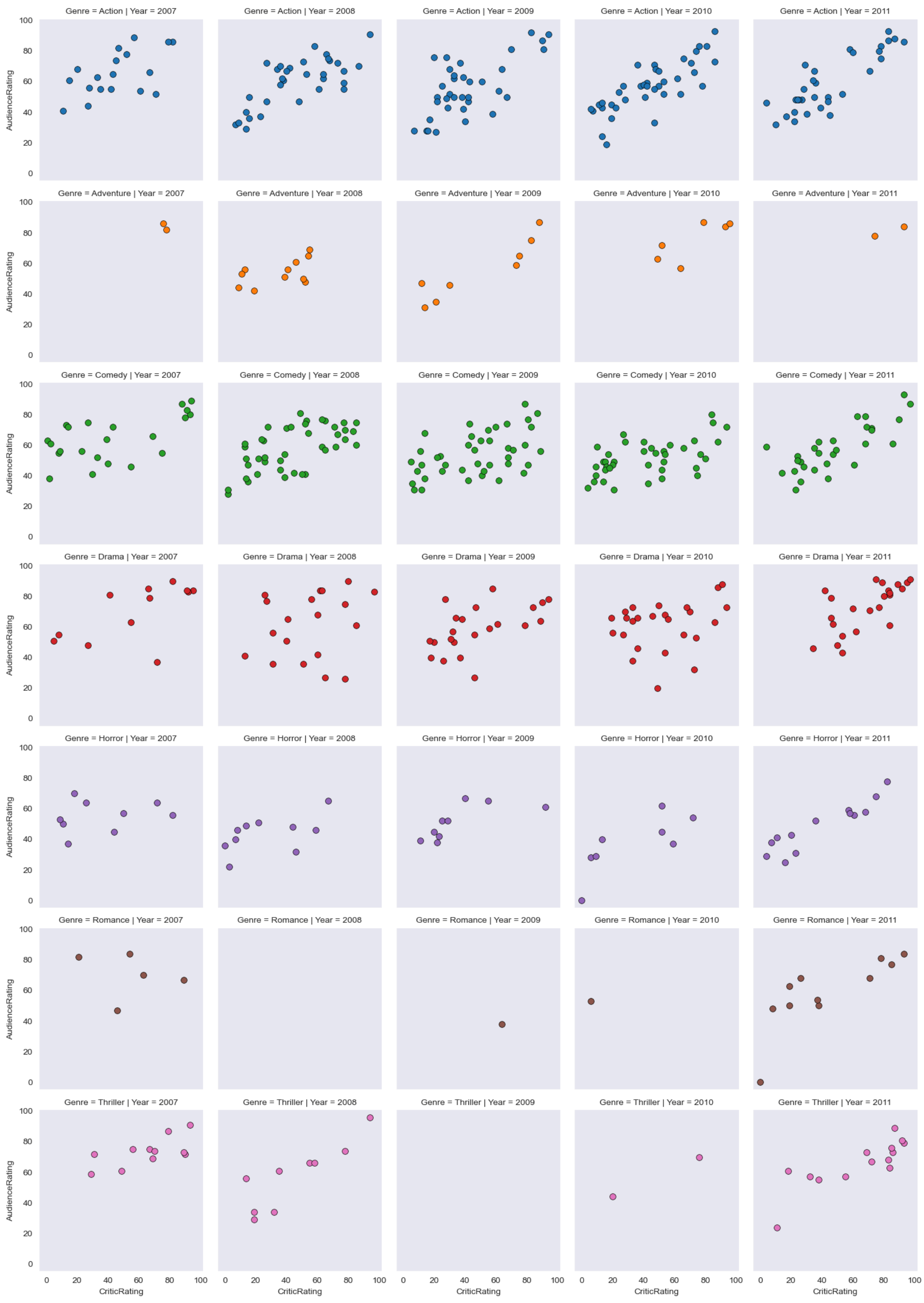
In [245…

```
# you can populated any type of chat.

g =sns.FacetGrid (movie, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.hist, 'BudgetMillions') #scatterplots are mapped in facetgrid
```



In [247…

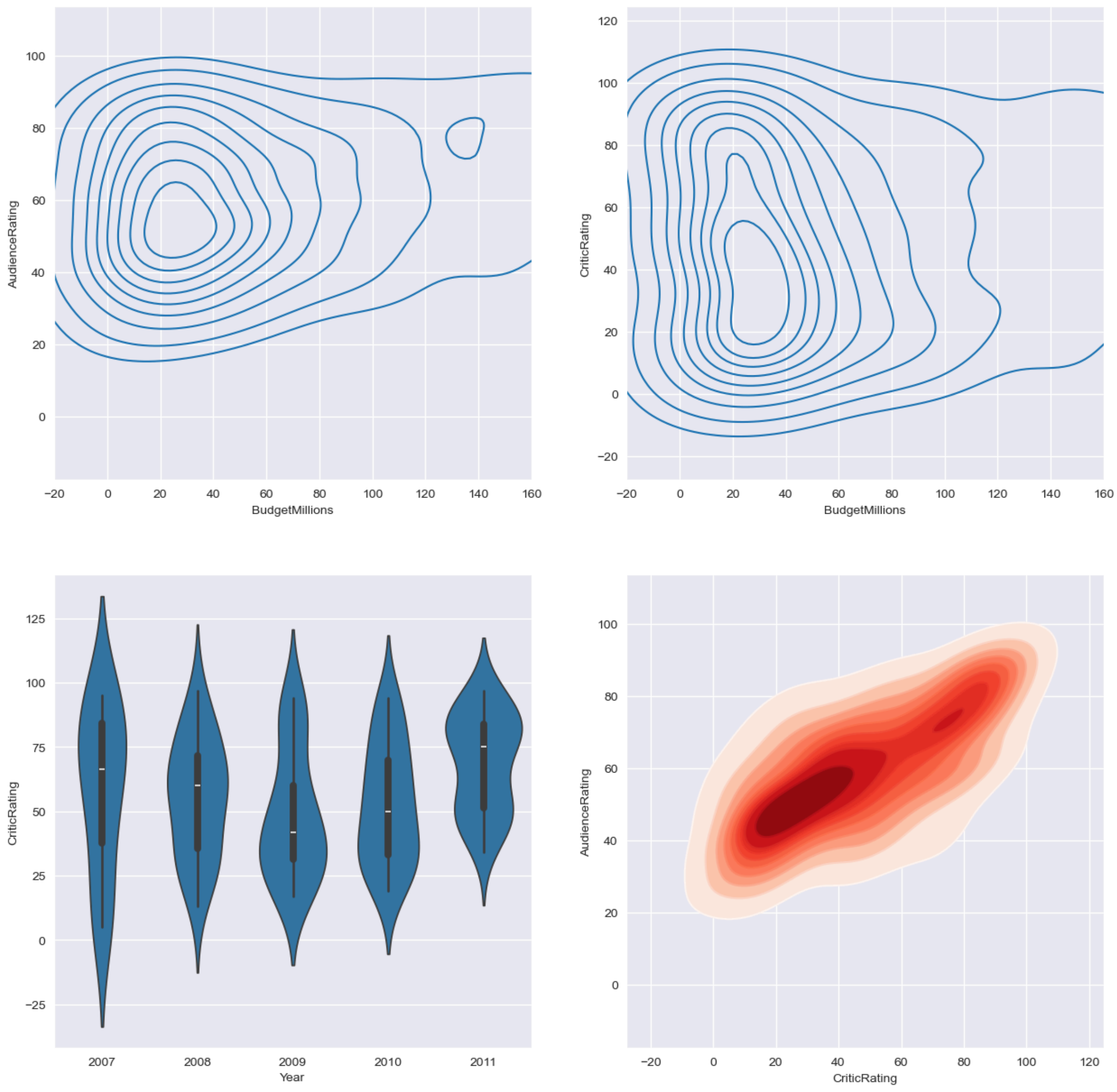
```
#
g =sns.FacetGrid (movie, row = 'Genre', col = 'Year', hue = 'Genre')
kws = dict(s=50, linewidth=0.5,edgecolor='black')
g = g.map(plt.scatter, 'CriticRating', 'AudienceRating',**kws ) #scatterplots ar
```



In [251…

```
# python is not vectorize programming language
# Building dashboards (dashboard - combination of chats)

sns.set_style('darkgrid')
f, axes = plt.subplots (2,2, figsize = (15,15))

k1 = sns.kdeplot(x = movie.BudgetMillions,y = movie.AudienceRating,ax=axes[0,0])
k2 = sns.kdeplot(x = movie.BudgetMillions,y = movie.CriticRating,ax = axes[0,1])

k1.set(xlim=(-20,160))
```

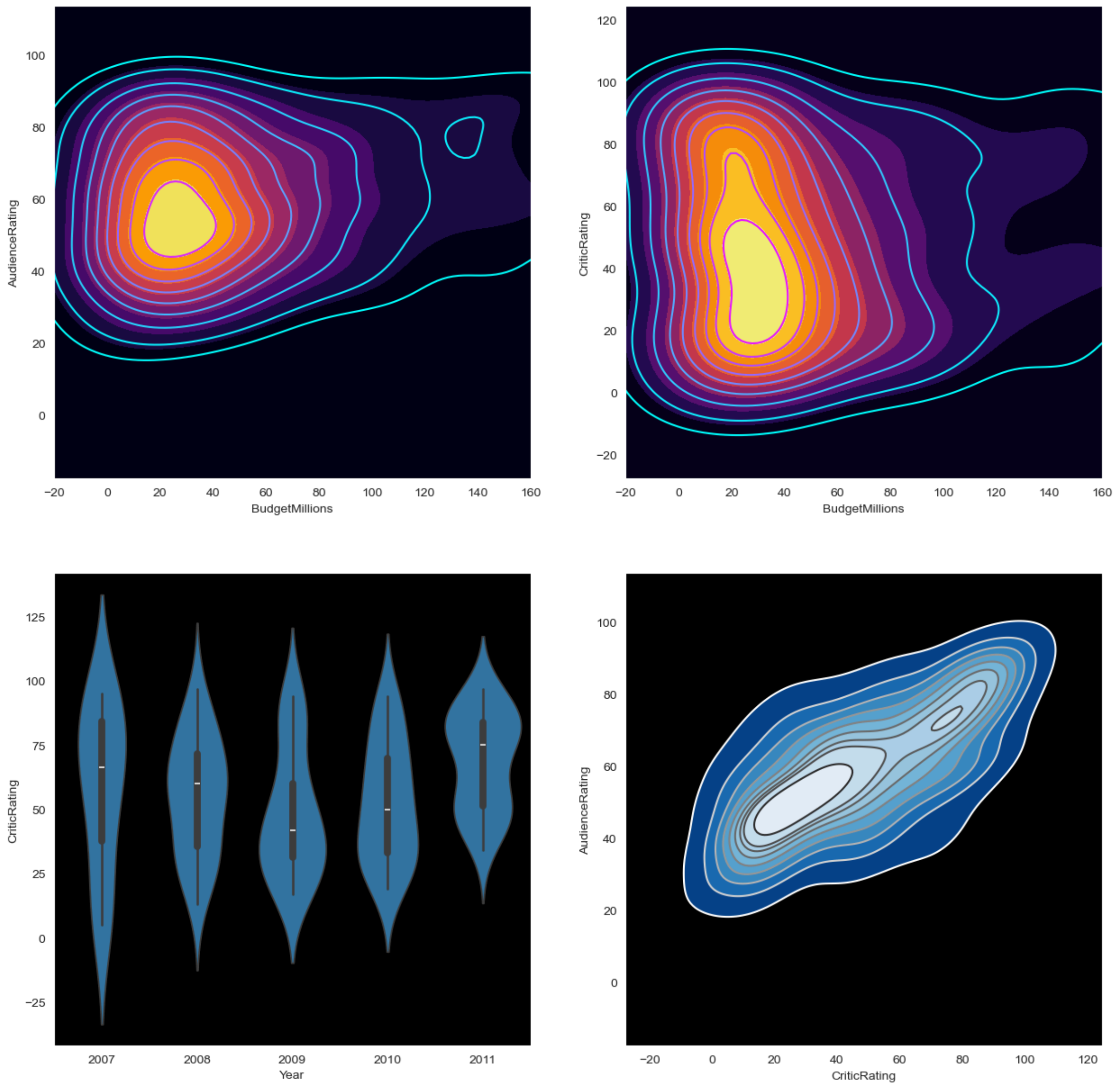
```
k2.set(xlim=(-20,160))

z = sns.violinplot(data=movie[movie.Genre=='Drama'], x='Year', y = 'CriticRating

k4 = sns.kdeplot(x = movie.CriticRating,y = movie.AudienceRating,shade = True,sh

k4b = sns.kdeplot(x = movie.CriticRating, y = movie.AudienceRating,cmap='Reds',a

plt.show()
```



In [261…

```
# How can you style your dashboard  using different color map

# python is not vectorize programming language
# Building dashboards (dashboard - combination of chats)

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

#plot [0,0]
k1 = sns.kdeplot(x = movie.BudgetMillions,y = movie.AudienceRating, \
                 shade = True, shade_lowest=True,cmap = 'inferno', \
                 ax = axes[0,0])
k1b = sns.kdeplot(x = movie.BudgetMillions, y = movie.AudienceRating, \
                 cmap = 'cool',ax = axes[0,0])
```

```
#plot [0,1]
k2 = sns.kdeplot(x = movie.BudgetMillions,y = movie.CriticRating,\
                 shade=True, shade_lowest=True, cmap='inferno',\
                 ax = axes[0,1])
k2b = sns.kdeplot(x = movie.BudgetMillions,y = movie.CriticRating,\
                  cmap = 'cool', ax = axes[0,1])

#plot[1,0]
z = sns.violinplot(data=movie[movie.Genre=='Drama'], \
                   x='Year', y = 'CriticRating', ax=axes[1,0])

#plot[1,1]
k4 = sns.kdeplot(x = movie.CriticRating, y = movie.AudienceRating, \
                 shade = True,shade_lowest=False,cmap='Blues_r', \
                 ax=axes[1,1])

k4b = sns.kdeplot(x = movie.CriticRating, y = movie.AudienceRating, \
                  cmap='gist_gray_r',ax = axes[1,1])


k1.set(xlim=(-20,160))
k2.set(xlim=(-20,160))

plt.show()
```



```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```