Movie Rating Analysis and Visualization

In [1]:
```
#pip uninstall numpy
```

In [2]:
```
#pip install numpy==1.26.4 --force-reinstall
```

In [3]:
```
#pip install --upgrade --force-reinstall pandas
```

In [4]:
```
import pandas as pd
import numpy as np
```

In [5]:
```
movie = pd.read_csv(r"C:\Users\Affan\OneDrive\Desktop\FSDS Course NIT\Prakash Si
movie
```

Out[5]:

	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 [6]:
```
type(movie)
```

Out[6]:
```
pandas.core.frame.DataFrame
```

In [7]:
```
len(movie)
```

Out[7]:
```
559
```

In [8]:
```
movie.shape
```

Out[8]:
```
(559, 6)
```

In [9]:
```
movie.columns
```

Out[9]:
```
Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
       'Budget (million $)', 'Year of release'],
      dtype='object')
```

In [10]:
```
np.__version__
```

Out[10]:
```
'1.26.4'
```

In [11]:
```
movie.head()
```

Out[11]:

	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 [12]:
```
movie.tail()
```

Out[12]:

	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 [13]:
```
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    object
 1   Genre                      559 non-null    object
 2   Rotten Tomatoes Ratings %  559 non-null    int64 
 3   Audience Ratings %         559 non-null    int64 
 4   Budget (million $)         559 non-null    int64 
 5   Year of release            559 non-null    int64 
dtypes: int64(4), object(2)
memory usage: 26.3+ KB
```

In [14]:
```
movie.describe()
```

Out[14]:

	Rotten Tomatoes Ratings %	Audience Ratings %	Budget (million $)	Year of release
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 [15]:
```
movie.columns=['Film', 'Genre', 'CriticRating', 'AudienceRatings',
       'BudgetInMill', 'Year']
```

In [16]:
```
movie
```

Out[16]:

	Film	Genre	CriticRating	AudienceRatings	BudgetInMill	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 [17]:
```
movie.head(1)
```

Out[17]:

	Film	Genre	CriticRating	AudienceRatings	BudgetInMill	Year
0	(500) Days of Summer	Comedy	87	81	8	2009

what astype does---converts object(default dtype) to userdefined dtype

In [18]:
```
movie.Film=movie.Film.astype('category')
movie.Genre=movie.Genre.astype('category')
movie.Year=movie.Year.astype('category')
movie
```

Out[18]:

	Film	Genre	CriticRating	AudienceRatings	BudgetInMill	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 [19]:
```
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   AudienceRatings  559 non-null    int64   
 4   BudgetInMill     559 non-null    int64   
 5   Year             559 non-null    category
dtypes: category(3), int64(3)
memory usage: 36.5 KB
```

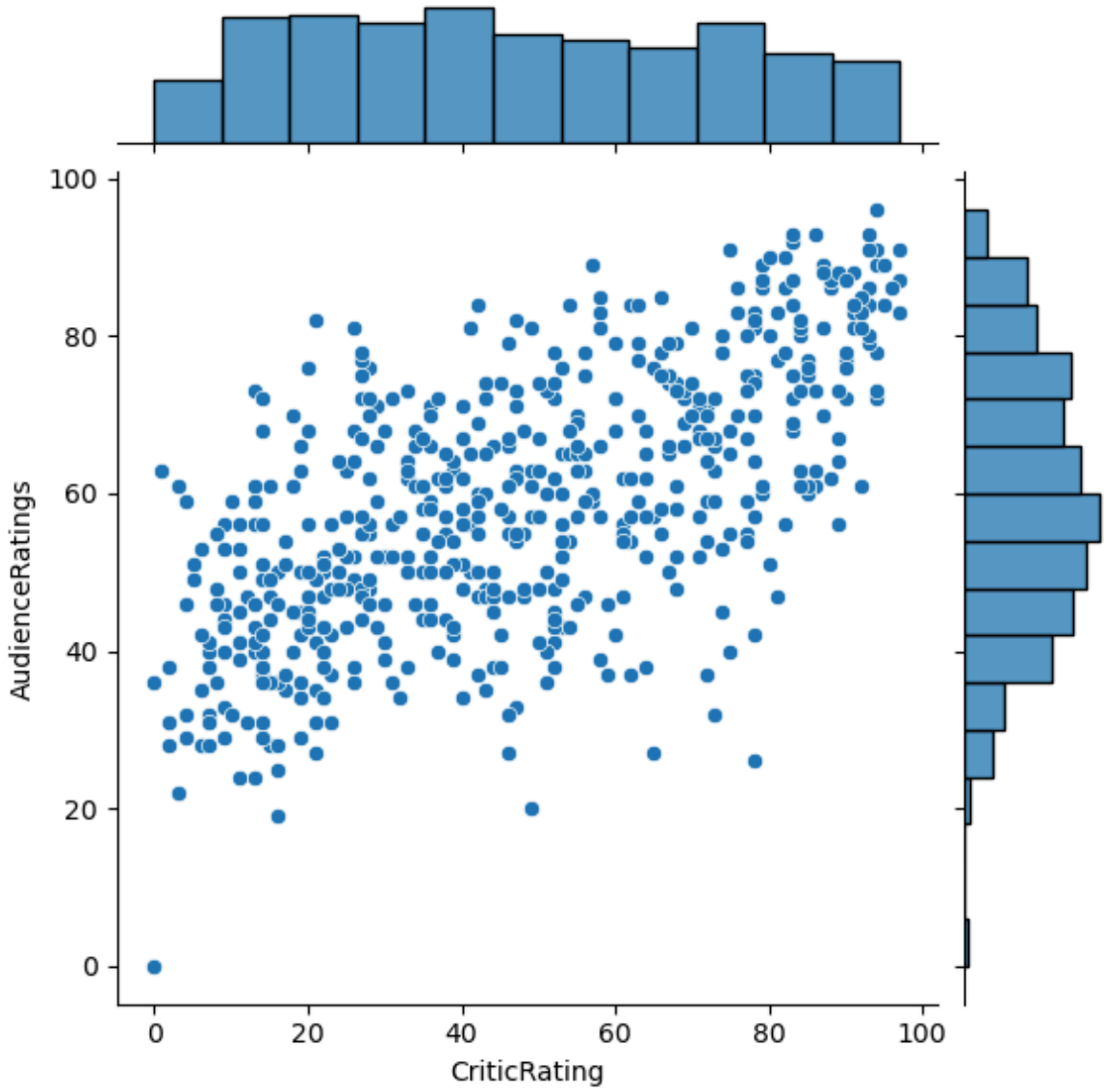
In [20]:
```
movie.Genre.unique()
```

Out[20]:
```
['Comedy', 'Adventure', 'Action', 'Horror', 'Drama', 'Romance', 'Thriller']
Categories (7, object): ['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'R
omance', 'Thriller']
```

In [21]:
```
from matplotlib import pyplot as plt #visualization
import seaborn as sns #advance visualization-

import warnings
warnings.filterwarnings('ignore')
```

In [22]:
```
j = sns.jointplot(data=movie,x='CriticRating',y='AudienceRatings',kind='scatter'
j
```

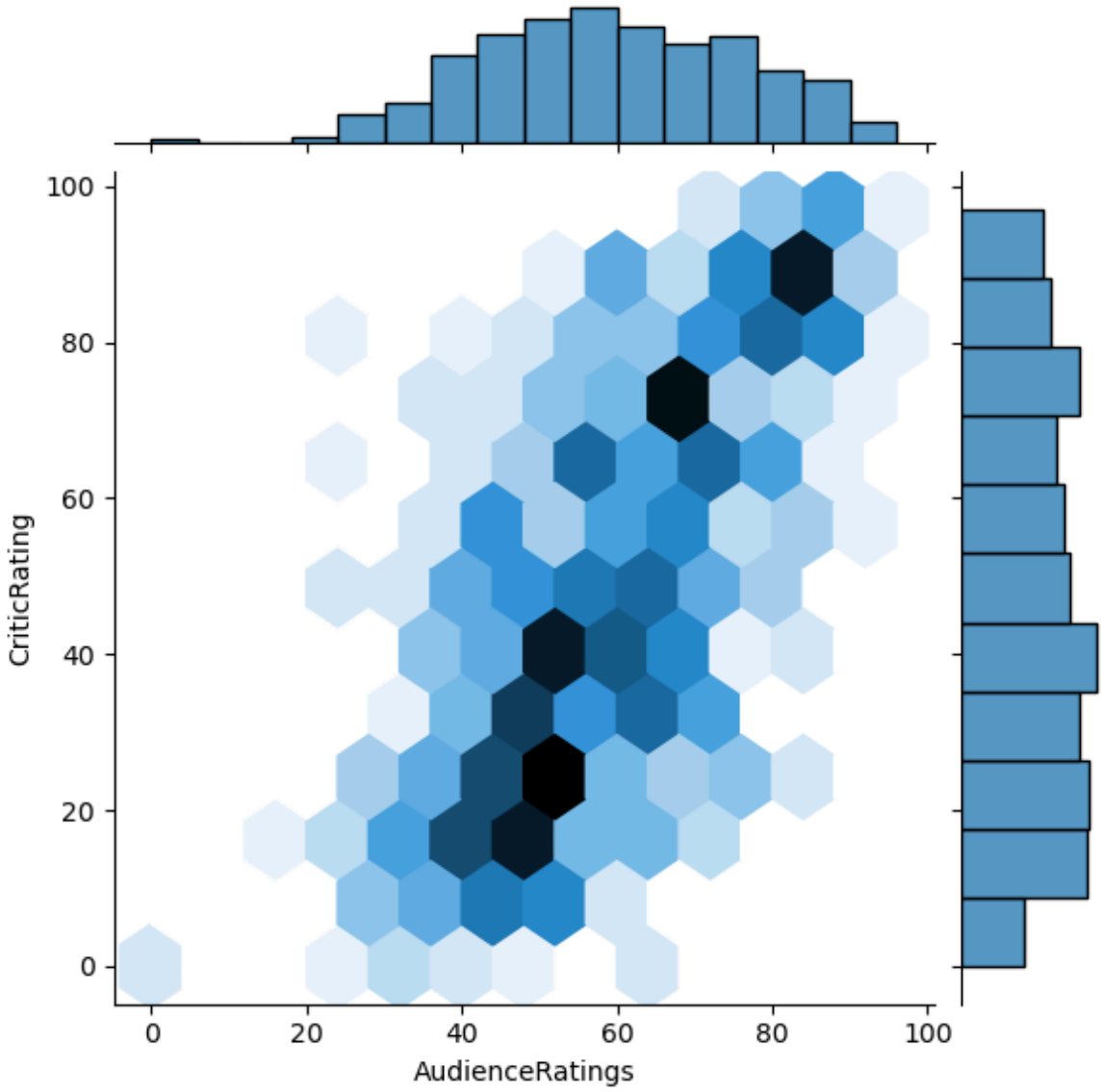
Out[22]:
```
<seaborn.axisgrid.JointGrid at 0x186ffe286e0>
```



'0' is outlier, positive correlation analysis

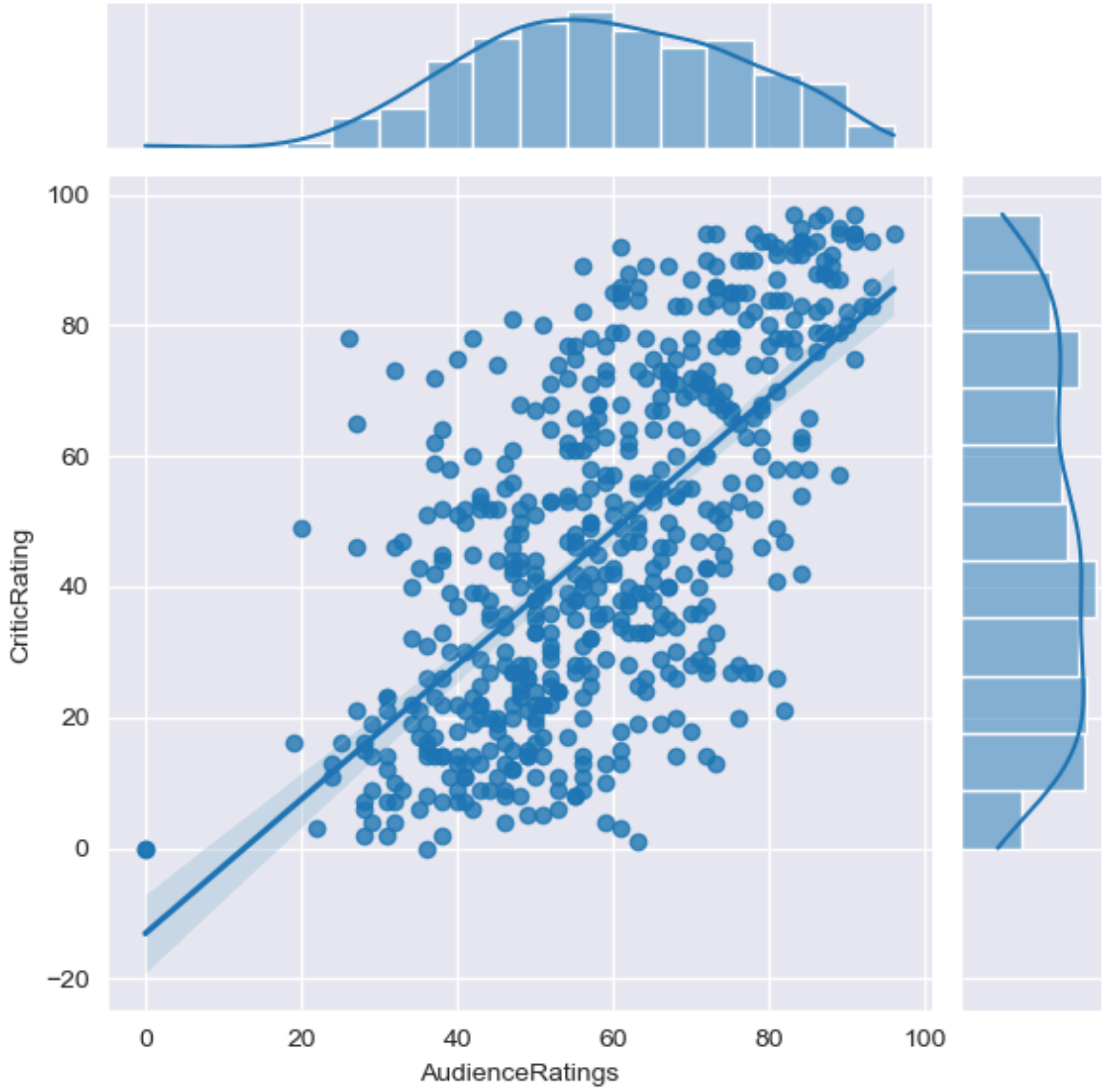
In [23]:
```
d = sns.jointplot(data=movie,x='AudienceRatings',y='CriticRating',kind='hex')
d
```

Out[23]:
```
<seaborn.axisgrid.JointGrid at 0x18699a963f0>
```



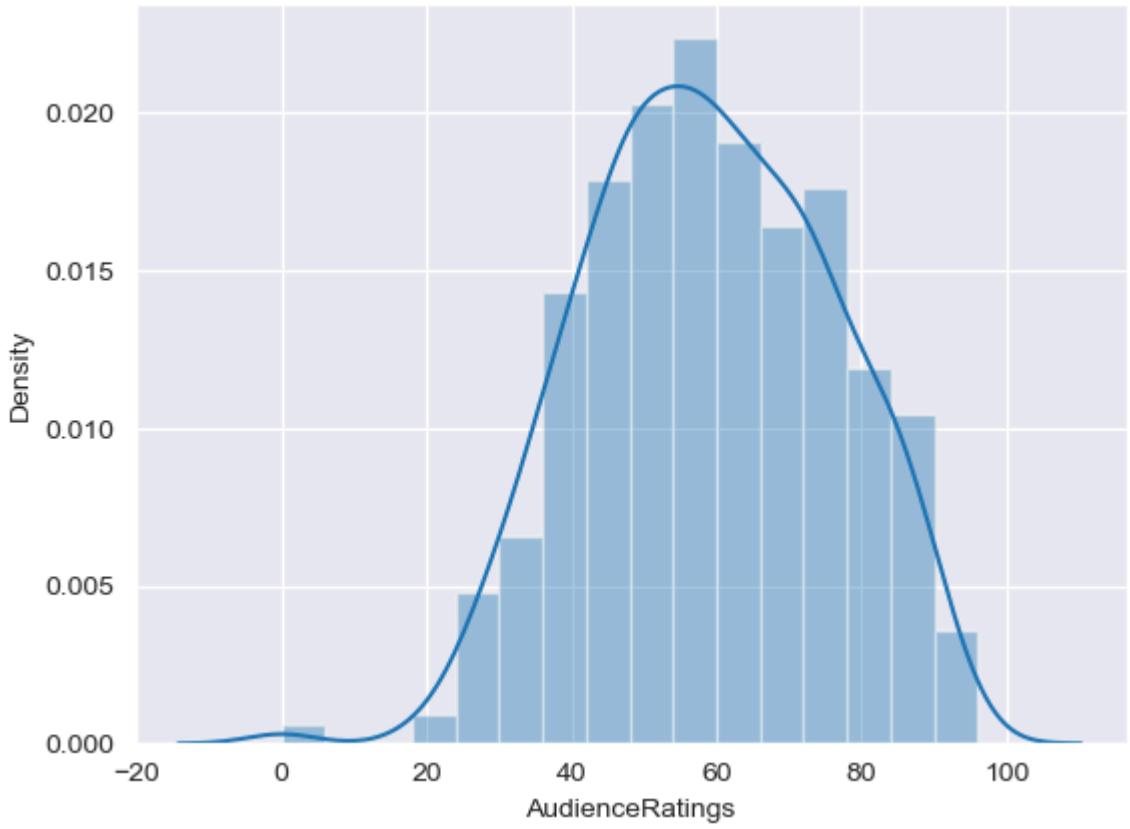
In [24]:
```
sns.set_style('darkgrid')
sns.jointplot(data=movie,x='AudienceRatings',y='CriticRating',kind='reg')
```

Out[24]: <seaborn.axisgrid.JointGrid at 0x1869c07cfb0>



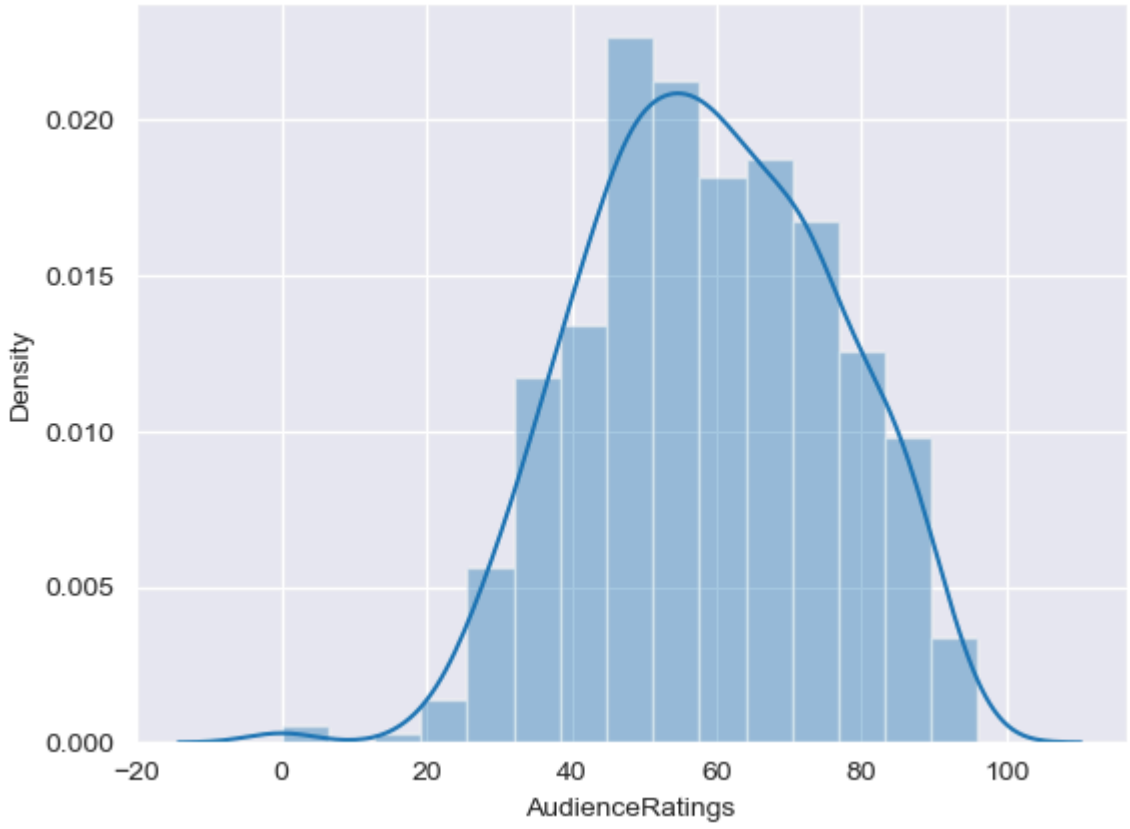
Histogram

In [25]:
```
m1 = sns.distplot(movie.AudienceRatings)
```

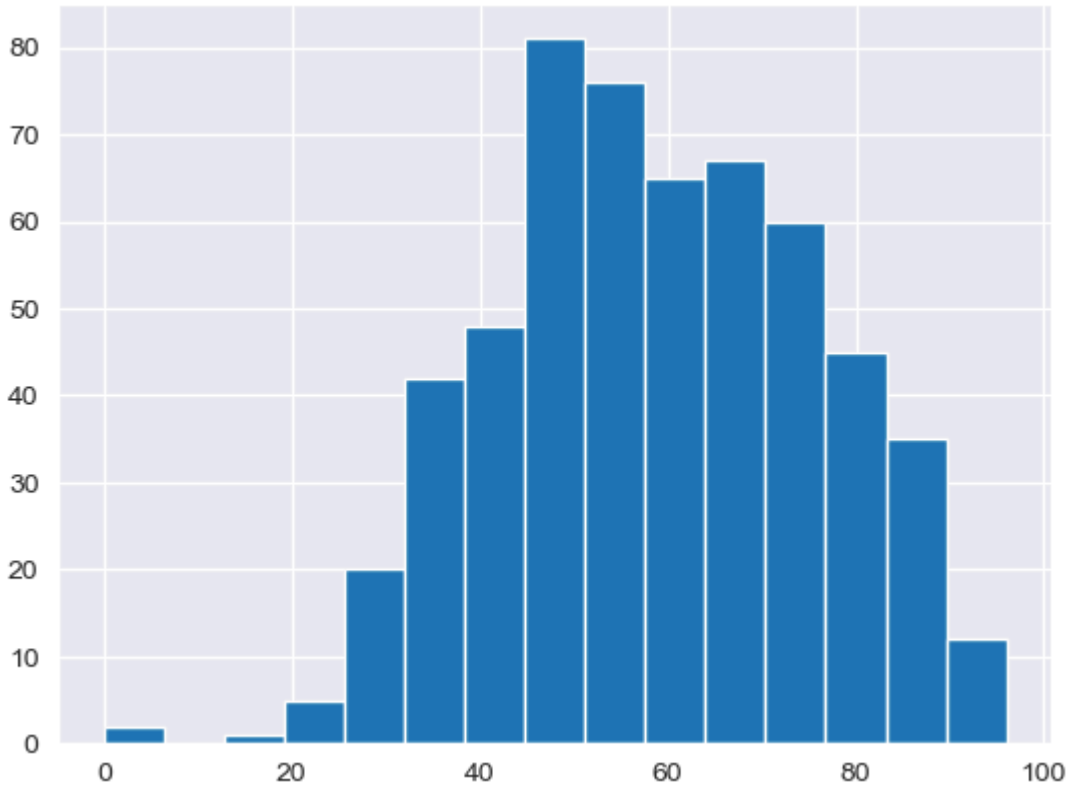


In [26]:
```
sns.set_style('darkgrid')
```

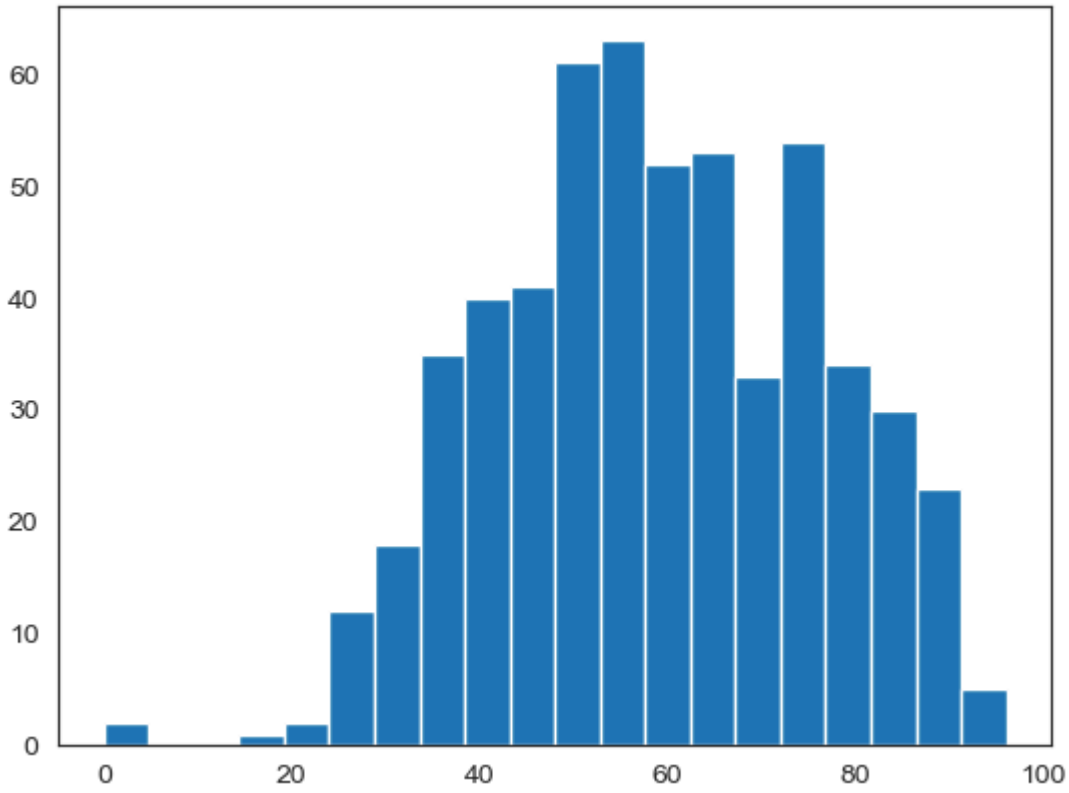
In [27]:
```
m2 = sns.distplot(movie.AudienceRatings,bins=15)
```



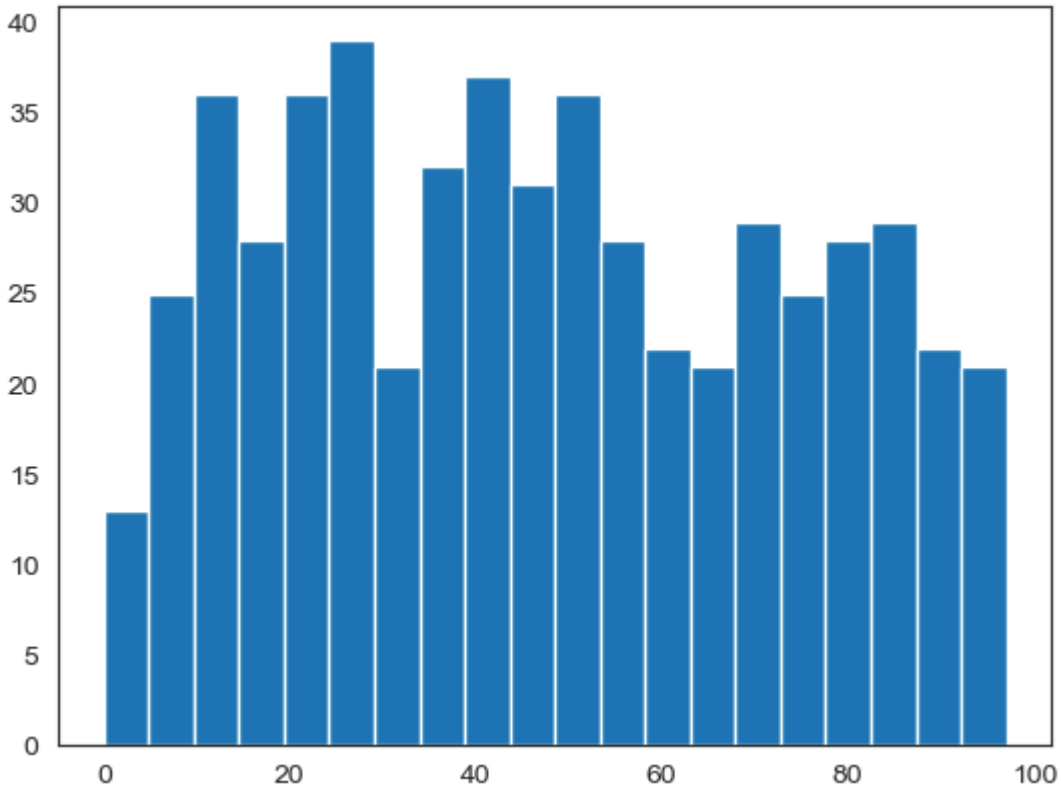
In [28]:
```
n1 =  plt.hist(movie.AudienceRatings,bins=15)
```



In [29]:
```
sns.set_style('white')
n1 = plt.hist(movie.AudienceRatings,bins=20)
```



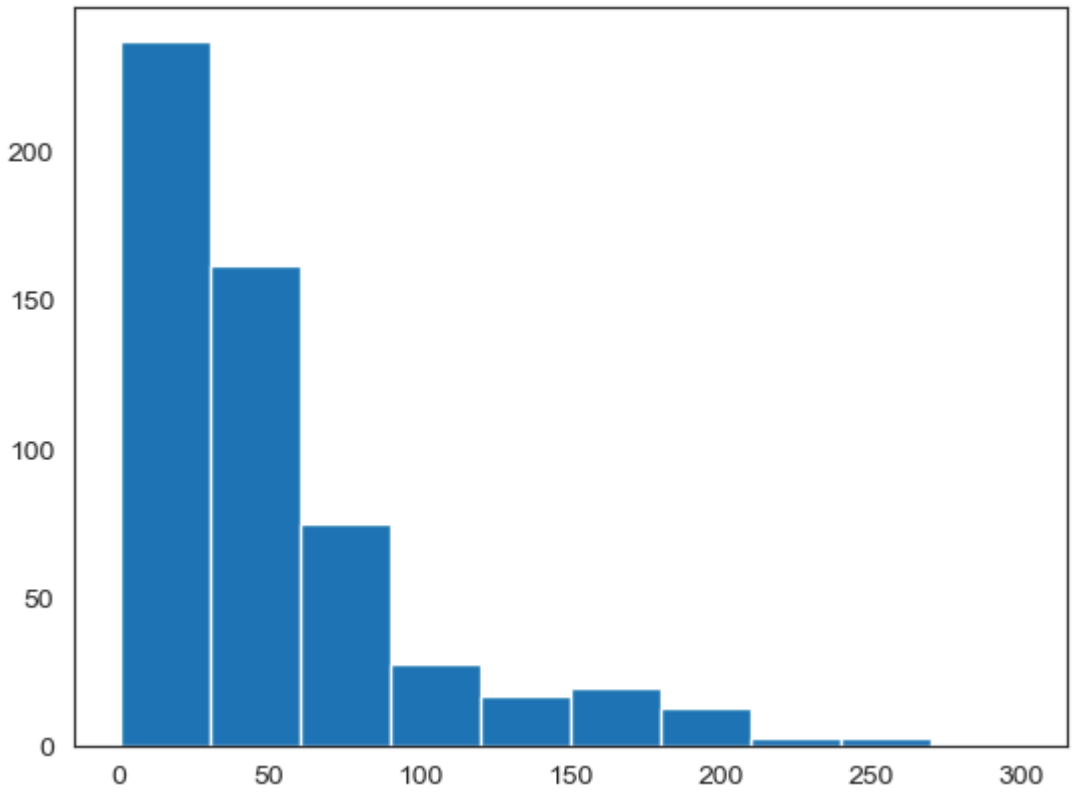
In [30]:
```
n1 = plt.hist(movie.CriticRating,bins=20)
```



Creating stacked histograms

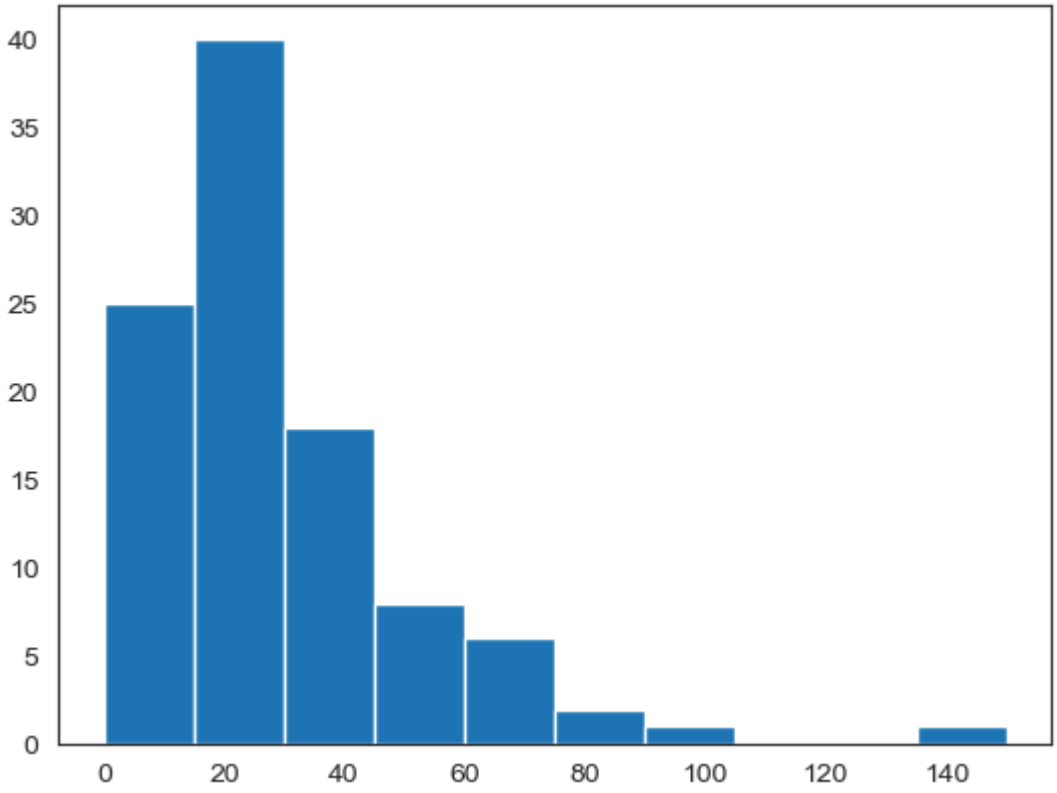
In [31]:
```
plt.hist(movie.BudgetInMill)
```

Out[31]:
```
(array([237., 162.,  75.,  28.,  17.,  20.,  13.,   3.,   3.,   1.]),
 array([  0.,  30.,  60.,  90., 120., 150., 180., 210., 240., 270., 300.]),
 <BarContainer object of 10 artists>)
```



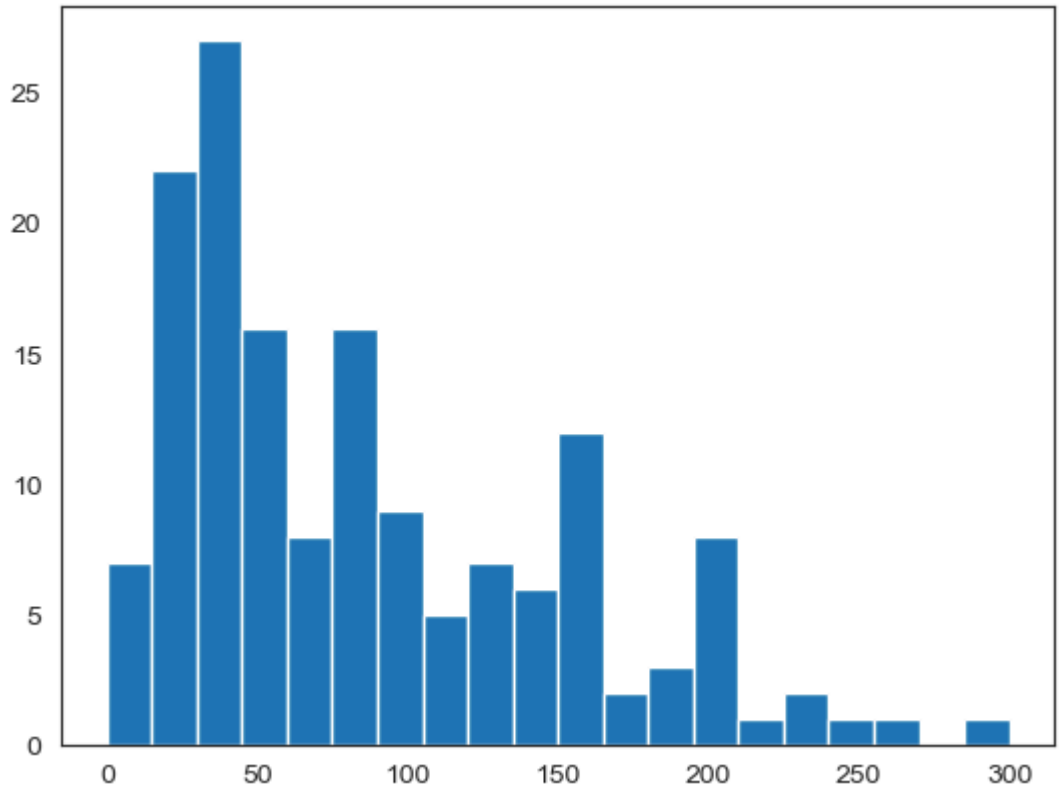
In [32]:
```
plt.hist(movie[movie.Genre == 'Drama'].BudgetInMill)
```

Out[32]:
```
(array([25., 40., 18.,  8.,  6.,  2.,  1.,  0.,  0.,  1.]),
 array([  0.,  15.,  30.,  45.,  60.,  75.,  90., 105., 120., 135., 150.]),
 <BarContainer object of 10 artists>)
```



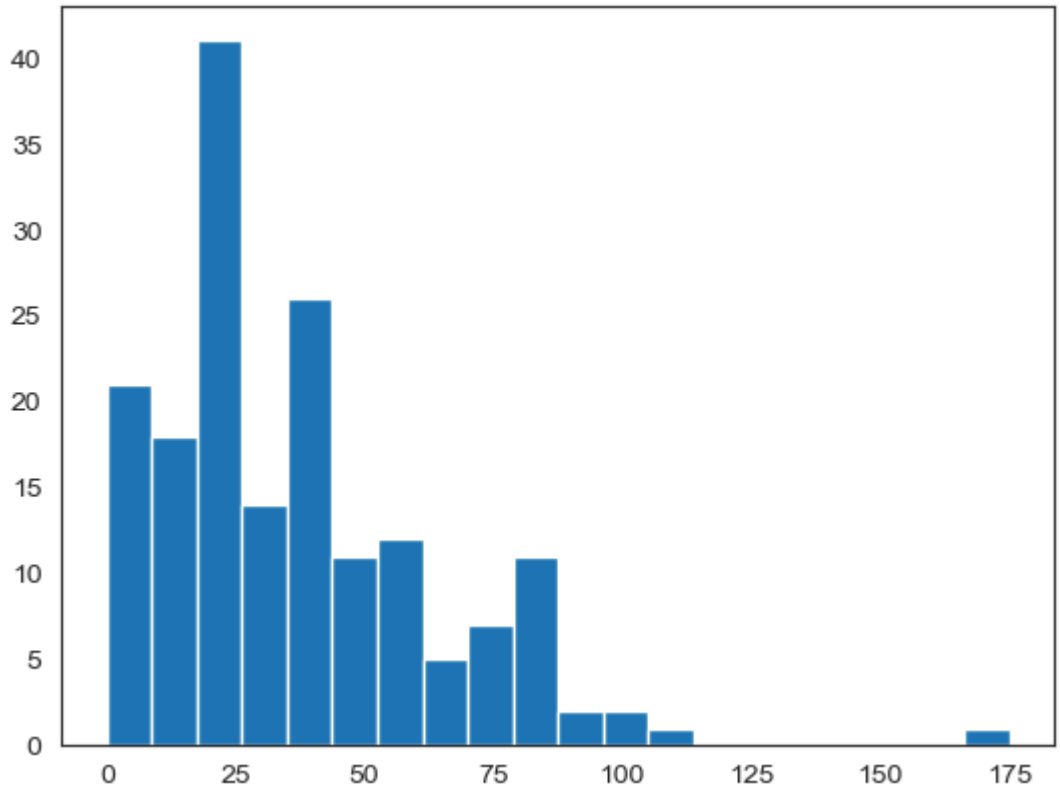
In [33]:
```
plt.hist(movie[movie.Genre=='Action'].BudgetInMill, bins=20)
```

Out[33]:
```
(array([ 7., 22., 27., 16.,  8., 16.,  9.,  5.,  7.,  6., 12.,  2.,  3.,
         8.,  1.,  2.,  1.,  1.,  0.,  1.]),
 array([  0.,  15.,  30.,  45.,  60.,  75.,  90., 105., 120., 135., 150.,
        165., 180., 195., 210., 225., 240., 255., 270., 285., 300.]),
 <BarContainer object of 20 artists>)
```



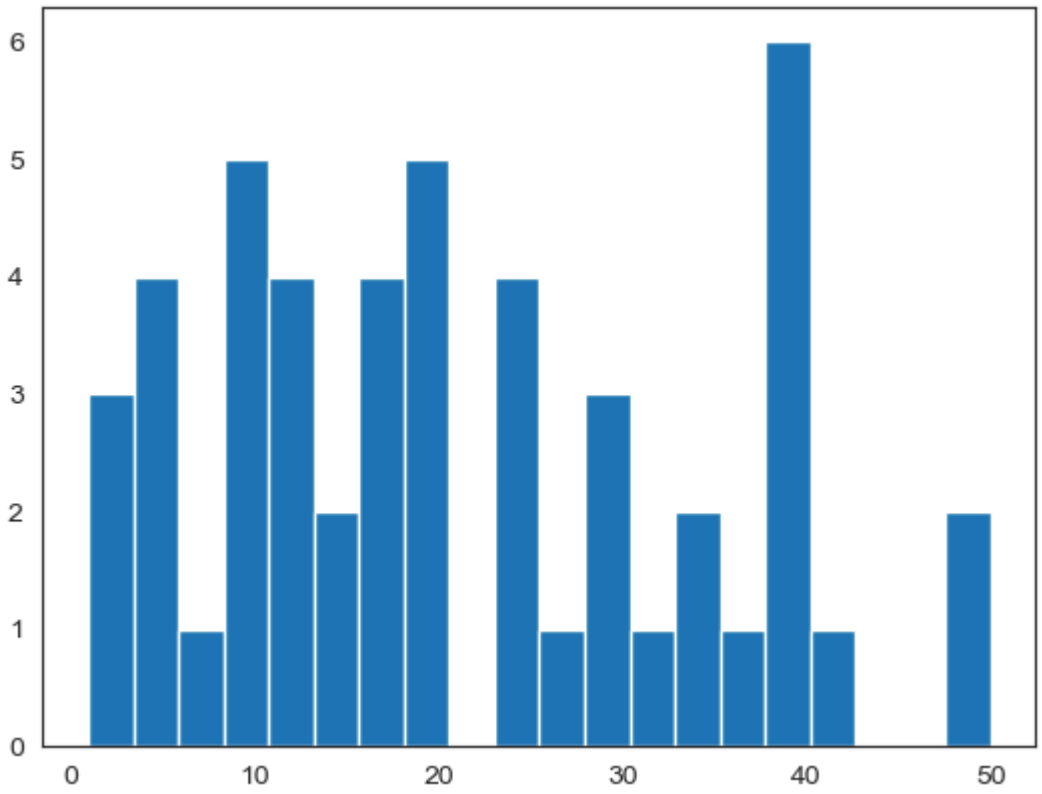
In [34]:
```
plt.hist(movie[movie.Genre=='Comedy'].BudgetInMill, bins=20)
```

Out[34]:
```
(array([21., 18., 41., 14., 26., 11., 12.,  5.,  7., 11.,  2.,  2.,  1.,
         0.,  0.,  0.,  0.,  0.,  0.,  1.]),
 array([  0.  ,   8.75,  17.5 ,  26.25,  35.  ,  43.75,  52.5 ,  61.25,
         70.  ,  78.75,  87.5 ,  96.25, 105.  , 113.75, 122.5 , 131.25,
        140.  , 148.75, 157.5 , 166.25, 175.  ]),
 <BarContainer object of 20 artists>)
```

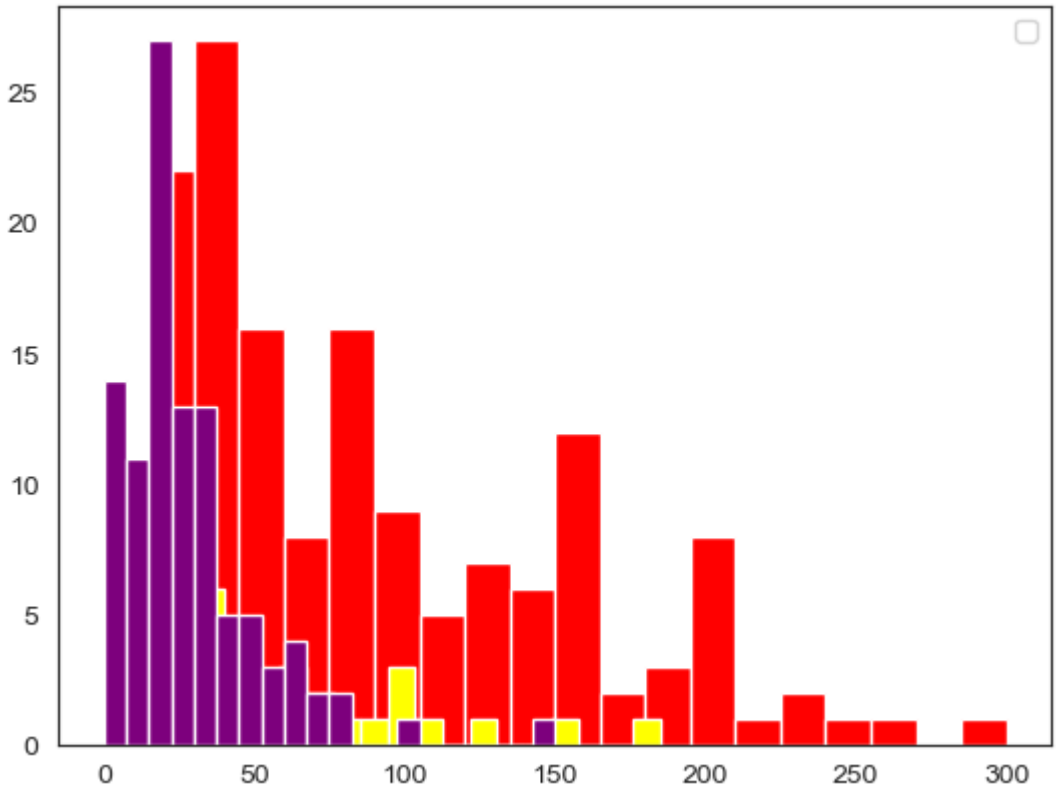


In [35]:
```
plt.hist(movie[movie.Genre=='Horror'].BudgetInMill, bins=20)
```

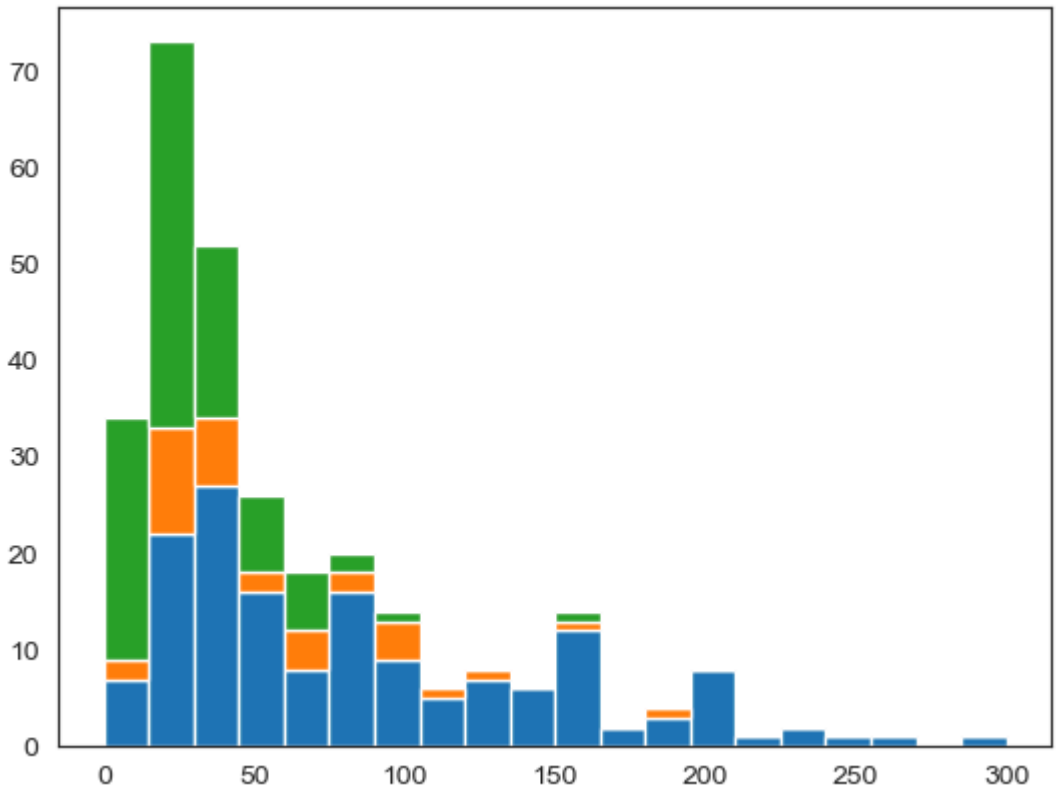
Out[35]:
```
(array([3., 4., 1., 5., 4., 2., 4., 5., 0., 4., 1., 3., 1., 2., 1., 6., 1.,
        0., 0., 2.]),
 array([ 1.  ,  3.45,  5.9 ,  8.35, 10.8 , 13.25, 15.7 , 18.15, 20.6 ,
        23.05, 25.5 , 27.95, 30.4 , 32.85, 35.3 , 37.75, 40.2 , 42.65,
        45.1 , 47.55, 50.  ]),
 <BarContainer object of 20 artists>)
```



In [36]:
```
plt.hist(movie[movie.Genre == 'Action'].BudgetInMill, bins = 20,color='red')
plt.hist(movie[movie.Genre == 'Thriller'].BudgetInMill, bins = 20,color='yellow'
plt.hist(movie[movie.Genre == 'Drama'].BudgetInMill, bins = 20,color='purple')
plt.legend()
plt.show()
```



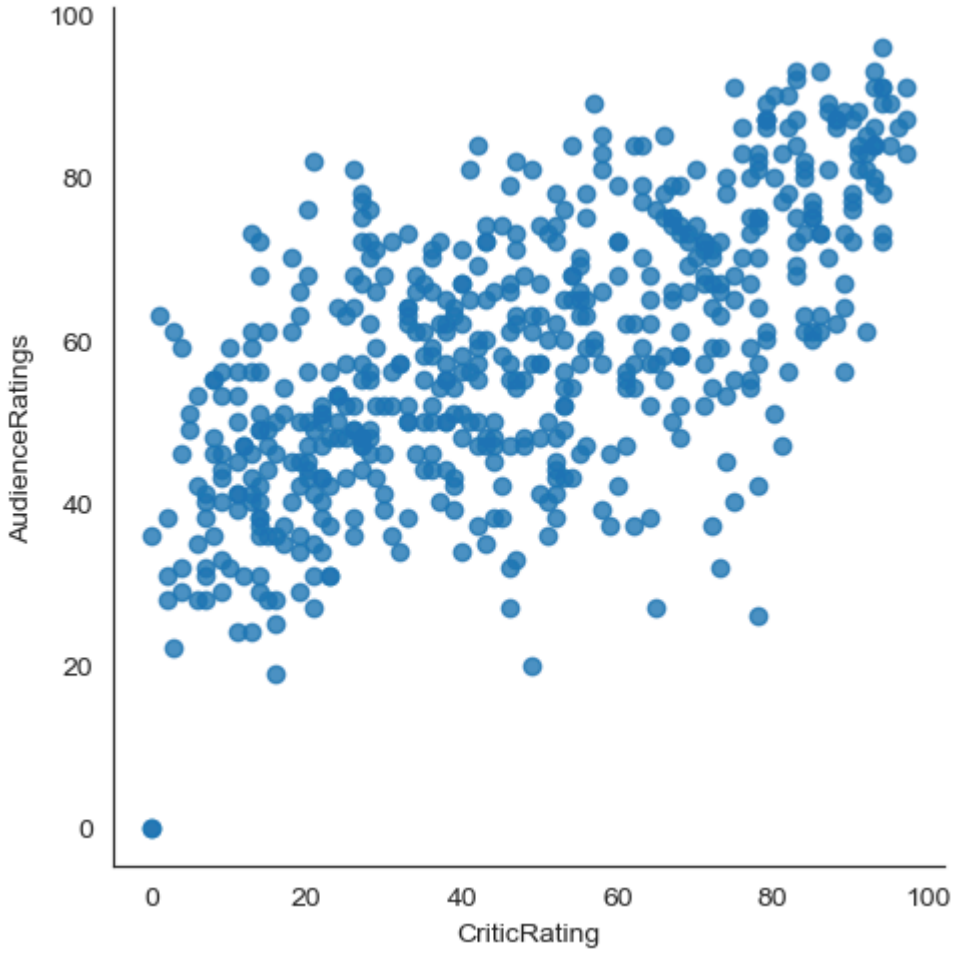
In [37]:
```
plt.hist([movie[movie.Genre == 'Action'].BudgetInMill,\
          movie[movie.Genre == 'Thriller'].BudgetInMill,\
          movie[movie.Genre == 'Drama'].BudgetInMill], bins = 20,stacked=True)
plt.show()
```



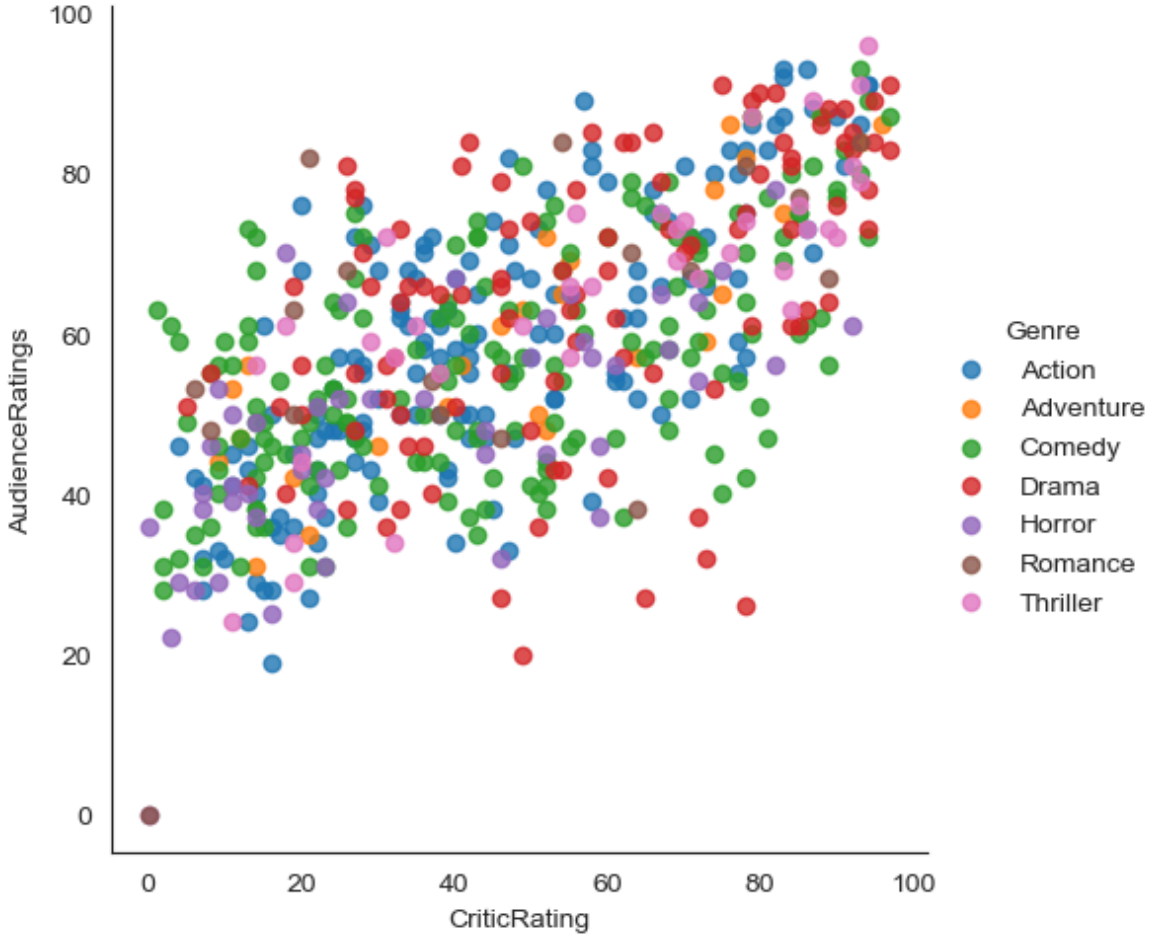
In [38]:
```
# if you have 100 categories you cannot copy & paste all the things
for gen in movie.Genre.cat.categories:
    print(gen)
```

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

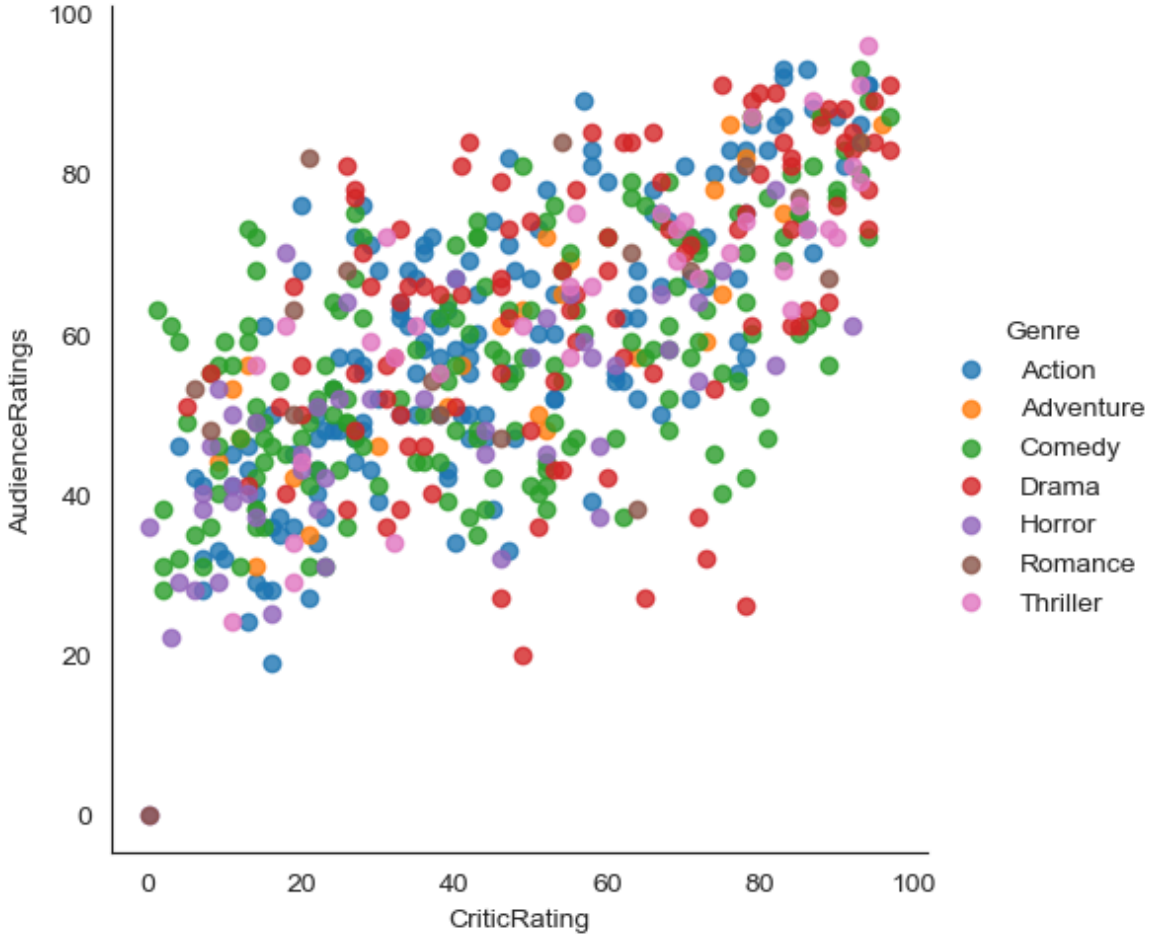
In [39]:
```
vis1 = sns.lmplot(data=movie, x='CriticRating',y='AudienceRatings',fit_reg=False
```



In [40]:
```
vis1 = sns.lmplot(data=movie, x='CriticRating',y='AudienceRatings',fit_reg=False
```



In [41]:
```
vis1 = sns.lmplot(data=movie, x='CriticRating',y='AudienceRatings',\
                  fit_reg=False, hue='Genre',aspect=1)
```

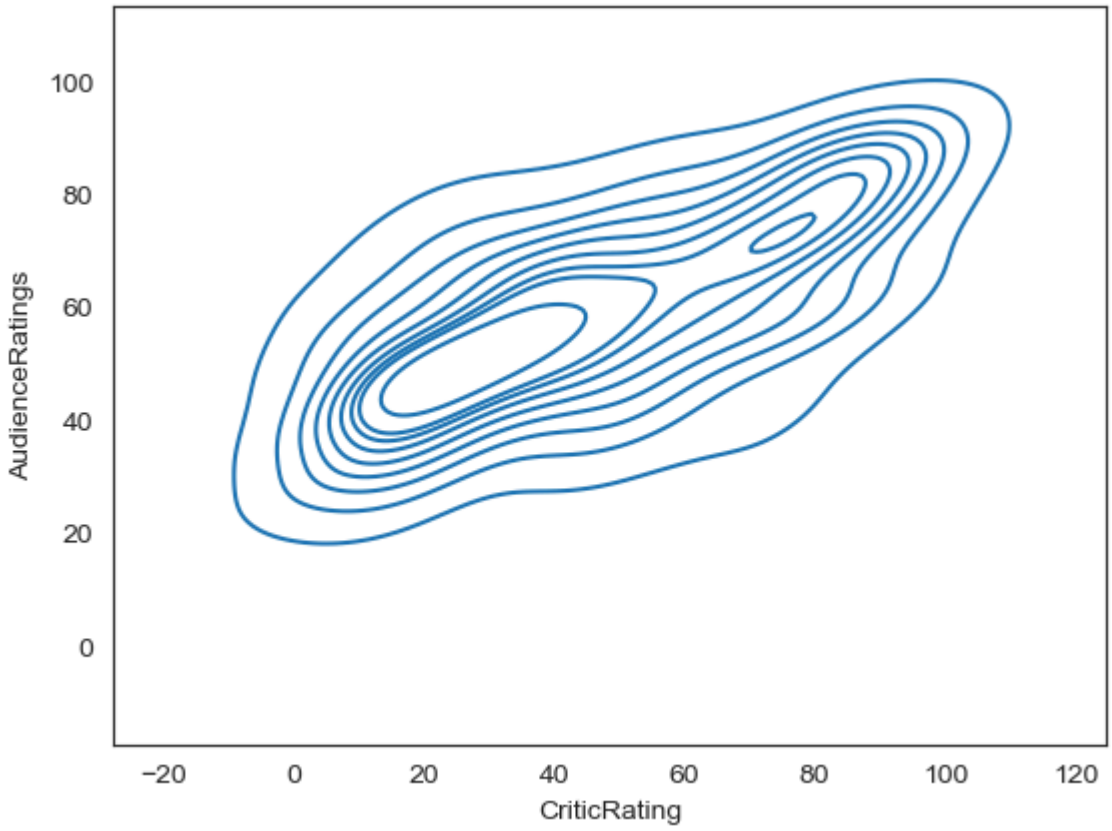


KDE plot to visualize audience and critic rating

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

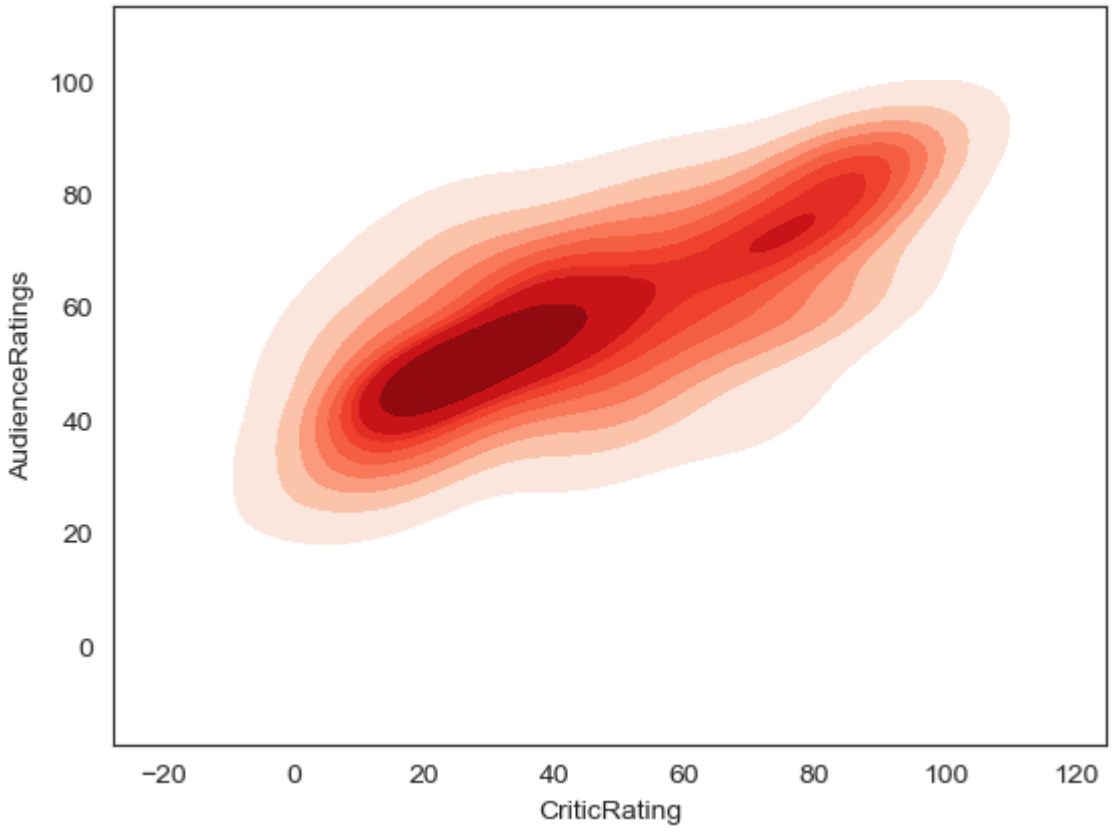
In [42]:
```
import seaborn as sns

sns.kdeplot(x=movie.CriticRating, y=movie.AudienceRatings)
```

Out[42]:
```
<Axes: xlabel='CriticRating', ylabel='AudienceRatings'>
```



In [43]:
```
import seaborn as sns

sns.kdeplot(x=movie.CriticRating, y=movie.AudienceRatings, shade=True, cmap='Red
```

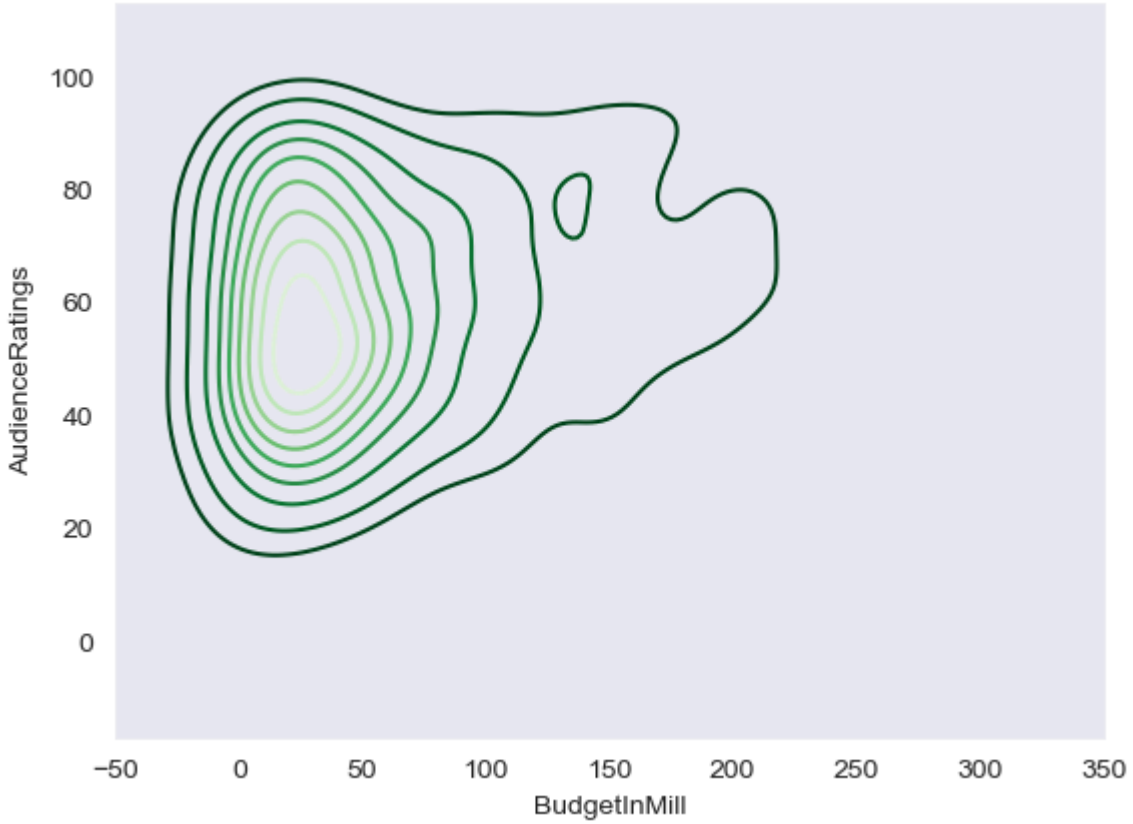
Out[43]: <Axes: xlabel='CriticRating', ylabel='AudienceRatings'>



In [44]:
```
sns.set_style('dark')
```

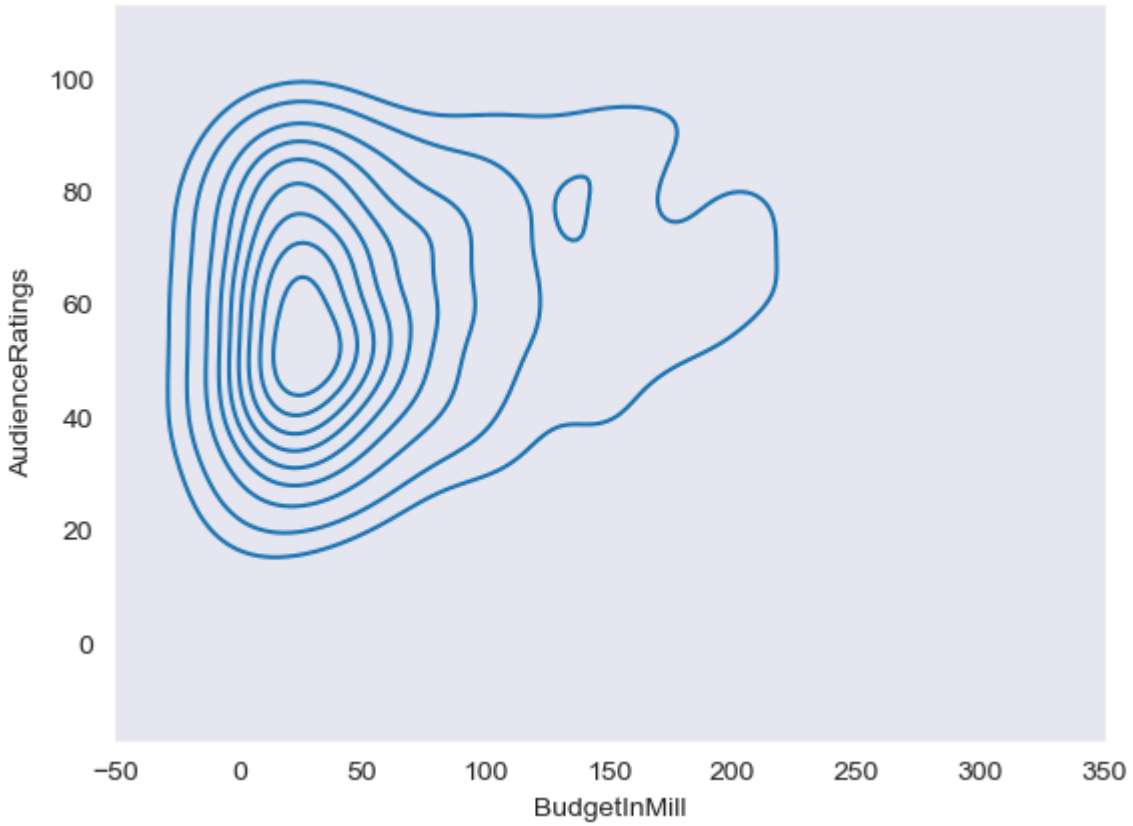
```
sns.kdeplot(x=movie.BudgetInMill,y=movie.AudienceRatings,shade_lowest=False,cmap
```

Out[44]: <Axes: xlabel='BudgetInMill', ylabel='AudienceRatings'>



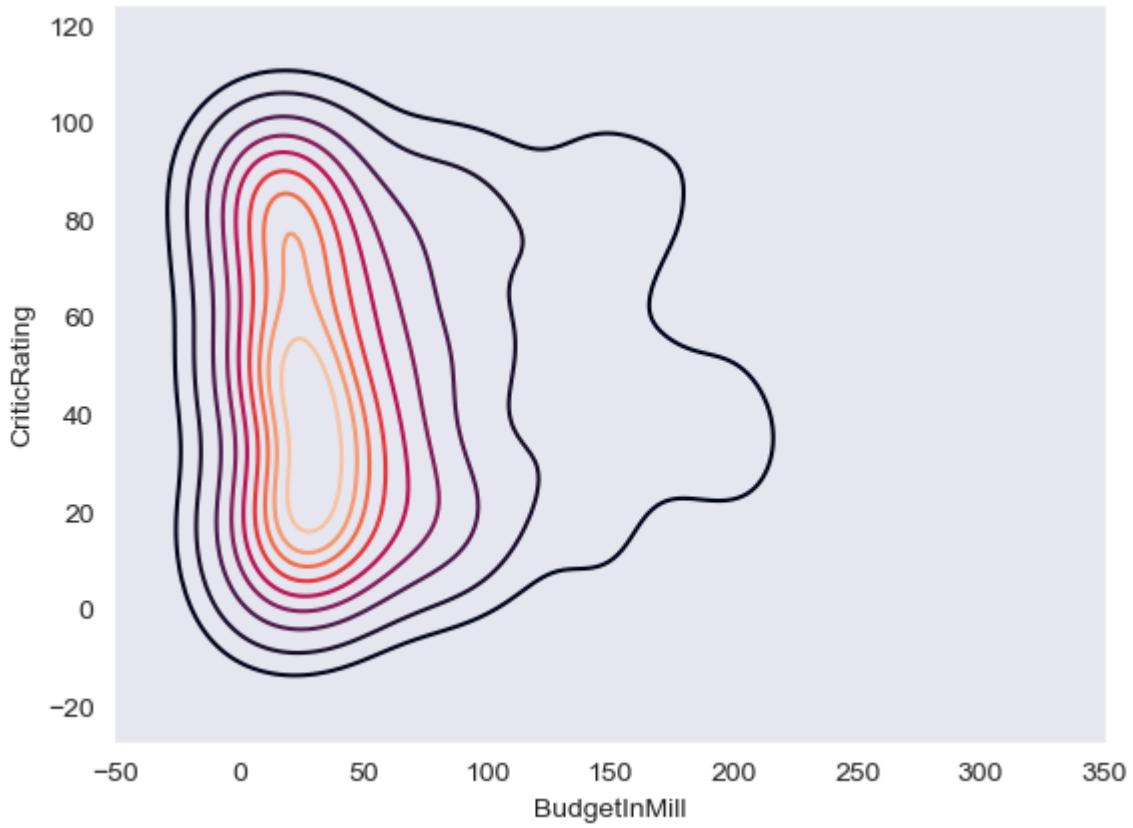
In [45]:
```
sns.set_style('dark')
sns.kdeplot(x=movie.BudgetInMill,y=movie.AudienceRatings)
```

Out[45]: <Axes: xlabel='BudgetInMill', ylabel='AudienceRatings'>

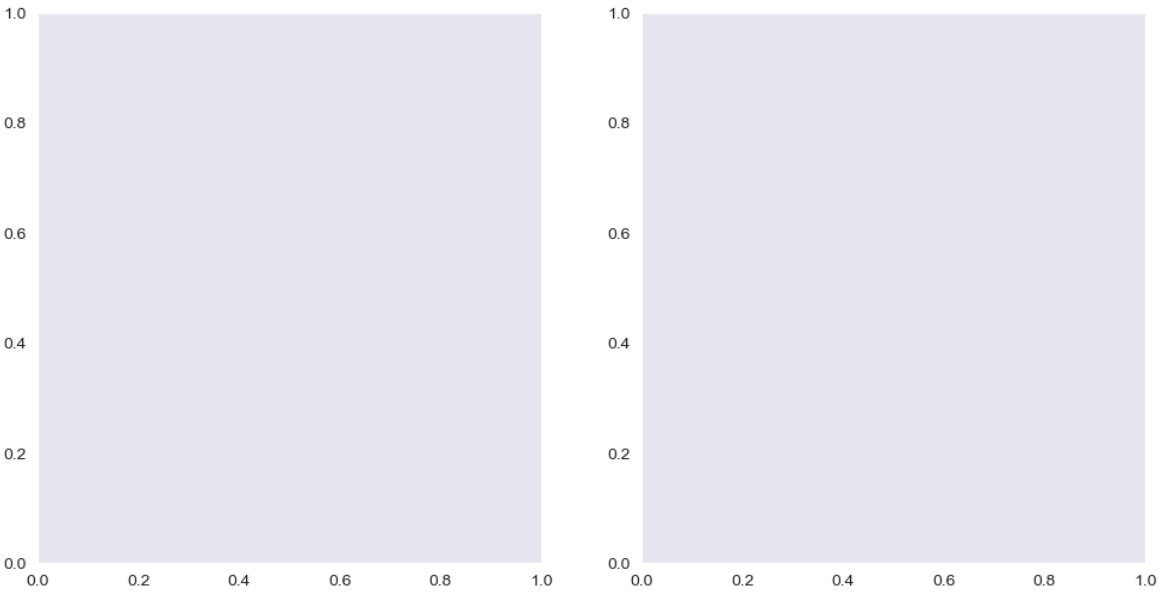


In [46]:
```
sns.set_style('dark')
sns.kdeplot(x=movie.BudgetInMill,y=movie.CriticRating,cmap='rocket')
```

Out[46]: <Axes: xlabel='BudgetInMill', ylabel='CriticRating'>

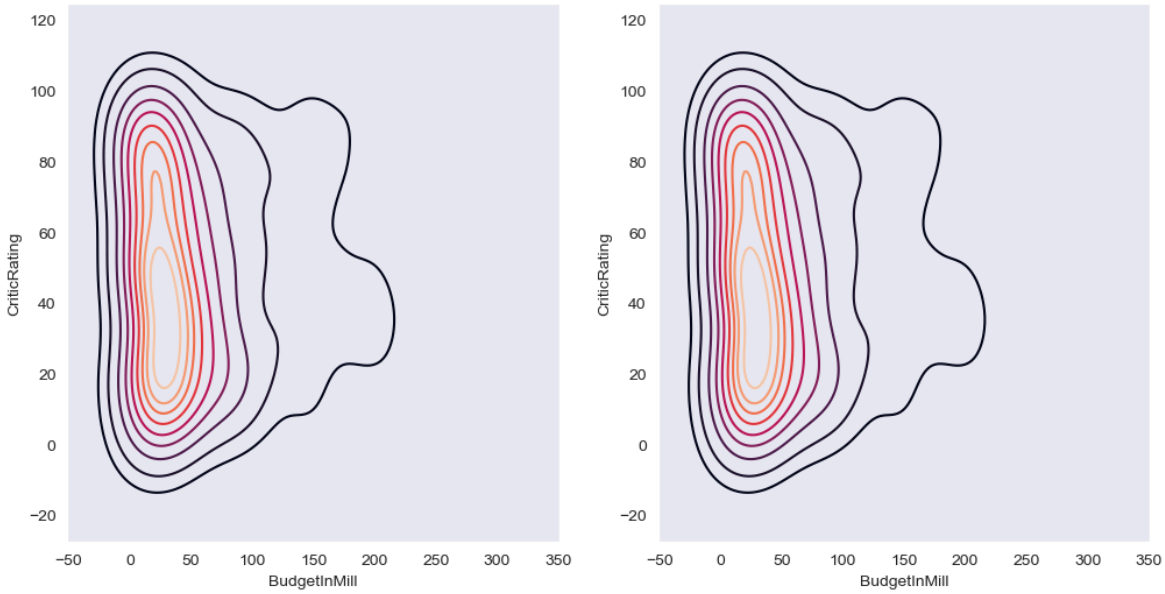


In [47]:
```
f,ax=plt.subplots(1,2,figsize=(12,6))
```



In [48]:
```
f,axes=plt.subplots(1,2,figsize=(12,6))
sns.kdeplot(x=movie.BudgetInMill,y=movie.CriticRating,cmap='rocket',ax=axes[0])
sns.kdeplot(x=movie.BudgetInMill,y=movie.CriticRating,cmap='rocket',ax=axes[1])
```

Out[48]: <Axes: xlabel='BudgetInMill', ylabel='CriticRating'>

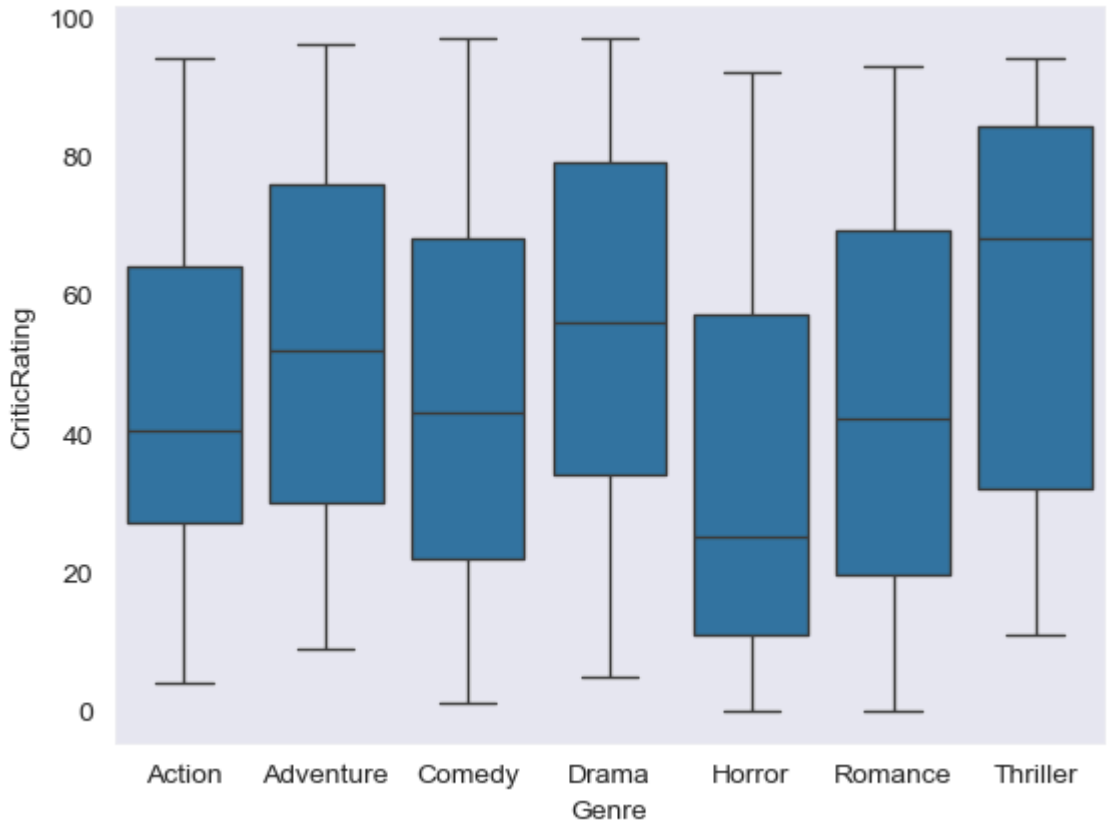


In [49]:
```
axes
```

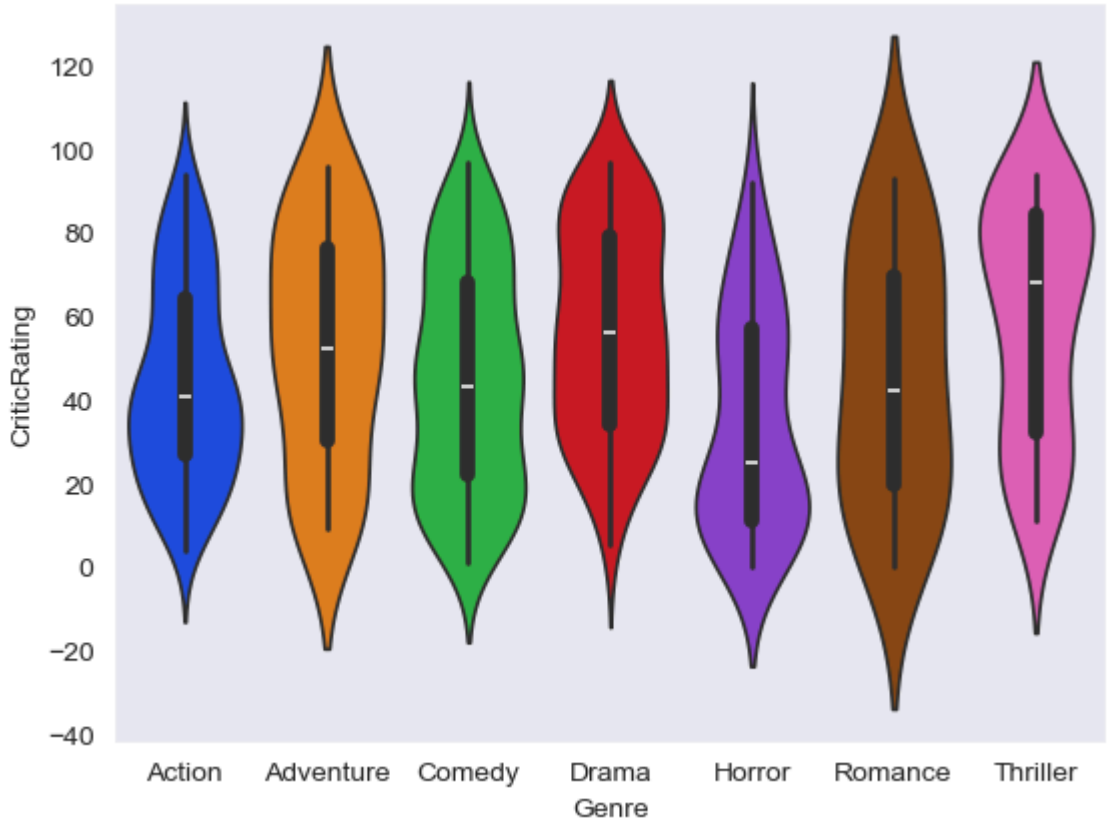
Out[49]:
```
array([<Axes: xlabel='BudgetInMill', ylabel='CriticRating'>,
       <Axes: xlabel='BudgetInMill', ylabel='CriticRating'>], dtype=object)
```

Box Plot

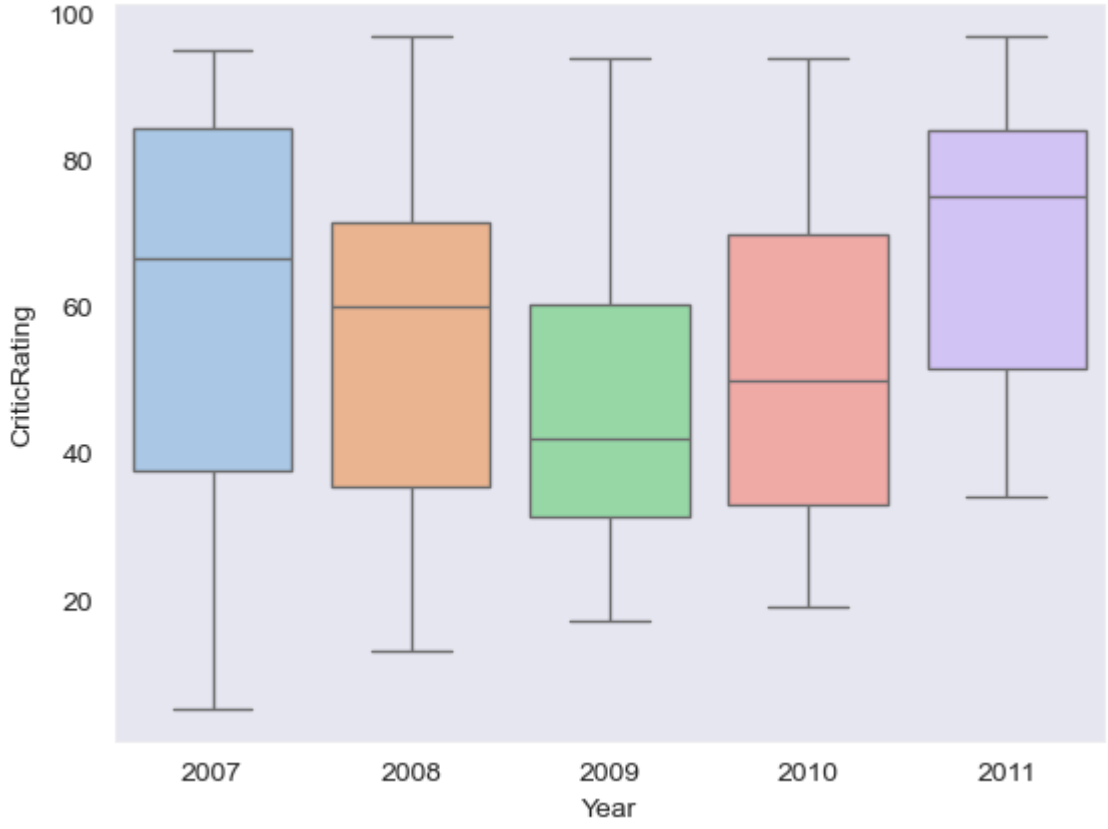
In [50]:
```
w = sns.boxplot(data=movie,x='Genre',y='CriticRating')
```



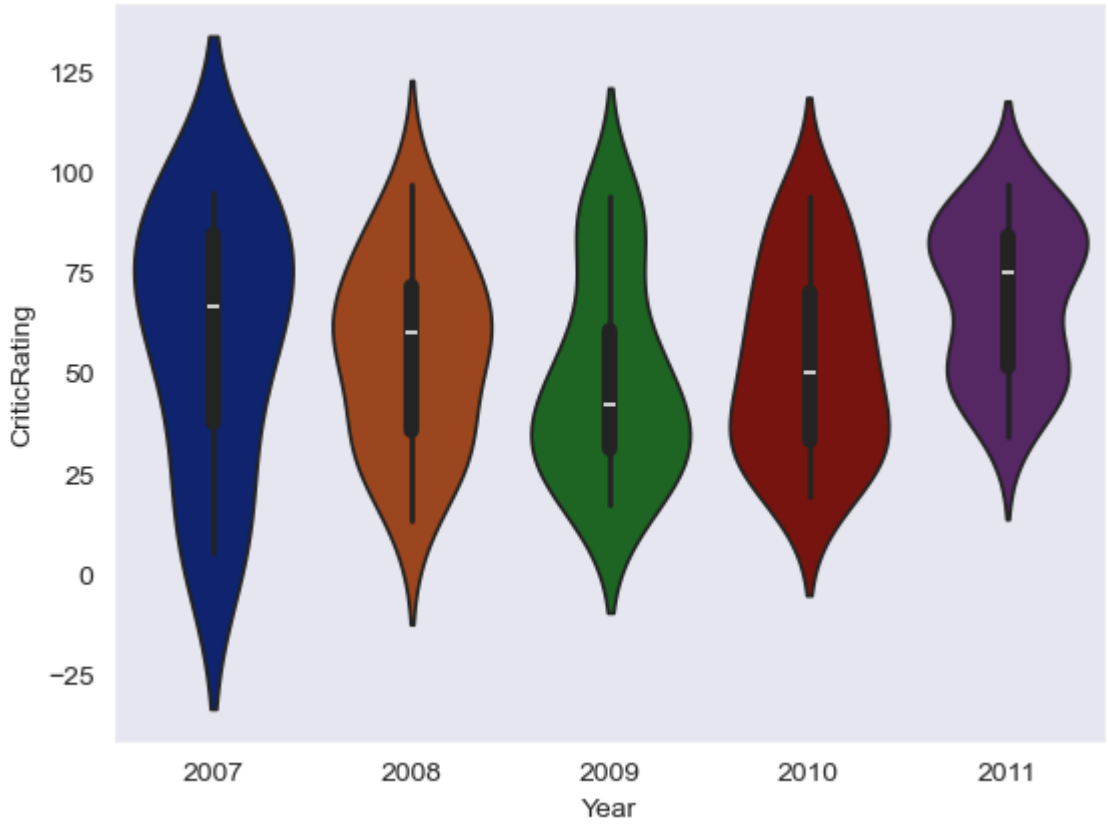
In [54]:
```
z = sns.violinplot(data=movie,x='Genre',y='CriticRating',palette='bright')
```



In [55]:
```
w1 = sns.boxplot(data=movie[movie.Genre=='Drama'],x='Year',y='CriticRating',pale
```



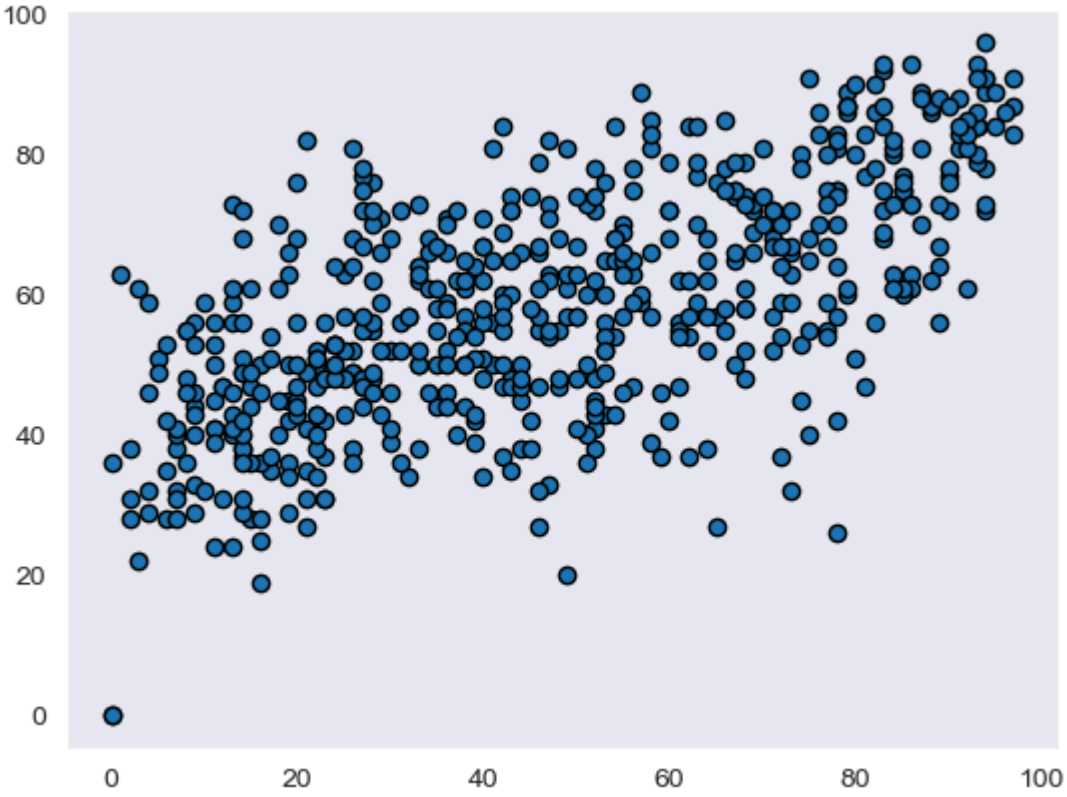
In [56]:
```
z= sns.violinplot(data=movie[movie.Genre=='Drama'],x='Year',y='CriticRating',pal
```



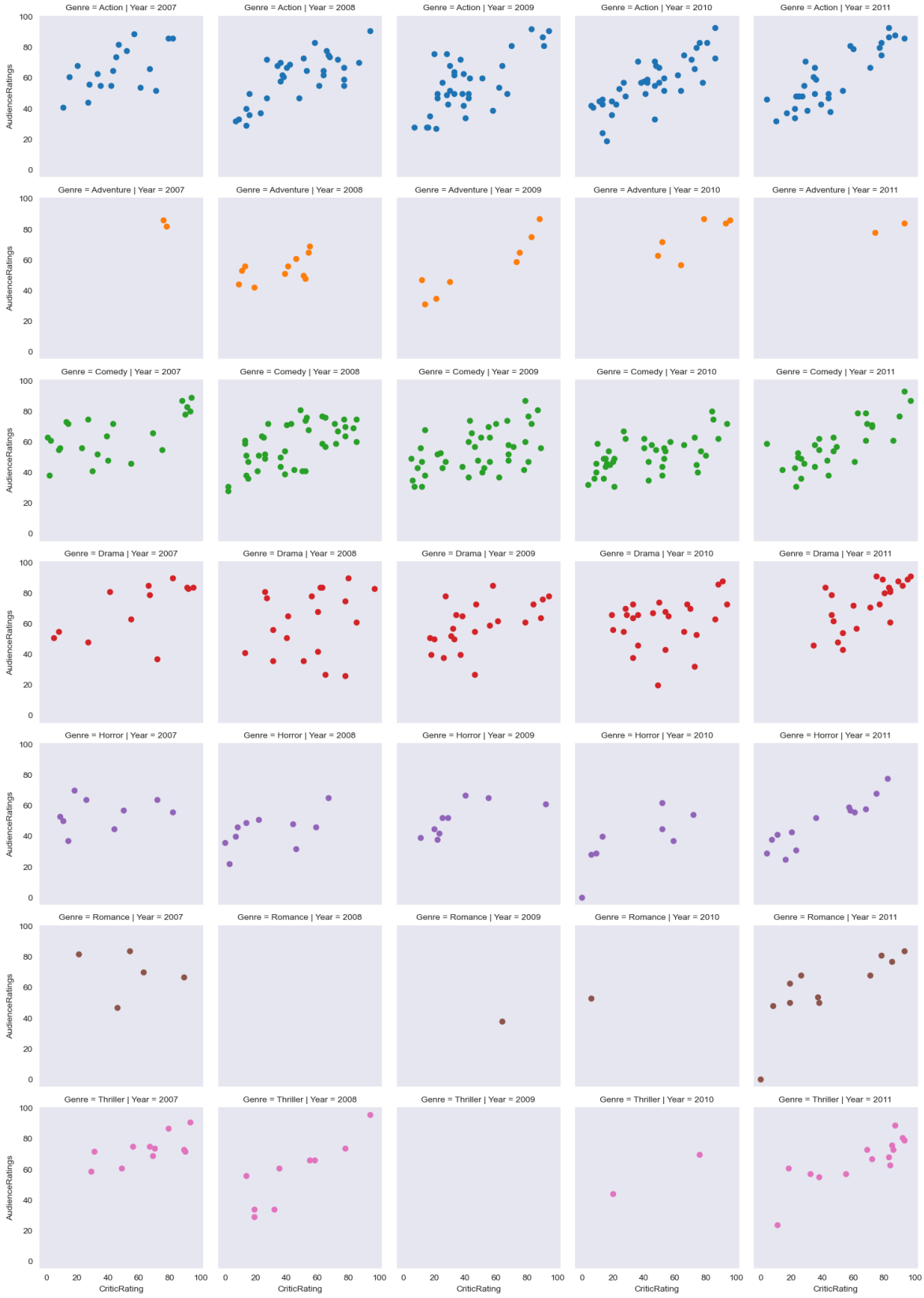
In [58]:
```
g = sns.FacetGrid(movie,row='Genre',col='Year',hue='Genre') #kinds of subplots
```

In [68]:
```
plt.scatter(x=movie.CriticRating,y=movie.AudienceRatings,edgecolor='black')
```

Out[68]: <matplotlib.collections.PathCollection at 0x186a7fb0830>



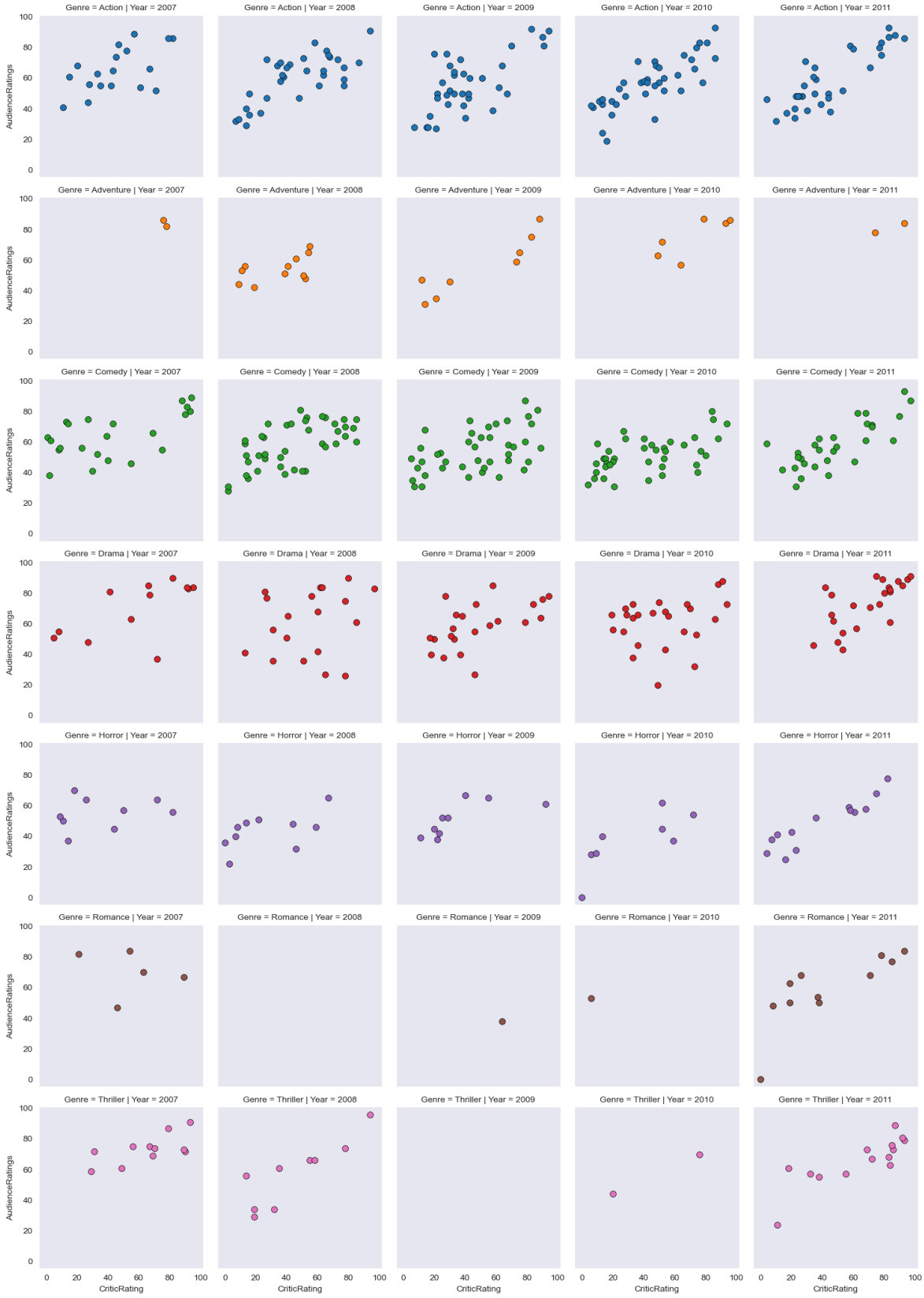
In [69]:
```
g = sns.FacetGrid(movie,row='Genre',col='Year',hue='Genre')
g=g.map(plt.scatter,'CriticRating','AudienceRatings')
```



In [71]:

```
g = sns.FacetGrid(movie,row='Genre',col='Year',hue='Genre')
g=g.map(plt.hist,'BudgetInMill') #scatter plots are mapped in facetgrids
```

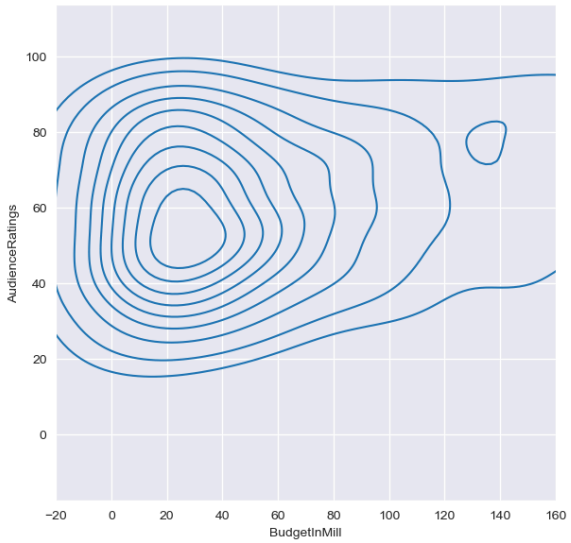
In [72]:
```
g = sns.FacetGrid(movie,row='Genre',col='Year',hue='Genre')
kws = dict(s=50,linewidth=0.5,edgecolor='black')
g=g.map(plt.scatter,'CriticRating','AudienceRatings',**kws)
```

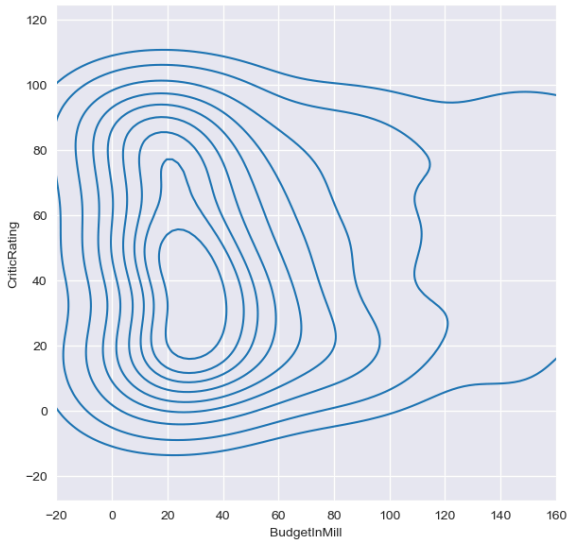


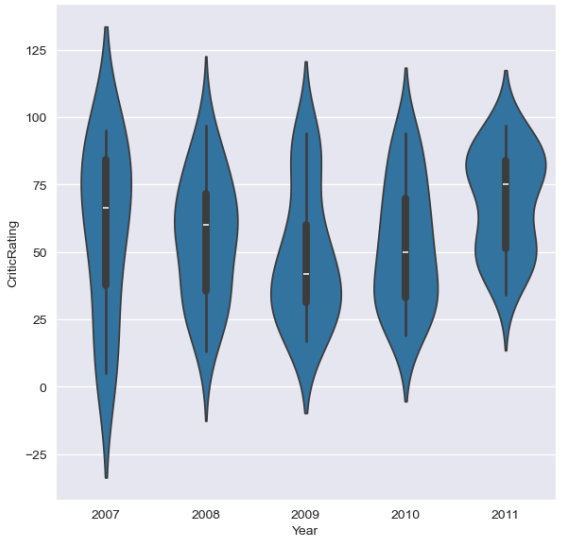
Building dashboards (dashboard - combination of chats)

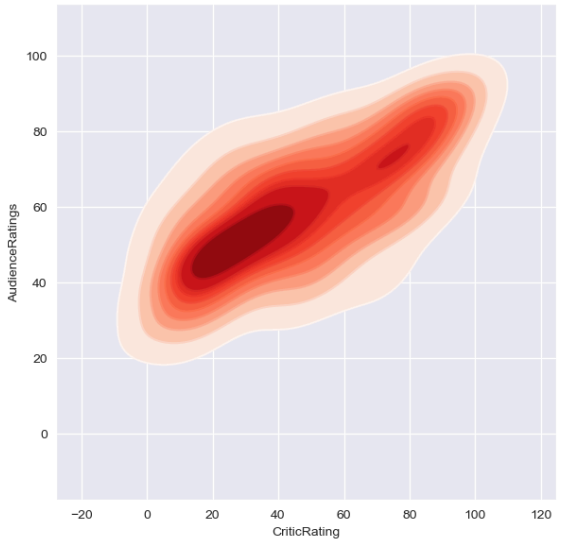
In [79]:
```
sns.set_style('darkgrid')
f, axes = plt.subplots (2,2, figsize = (15,15))

k1 = sns.kdeplot(x=movie.BudgetInMill,y=movie.AudienceRatings,ax=axes[0,0])
```

```
k2 = sns.kdeplot(x=movie.BudgetInMill,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.AudienceRatings,shade = True,shade
k4b = sns.kdeplot(x=movie.CriticRating,y=movie.AudienceRatings,cmap='Reds',ax =

plt.show()
```





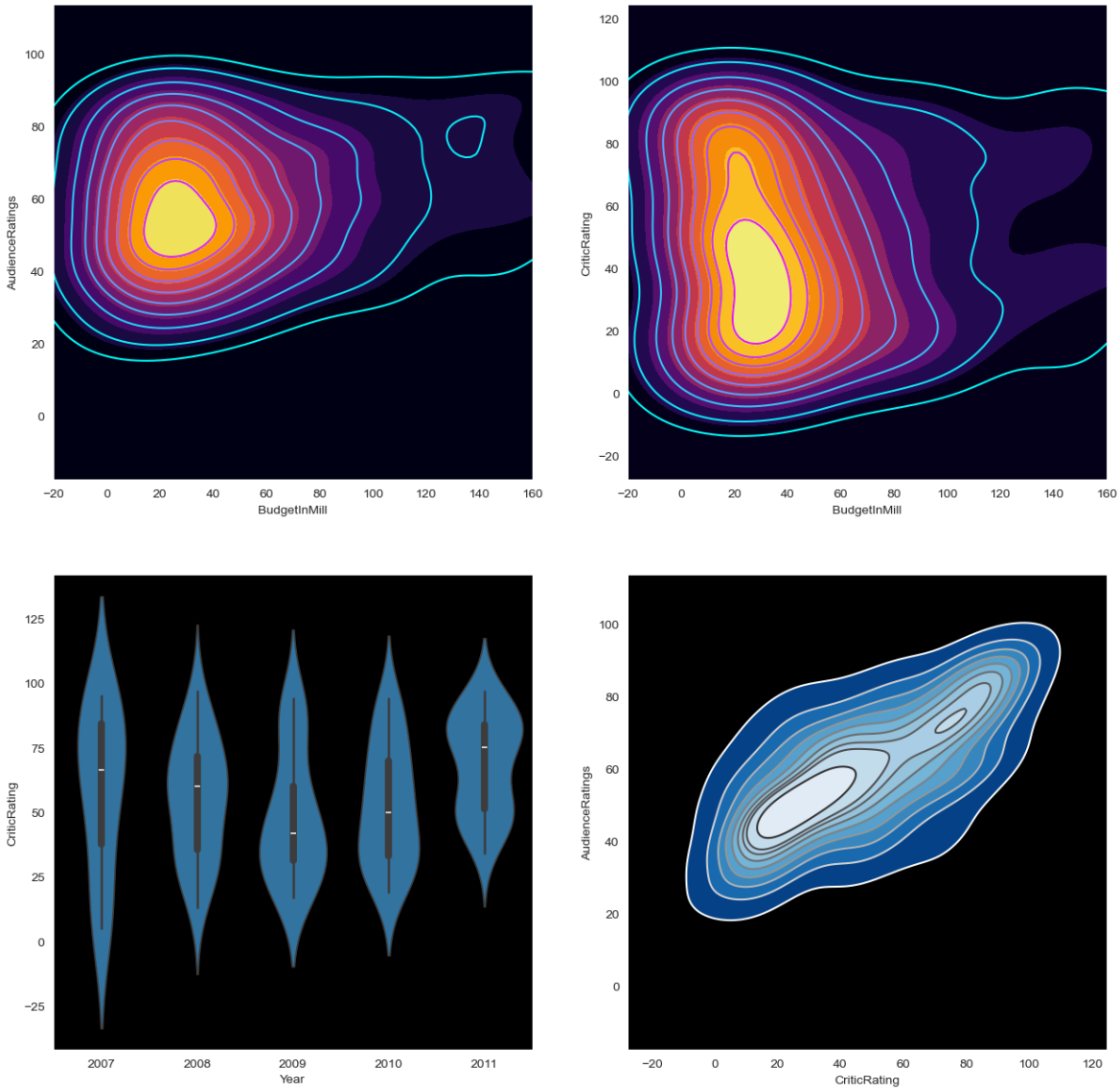




how to style the dashboard using diff color maps

In [86]:
```
sns.set_style('dark',{'axes.facecolor':'black'})
f,axes=plt.subplots(2,2,figsize=(15,15))

#plot[0,0]
k1 = sns.kdeplot(x=movie.BudgetInMill,y=movie.AudienceRatings, \
                 shade = True, shade_lowest=True,cmap = 'inferno', \
                 ax = axes[0,0])
k1b = sns.kdeplot(x=movie.BudgetInMill,y=movie.AudienceRatings, \
                 cmap = 'cool',ax = axes[0,0])
#plot [0,1]
```

```
k2 = sns.kdeplot(x=movie.BudgetInMill,y=movie.CriticRating,\
                 shade=True, shade_lowest=True, cmap='inferno',\
                 ax = axes[0,1])
k2b = sns.kdeplot(x=movie.BudgetInMill,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.AudienceRatings, \
                 shade = True,shade_lowest=False,cmap='Blues_r', \
                 ax=axes[1,1])
k4b = sns.kdeplot(x=movie.CriticRating,y=movie.AudienceRatings, \
                  cmap='gist_gray_r',ax = axes[1,1])

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

plt.show()
```



In this, we have completed performing operations on Analysis on Movie Rating, plotting grapg for better visualization like hist,boxplot,lmpolot,scatter,kdeplot,subplot,violinplot,facetgrid,stackedhist and finally ended with building a dashboard

In [ ]: