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

In [2]:
```
pd.__version__
```

Out[2]:
```
'2.2.2'
```

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

Out[3]:
```
'C:\\Users\\velug'
```

In [4]:
```
movies=pd.read_csv(r'C:\Users\velug\Downloads\Movie-Rating.csv')
```

In [5]:
```
movies
```

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]:
```
len(movies)
```

Out[6]:
```
559
```

In [7]:
```
movies.head()
```

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

In [8]:
```
movies.tail()
```

Out[8]:

	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 [9]:
```
movies.columns
```

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

In [10]:
```
movies.columns=['Film', 'Genre', 'CriticRatings', 'AudienceRatings','Budgetmillion', 'Year']
```

In [11]:
```
movies.columns
```

Out[11]:
```
Index(['Film', 'Genre', 'CriticRatings', 'AudienceRatings', 'Budgetmillion',
       'Year'],
      dtype='object')
```

In [12]:
```
movies.info()#Print a concise summary of a DataFrame.This method prints information about a DataFrame including
                #the index dtype and columns, non-null values and memory usage.
```

```
<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   CriticRatings    559 non-null    int64 
 3   AudienceRatings  559 non-null    int64 
 4   Budgetmillion    559 non-null    int64 
 5   Year             559 non-null    int64 
dtypes: int64(4), object(2)
memory usage: 26.3+ KB
```

In [13]:
```
movies.describe() # Generate descriptive statistics.
# if you look at the year the data type is int but when you look at the mean value it showing 2009 which is meaningless
# we have to change to categroy type
# also from object datatype we will convert to category datatypes
```

Out[13]:

	CriticRatings	AudienceRatings	Budgetmillion	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 [14]:
```
movies['Film']
```

Out[14]:
```
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: object
```

In [15]:
```
movies.Film
```

Out[15]:
```
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: object
```

In [16]:
```
movies.Film=movies.Film.astype('category')
#(.astype) Cast a pandas object to a specified dtype ``dtype``.
```

In [17]:
```
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    object  
 2   CriticRatings    559 non-null    int64   
 3   AudienceRatings  559 non-null    int64   
 4   Budgetmillion    559 non-null    int64   
 5   Year             559 non-null    int64   
dtypes: category(1), int64(4), object(1)
memory usage: 43.6+ KB
```

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

In [19]:
```
movies.info() # we change int to category
```

```
<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   CriticRatings    559 non-null    int64   
 3   AudienceRatings  559 non-null    int64   
 4   Budgetmillion    559 non-null    int64   
 5   Year             559 non-null    category
dtypes: category(3), int64(3)
memory usage: 36.5 KB
```

In [20]:
```
movies.Genre.cat.categories
```

Out[20]:
```
Index(['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'Romance',
       'Thriller'],
      dtype='object')
```

In [21]:
```
movies.describe() # we doesn't get the year because we change it to category datatype
```

Out[21]:

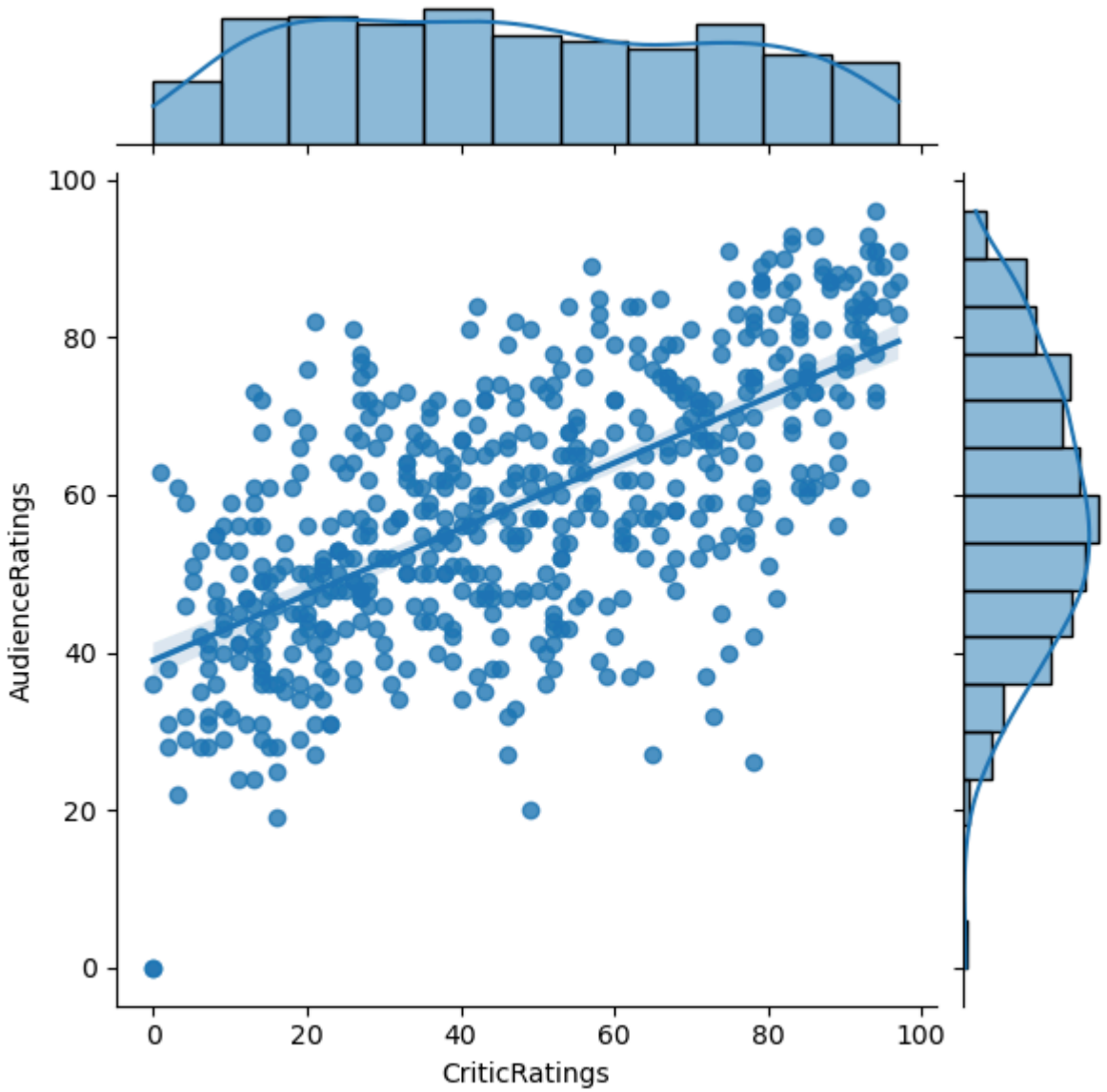
	CriticRatings	AudienceRatings	Budgetmillion
count	559.000000	559.000000	559.000000
mean	47.309481	58.744186	50.236136
std	26.413091	16.826887	48.731817
min	0.000000	0.000000	0.000000
25%	25.000000	47.000000	20.000000
50%	46.000000	58.000000	35.000000
75%	70.000000	72.000000	65.000000
max	97.000000	96.000000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

In [22]:
```
movies.CriticRatings
```

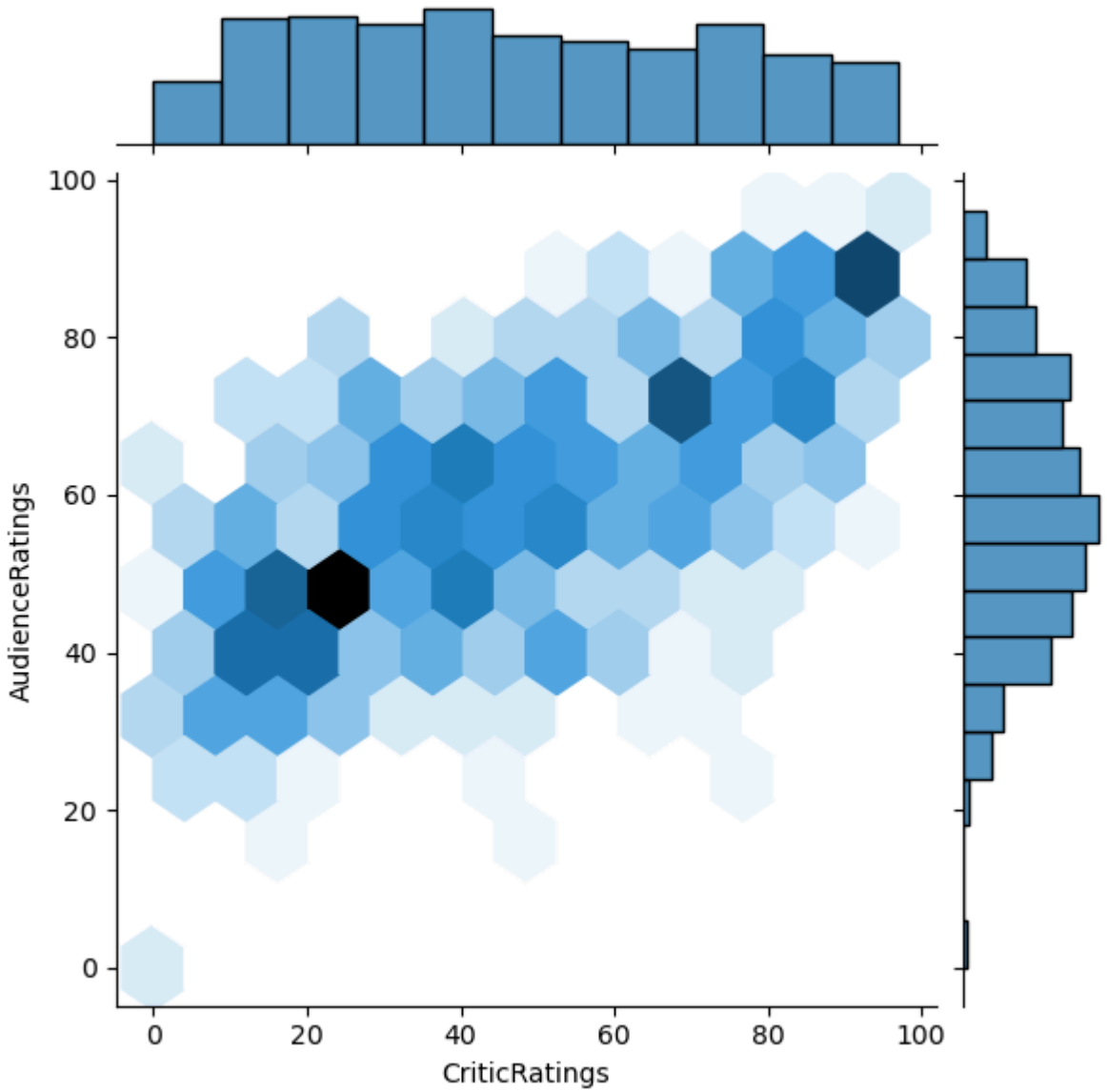
Out[22]:
```
0      87
1       9
2      30
3      93
4      55
       ..
554    26
555    68
556    89
557    90
558    14
Name: CriticRatings, Length: 559, dtype: int64
```

In [23]:
```
from matplotlib import pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

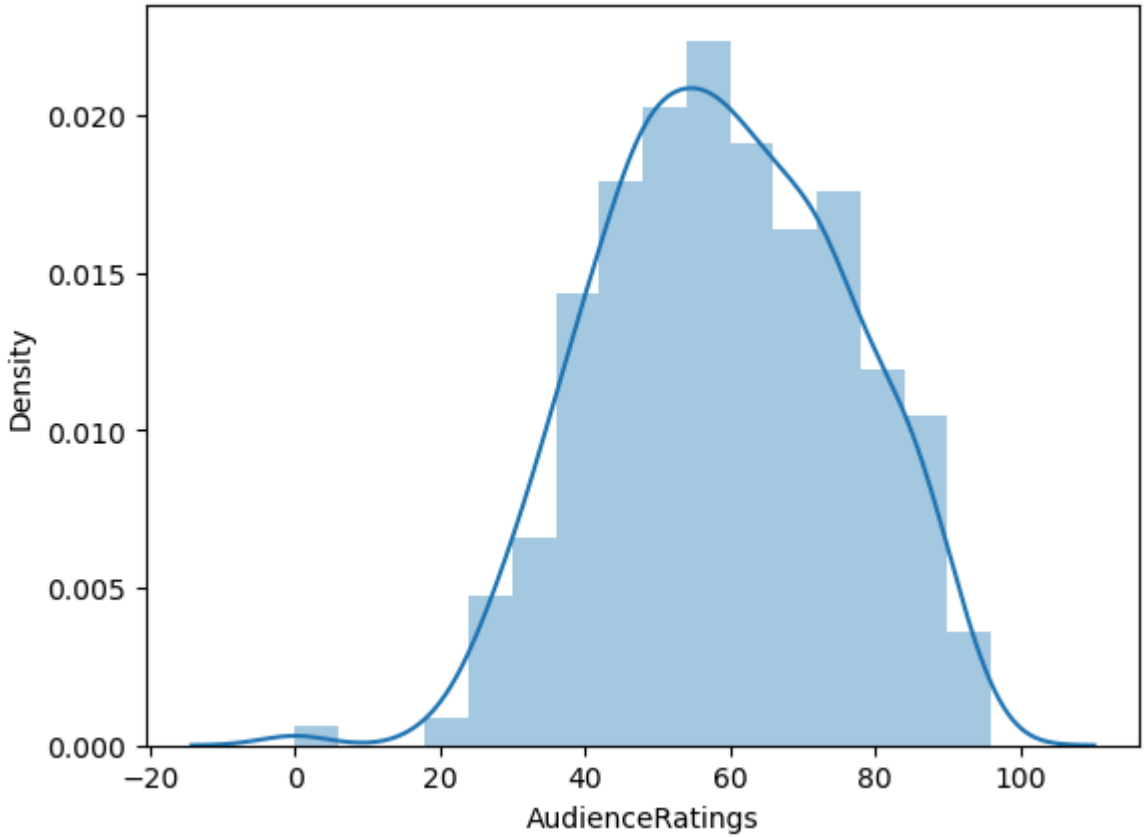
In [24]:
```
s = sns.jointplot( data = movies, x = 'CriticRatings', y = 'AudienceRatings', kind ="reg")
```



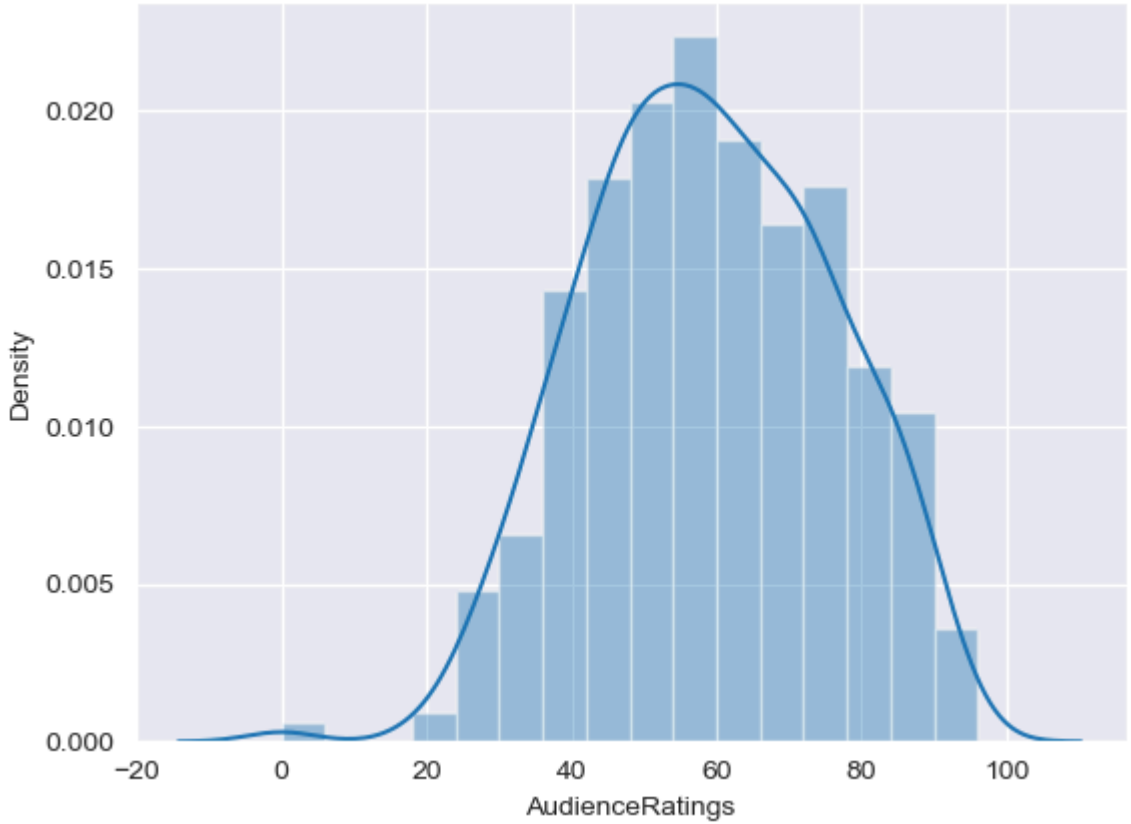
In [25]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind = 'hex')
```



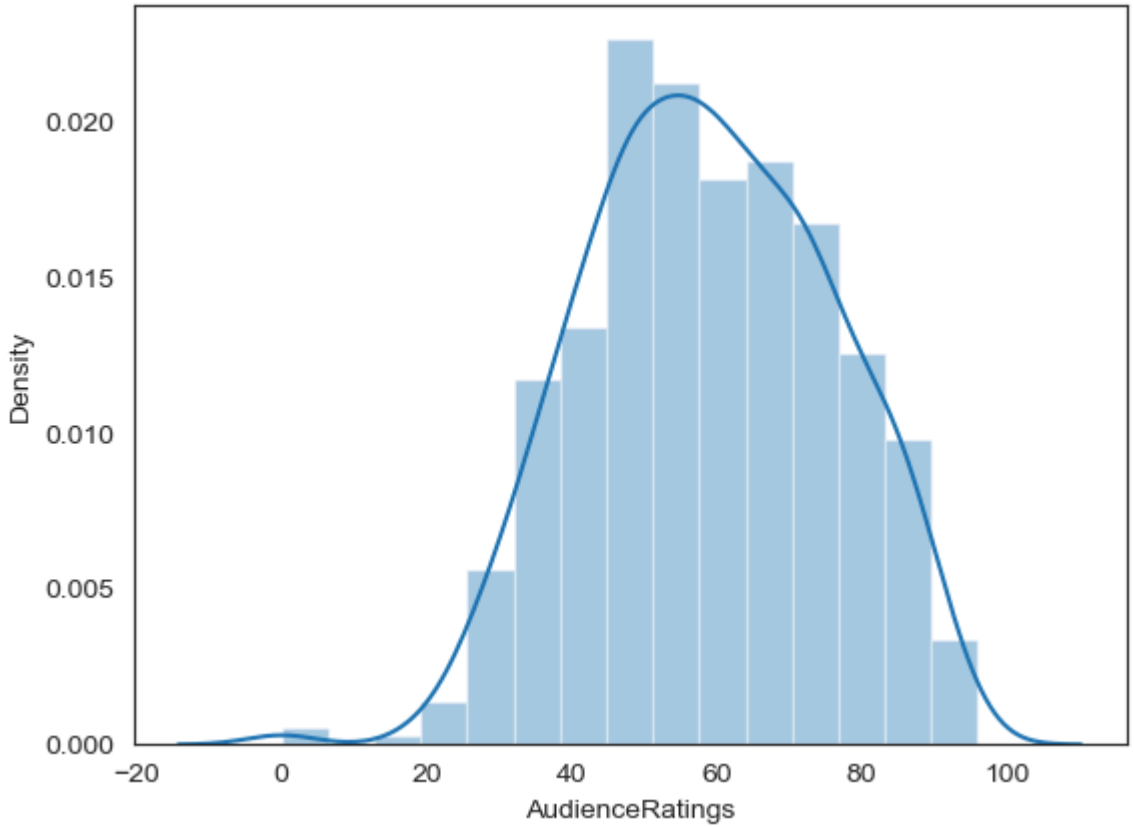
In [26]:
```
m1 = sns.distplot(movies.AudienceRatings)
```



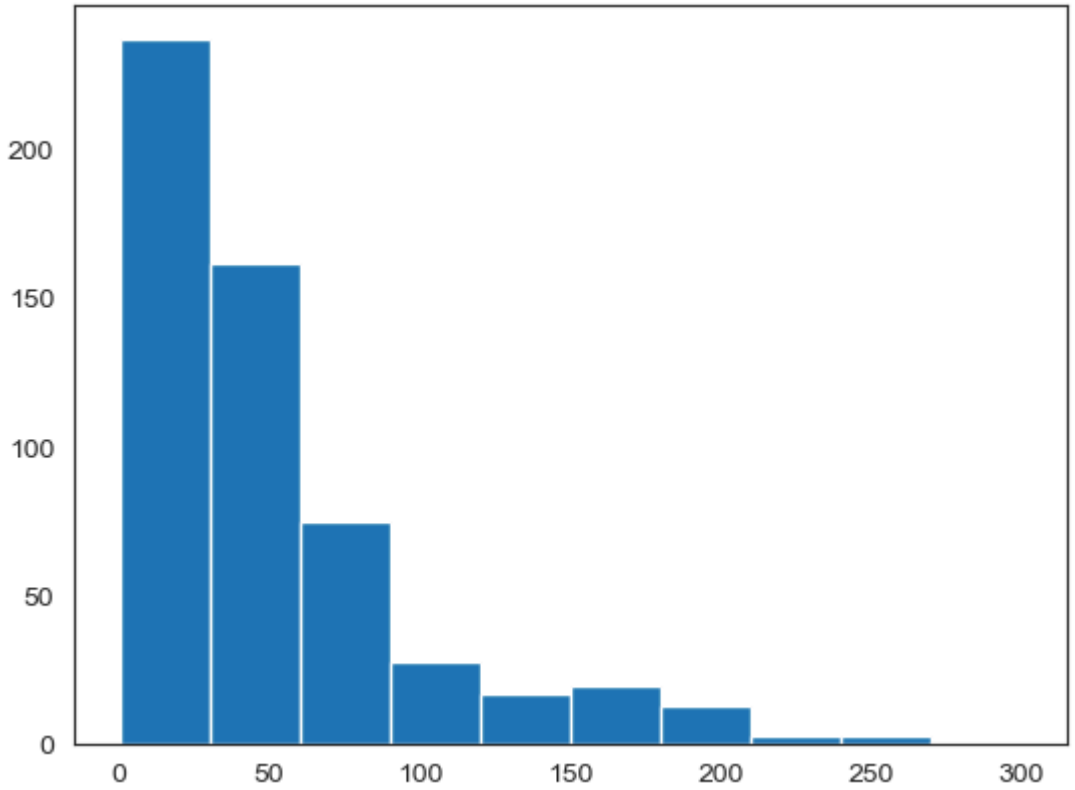
In [27]:
```
sns.set_style('darkgrid')
m1 = sns.distplot(movies.AudienceRatings)
```



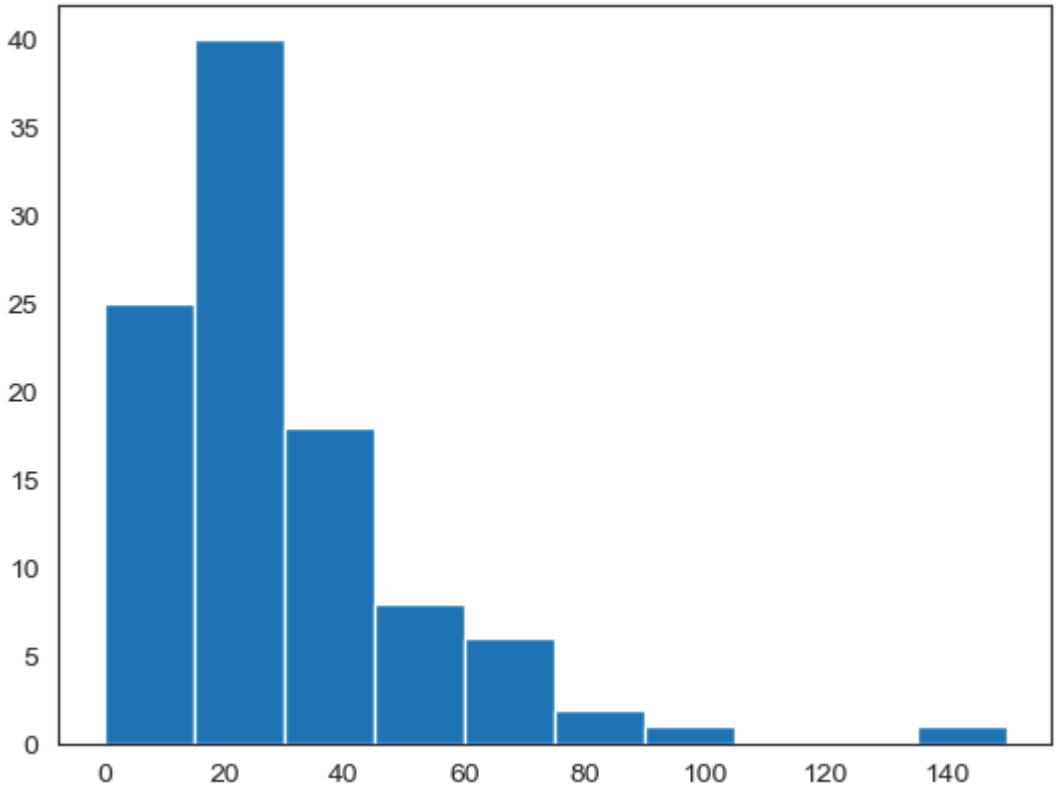
In [28]:
```
sns.set_style('white')
m2 = sns.distplot(movies.AudienceRatings, bins = 15)
```



In [29]:
```
plt.hist(movies.Budgetmillion)
plt.show()
```



In [30]:
```
plt.hist(movies[movies.Genre == 'Drama'].Budgetmillion)
plt.show()
```

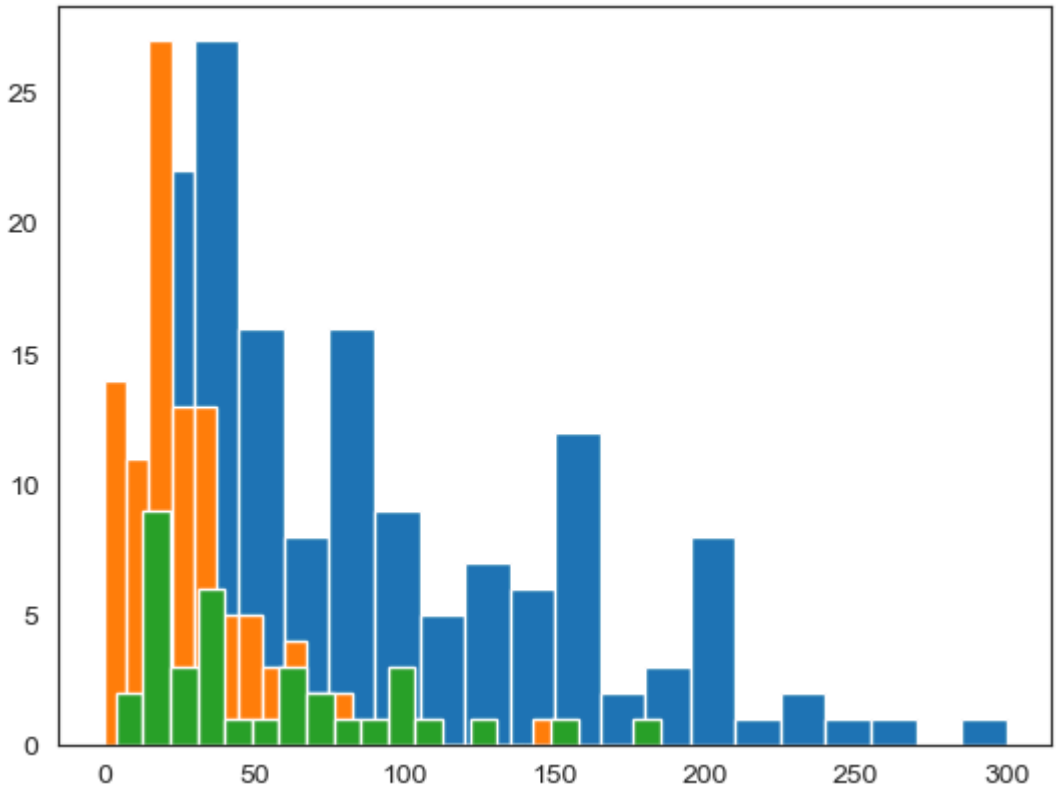


In [31]:
```
movies.head()
```

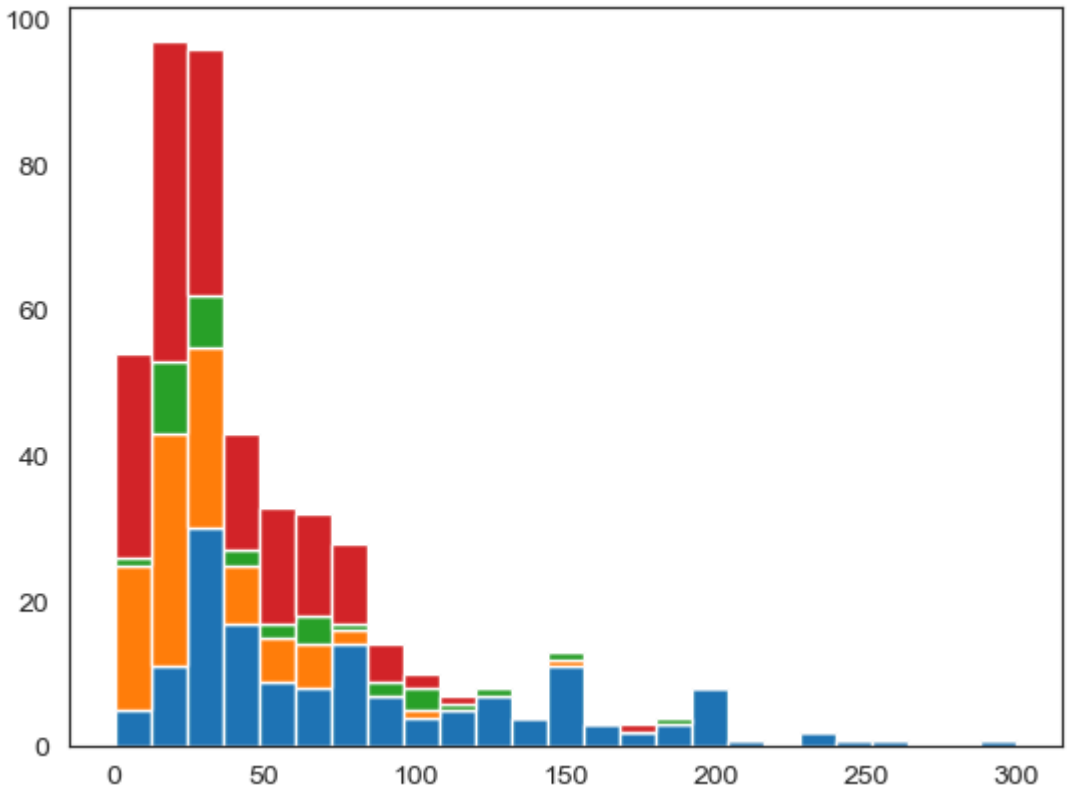
Out[31]:

	Film	Genre	CriticRatings	AudienceRatings	Budgetmillion	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 [32]:
```
plt.hist(movies[movies.Genre == 'Action'].Budgetmillion, bins = 20)
plt.hist(movies[movies.Genre == 'Drama'].Budgetmillion, bins = 20)
plt.hist(movies[movies.Genre == 'Thriller'].Budgetmillion, bins = 20)


plt.show()
```



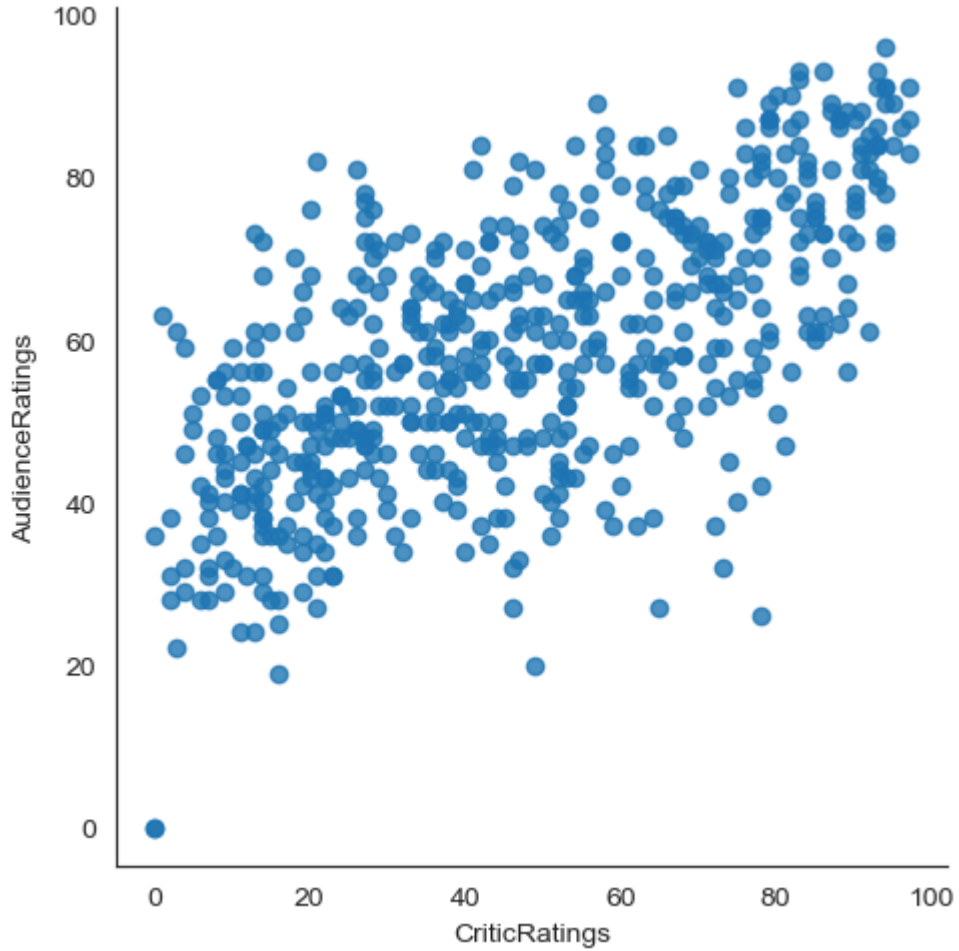
In [33]:
```
plt.hist([movies[movies.Genre == 'Action'].Budgetmillion,\
          movies[movies.Genre == 'Drama'].Budgetmillion, \
          movies[movies.Genre == 'Thriller'].Budgetmillion, \
          movies[movies.Genre == 'Comedy'].Budgetmillion],
         bins = 25, stacked = True)
plt.show()
```



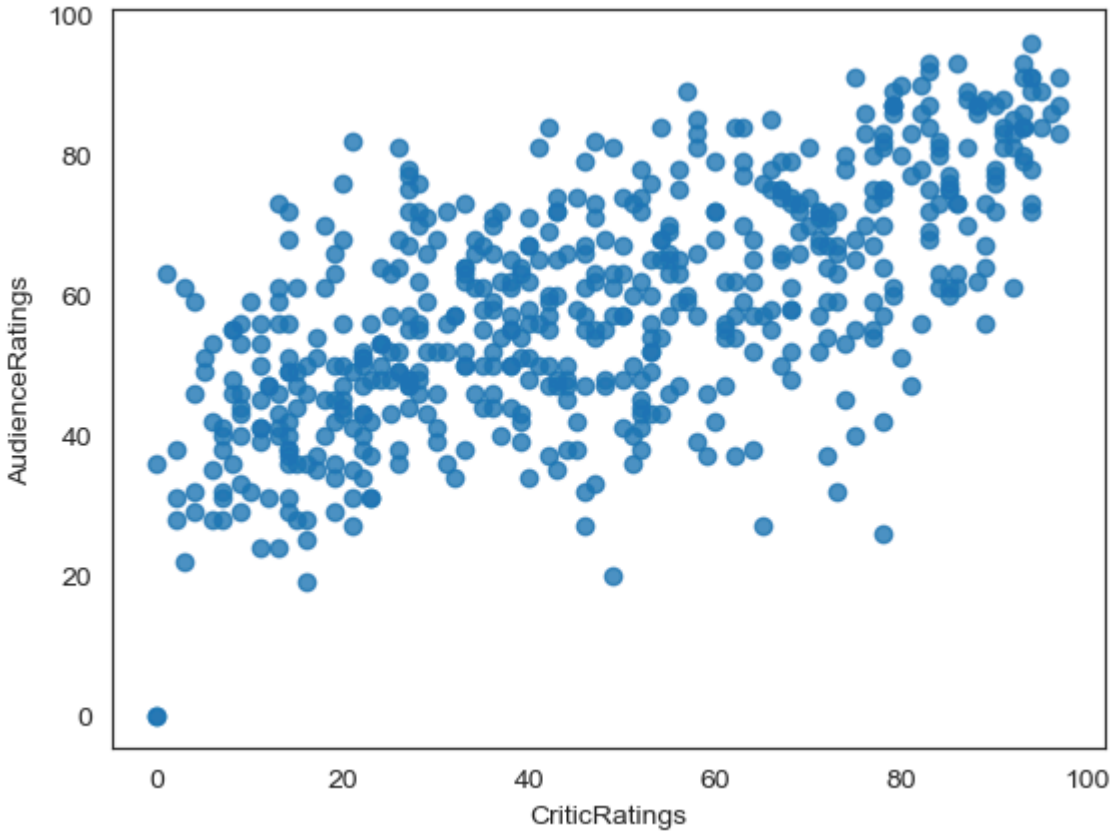
In [34]:
```
for gen in movies.Genre.cat.categories:
    print(gen)
```

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

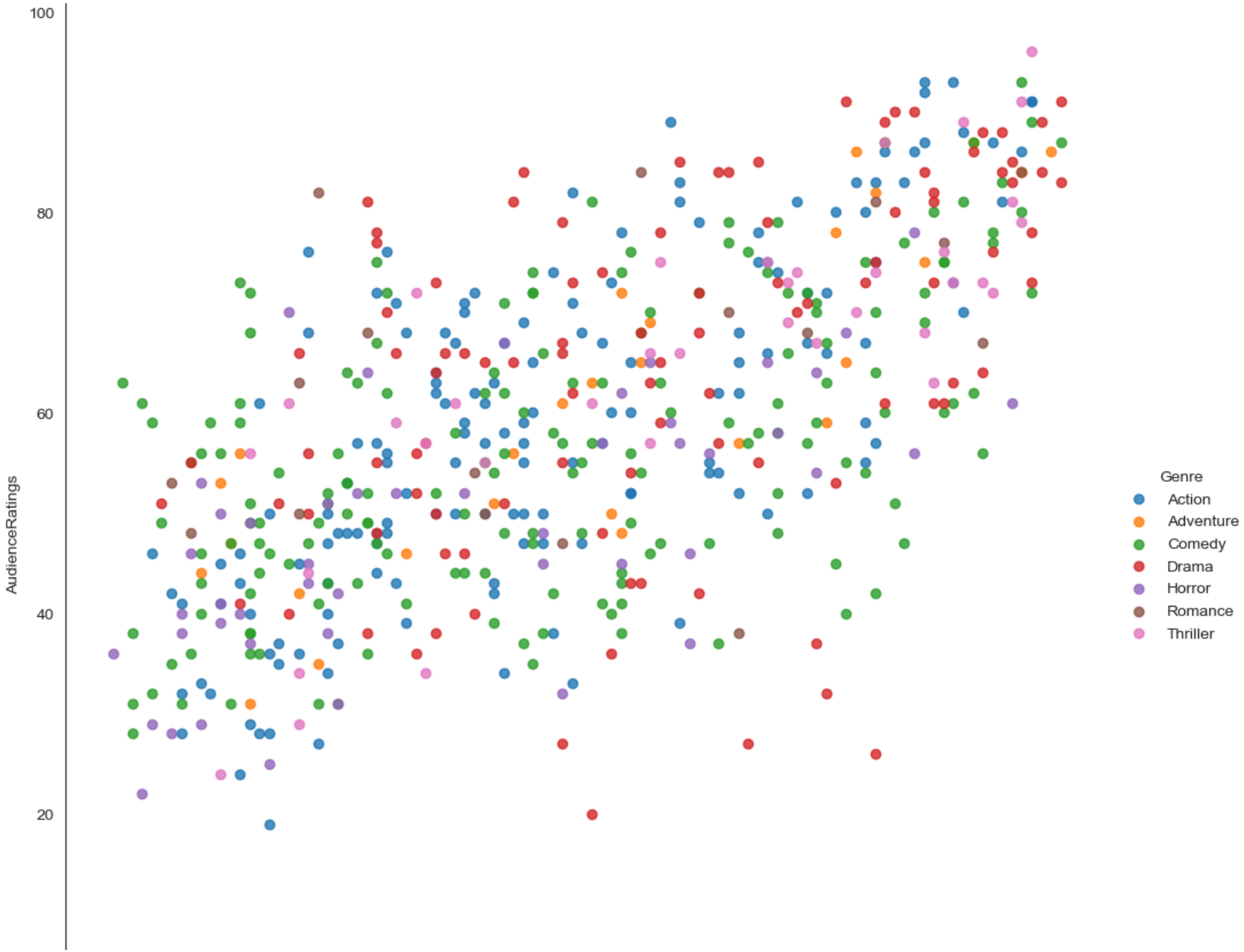
In [35]:
```
vis1 = sns.lmplot(data=movies, x='CriticRatings', y='AudienceRatings', fit_reg=False)
```



In [36]:
```
vis5 = sns.regplot(data=movies, x='CriticRatings', y='AudienceRatings',fit_reg=False, )
```



In [37]:
```
vis1 = sns.lmplot(data=movies, x='CriticRatings', y='AudienceRatings',fit_reg=False, hue = 'Genre', height =10,aspect=1)
```

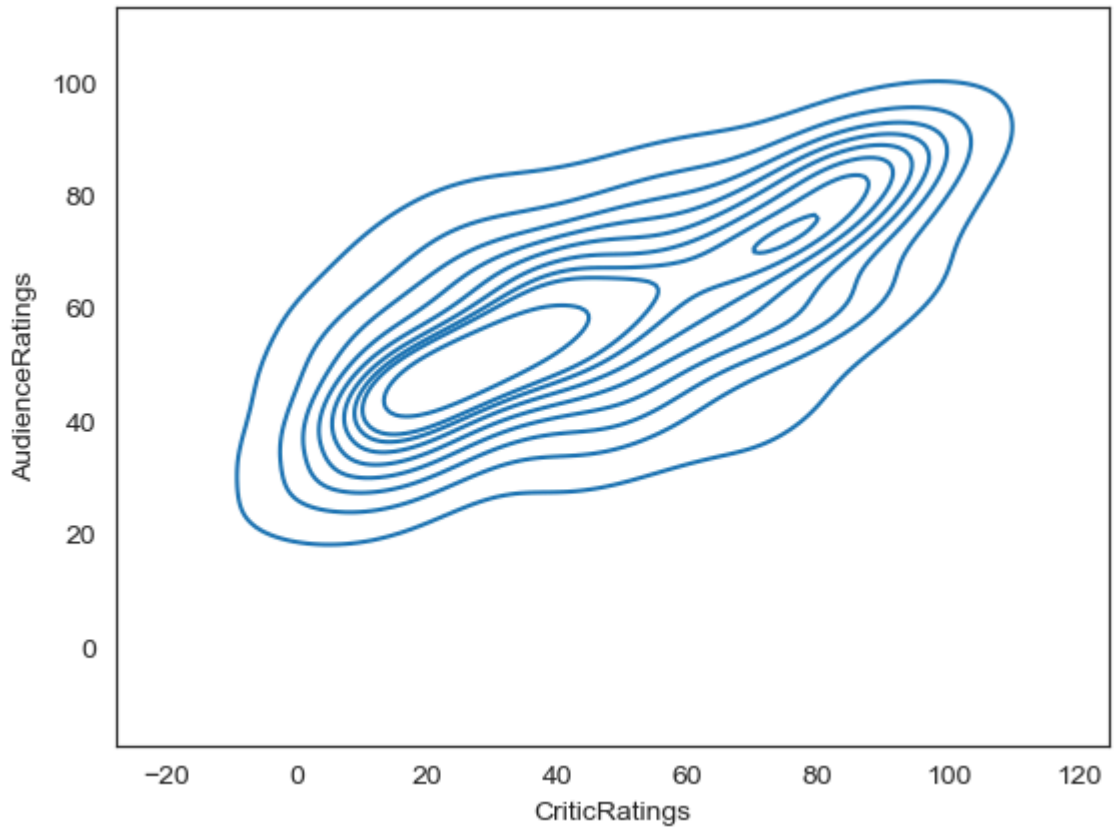

100
80
60
40
20
AudienceRatings
Genre
Action
Adventure
Comedy
Drama
Horror
Romance
Thriller

Kernal Density Estimate plot (KDE PLOT)

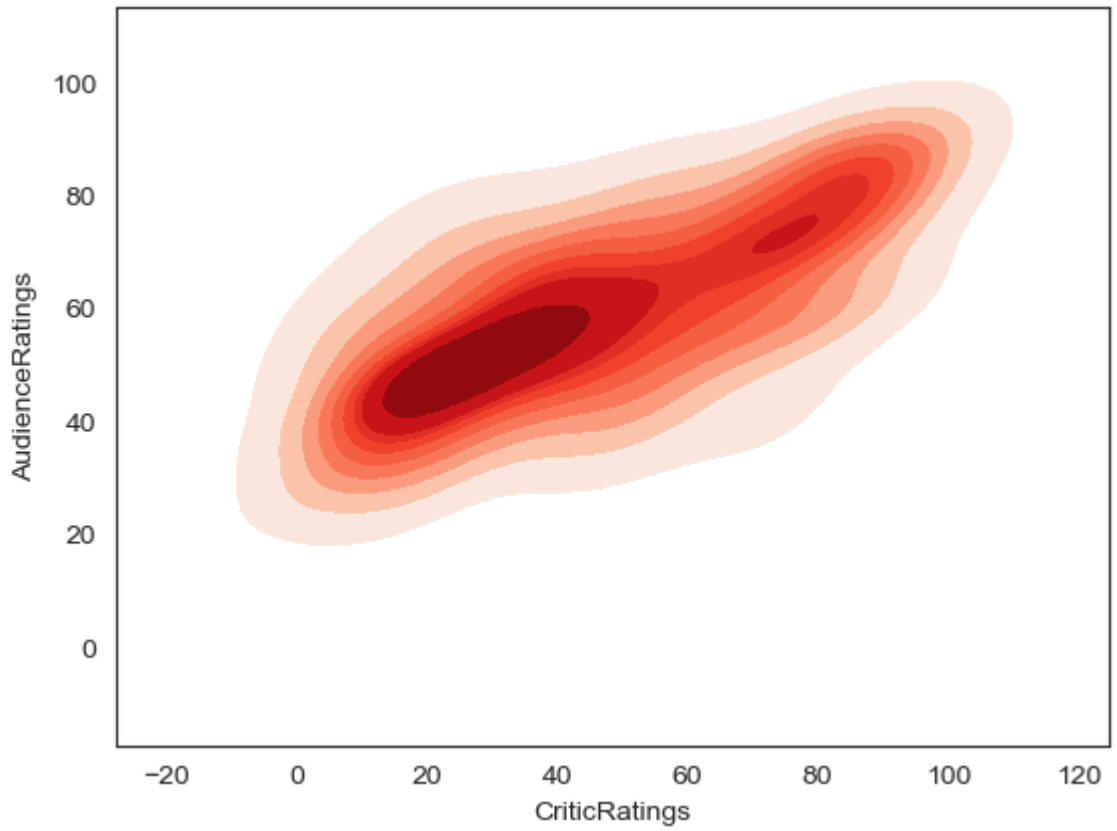
how can i visulize audience rating & critics rating . using scatterplot

- 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

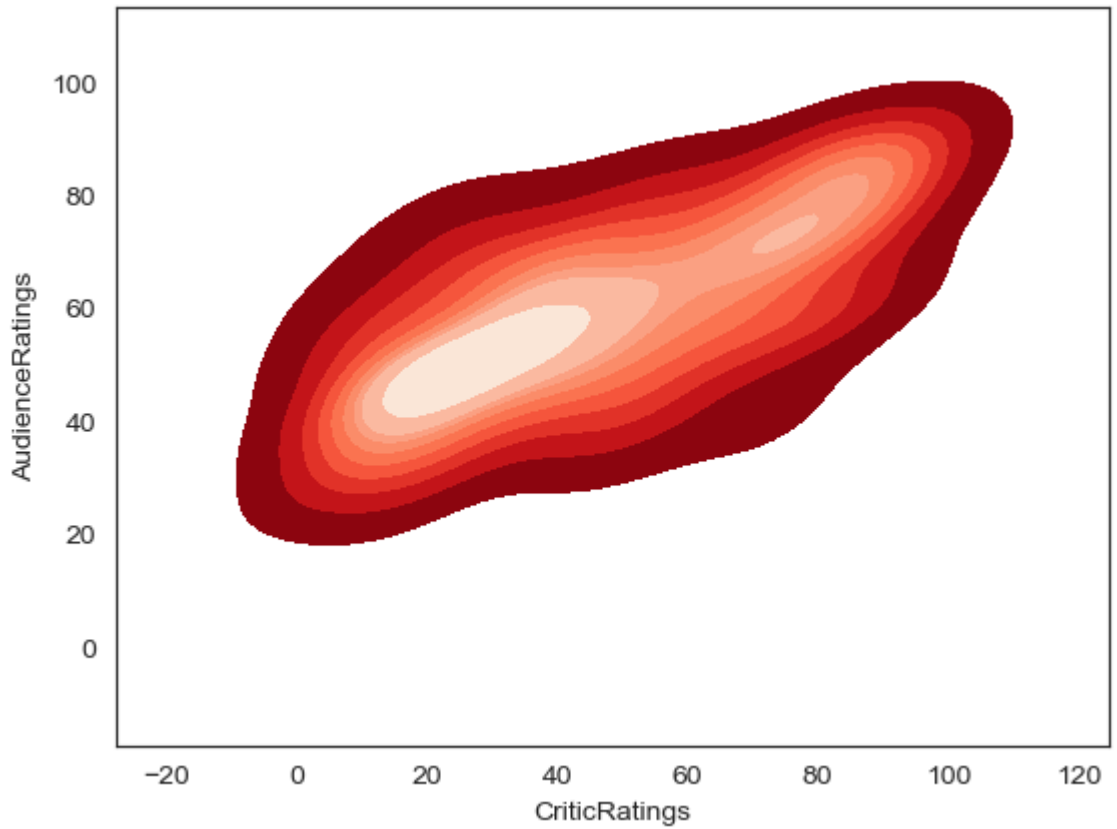
In [40]:
```
k1 = sns.kdeplot(x=movies.CriticRatings,y=movies.AudienceRatings)
```



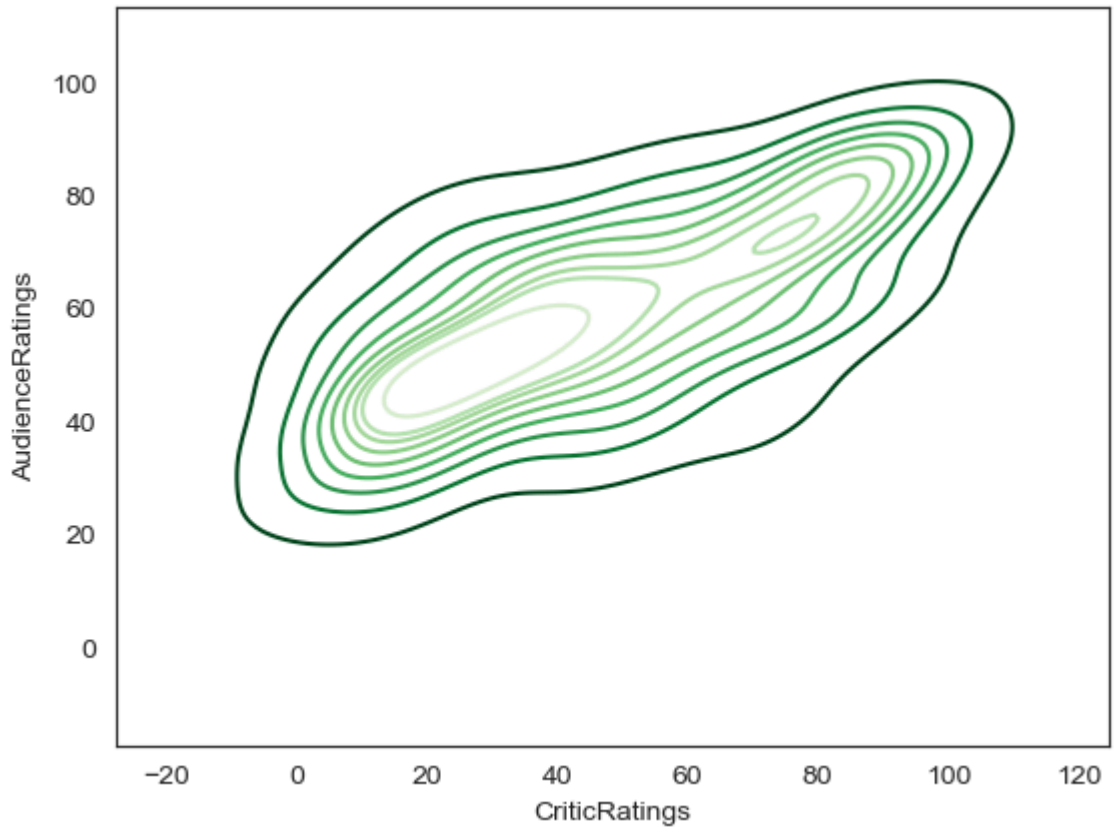
In [41]:
```
k1 = sns.kdeplot(x=movies.CriticRatings,y=movies.AudienceRatings,shade = True,shade_lowest=False,cmap='Reds')
```



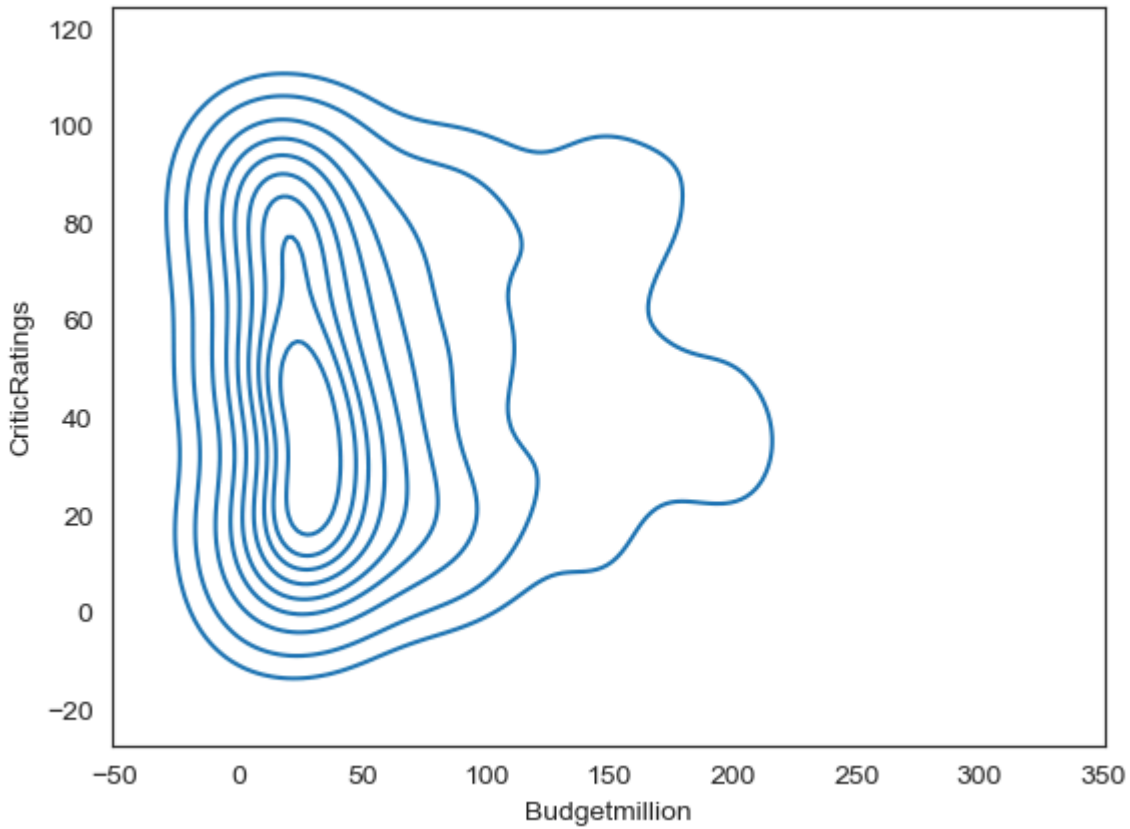
In [42]:
```
k1 = sns.kdeplot(x=movies.CriticRatings,y=movies.AudienceRatings,shade = True,shade_lowest=False,cmap='Reds_r')
```



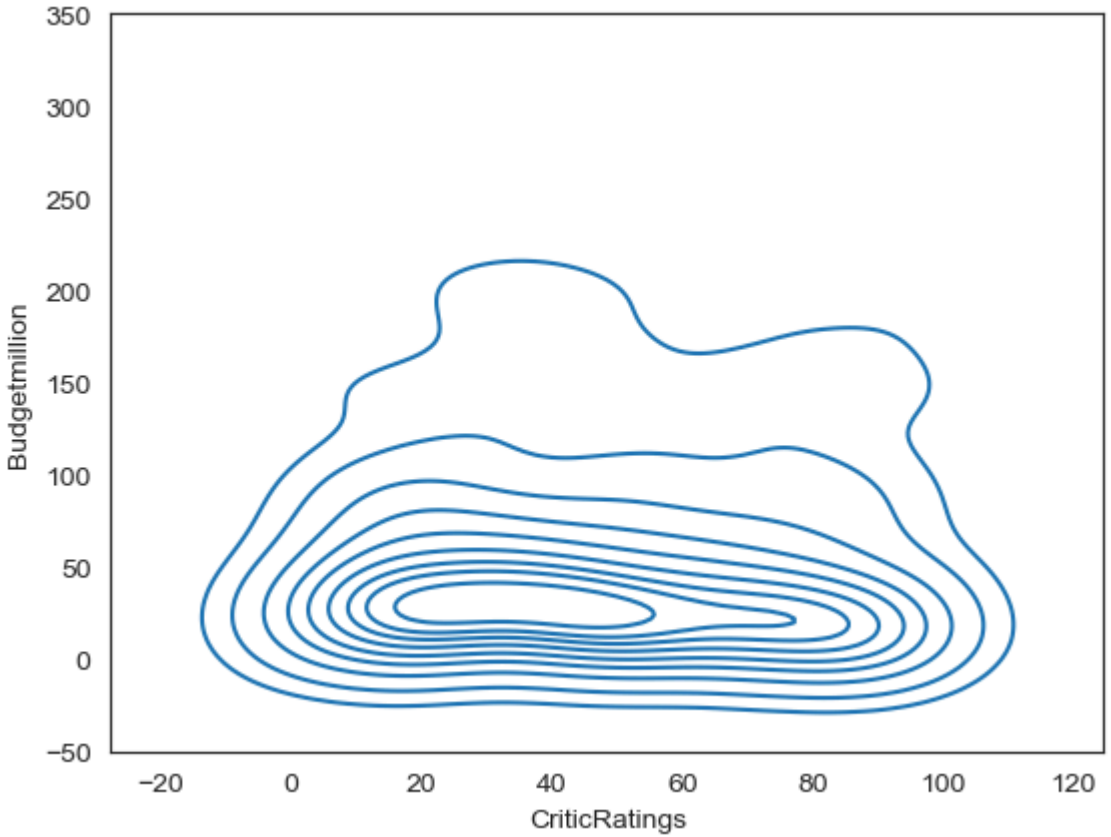
In [43]:
```
k2 = sns.kdeplot(x=movies.CriticRatings,y=movies.AudienceRatings,shade_lowest=False,cmap='Greens_r')
```



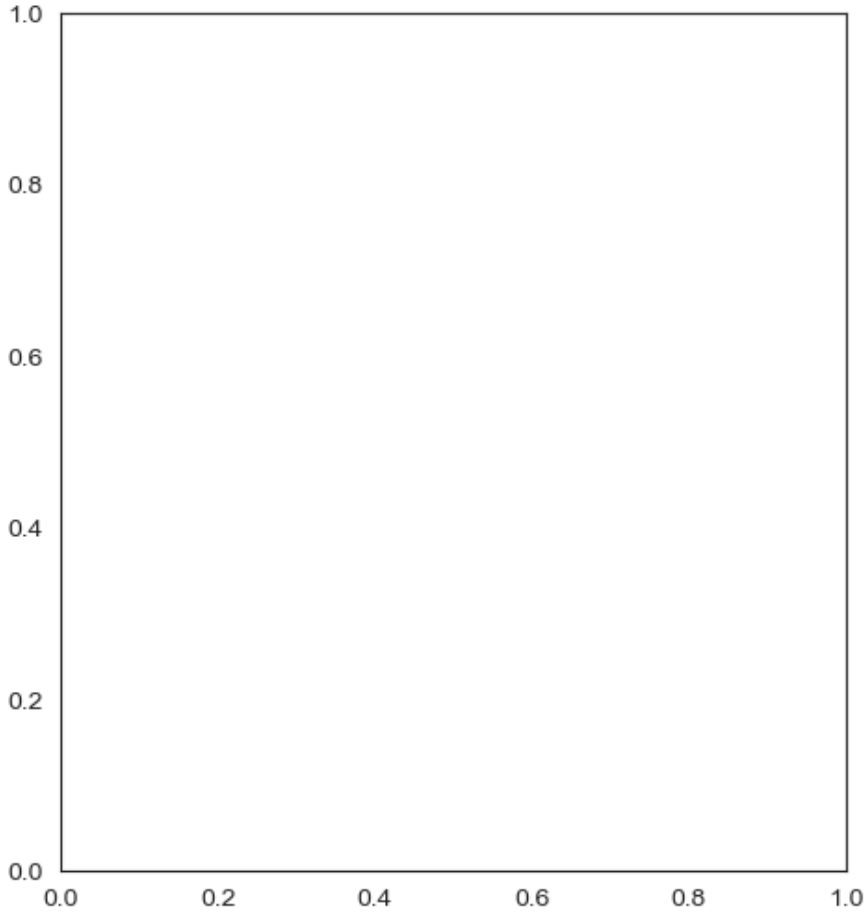
In [44]:
```
k2 = sns.kdeplot(x=movies.Budgetmillion,y=movies.CriticRatings)
```

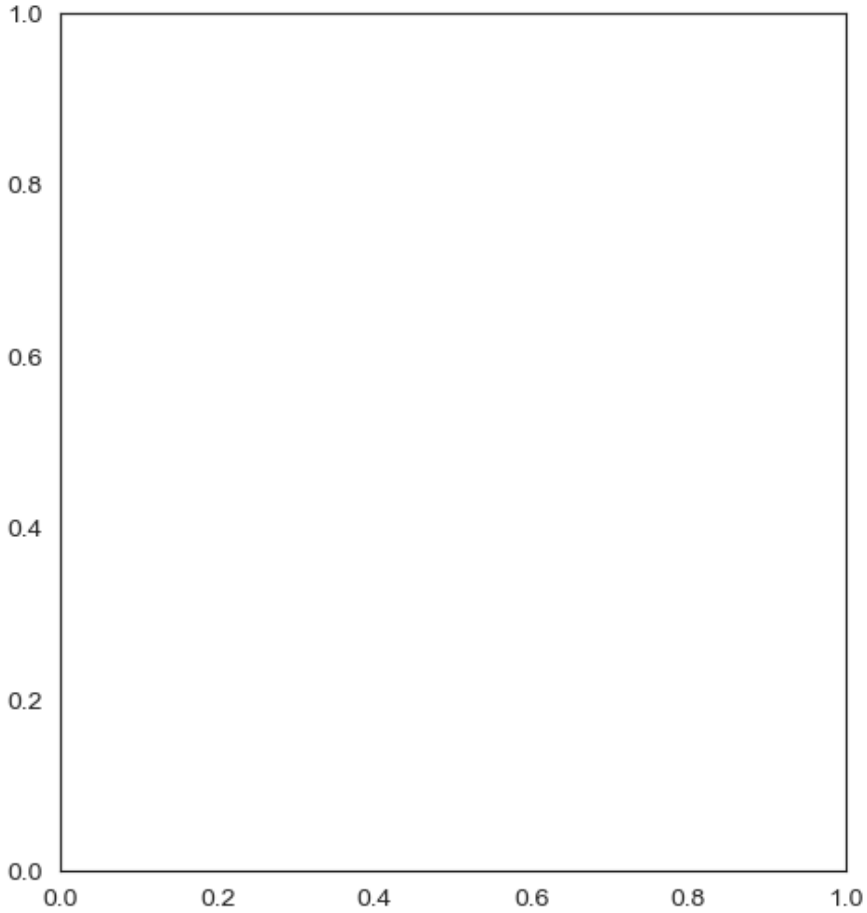


In [45]:
```
k2 = sns.kdeplot(y=movies.Budgetmillion,x=movies.CriticRatings)
```

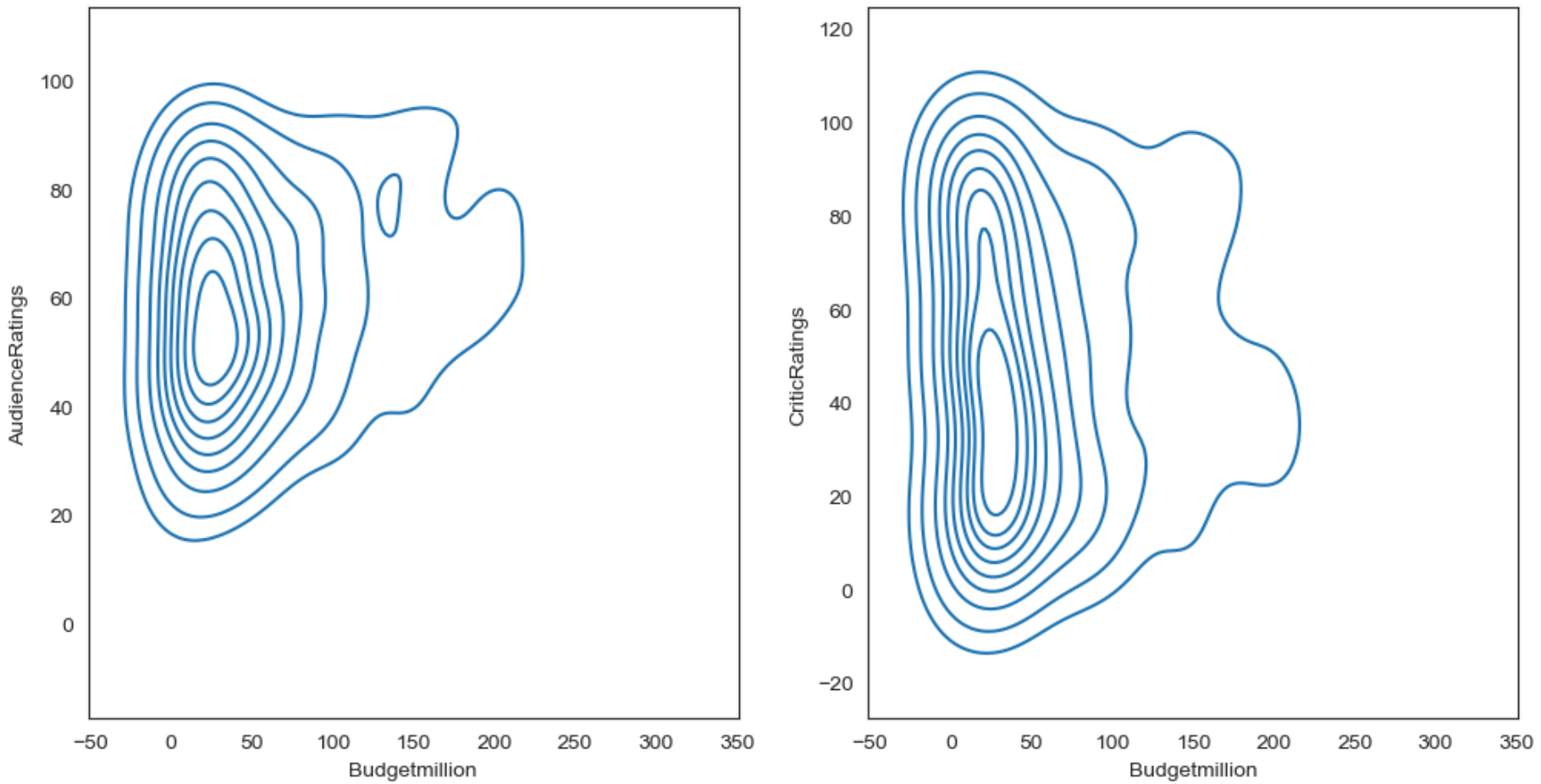


In [46]:
```
f,ax=plt.subplots(1,2, figsize=(12,6))
```





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

k1 = sns.kdeplot(x=movies.Budgetmillion,y=movies.AudienceRatings,ax=axes[0])
k2 = sns.kdeplot(x=movies.Budgetmillion,y=movies.CriticRatings,ax = axes[1])
```

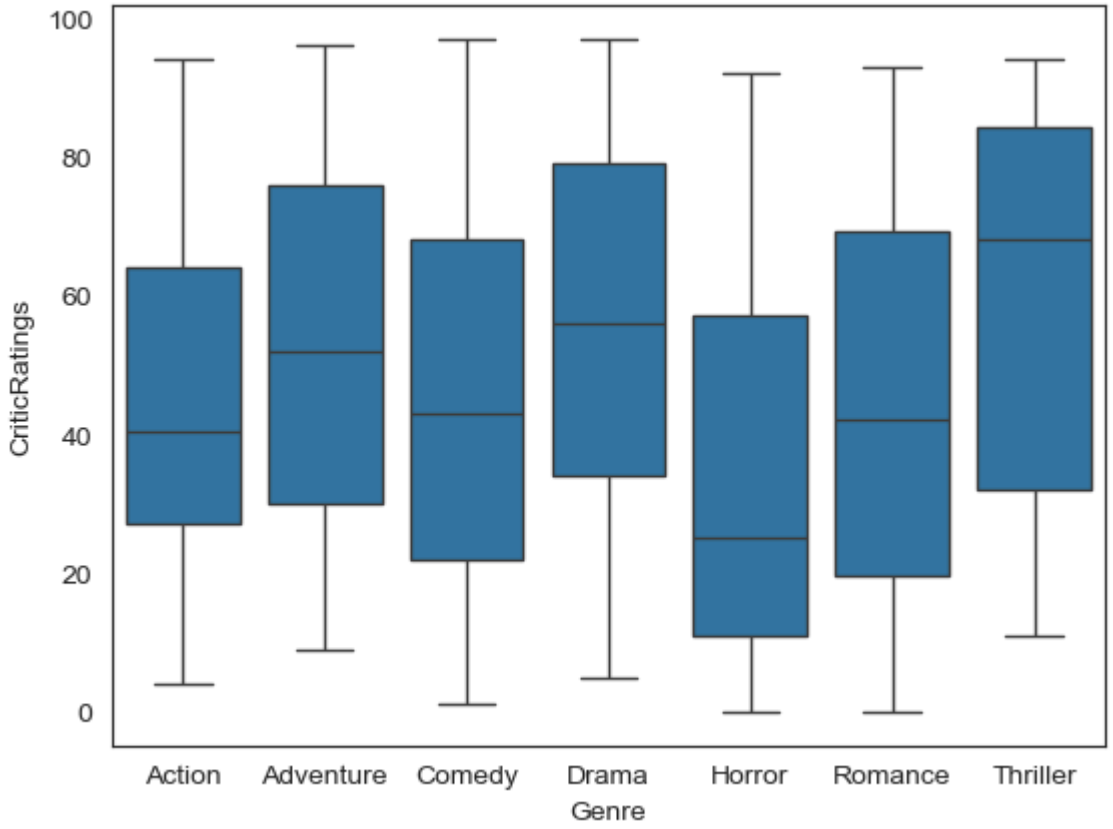


In [48]:
```
axes
```

Out[48]:
```
array([<Axes: xlabel='Budgetmillion', ylabel='AudienceRatings'>,
       <Axes: xlabel='Budgetmillion', ylabel='CriticRatings'>],
      dtype=object)
```

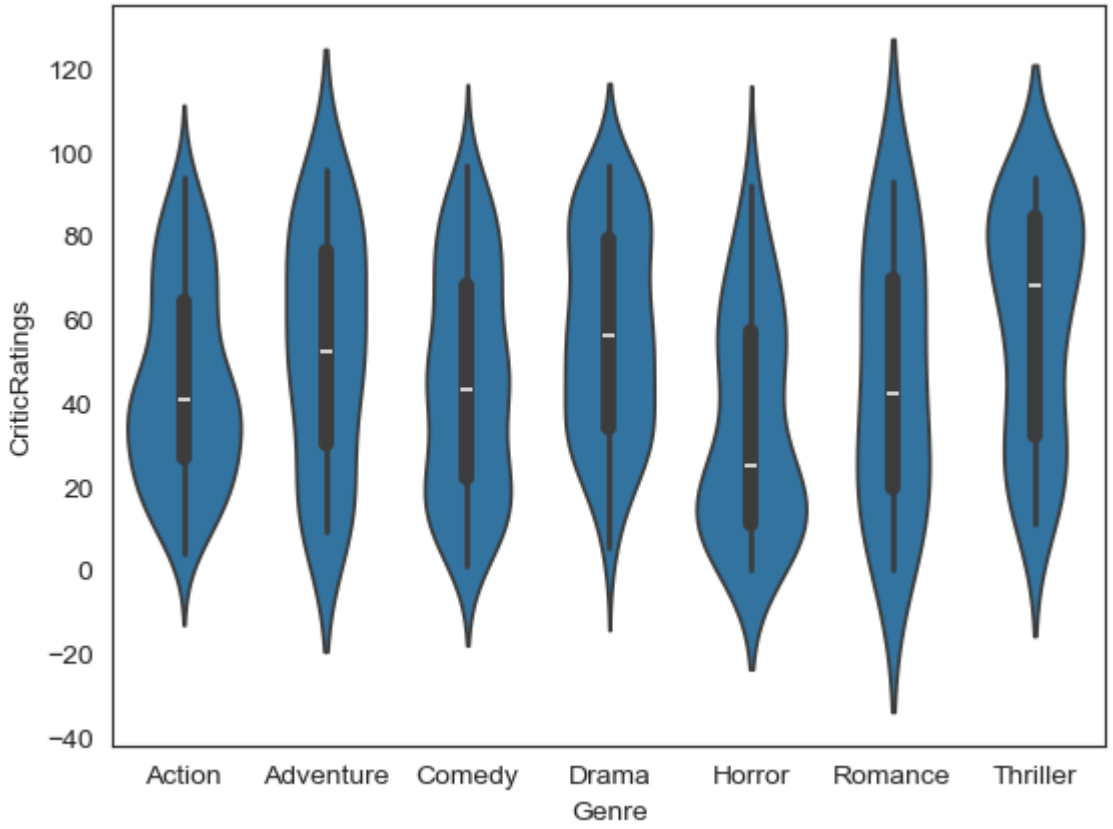
Boxplot

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

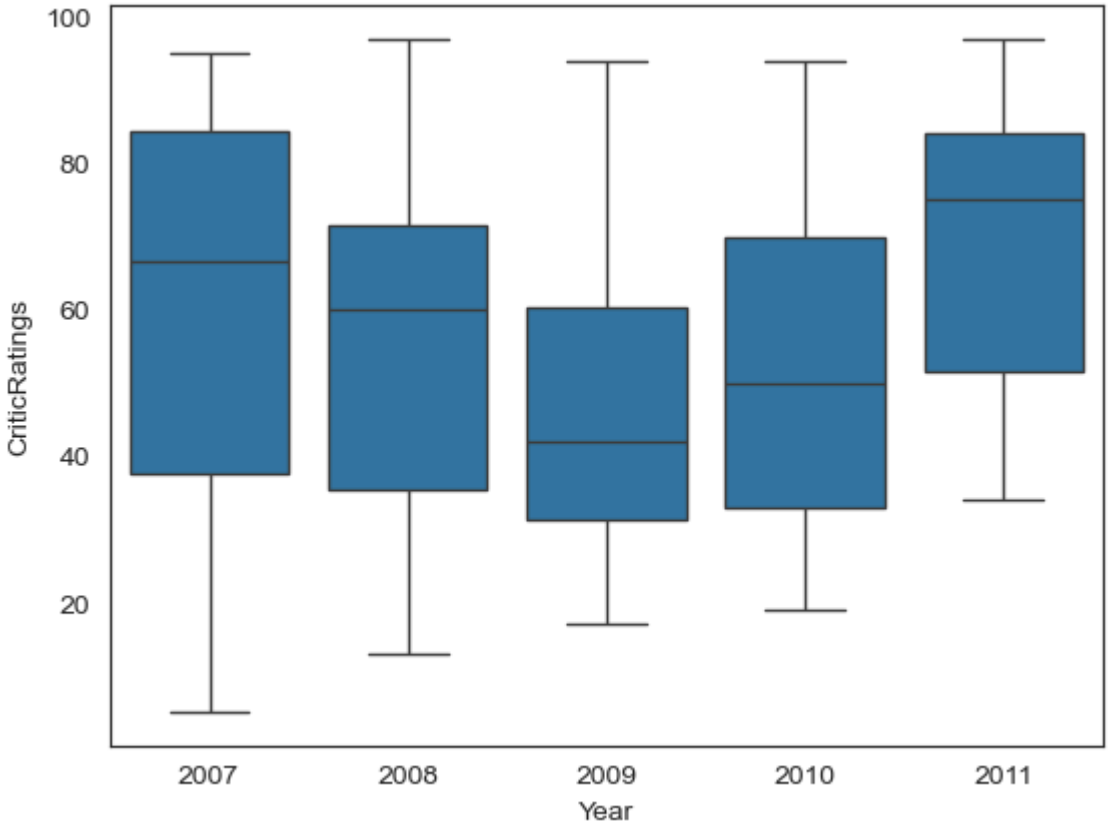


Violin plot

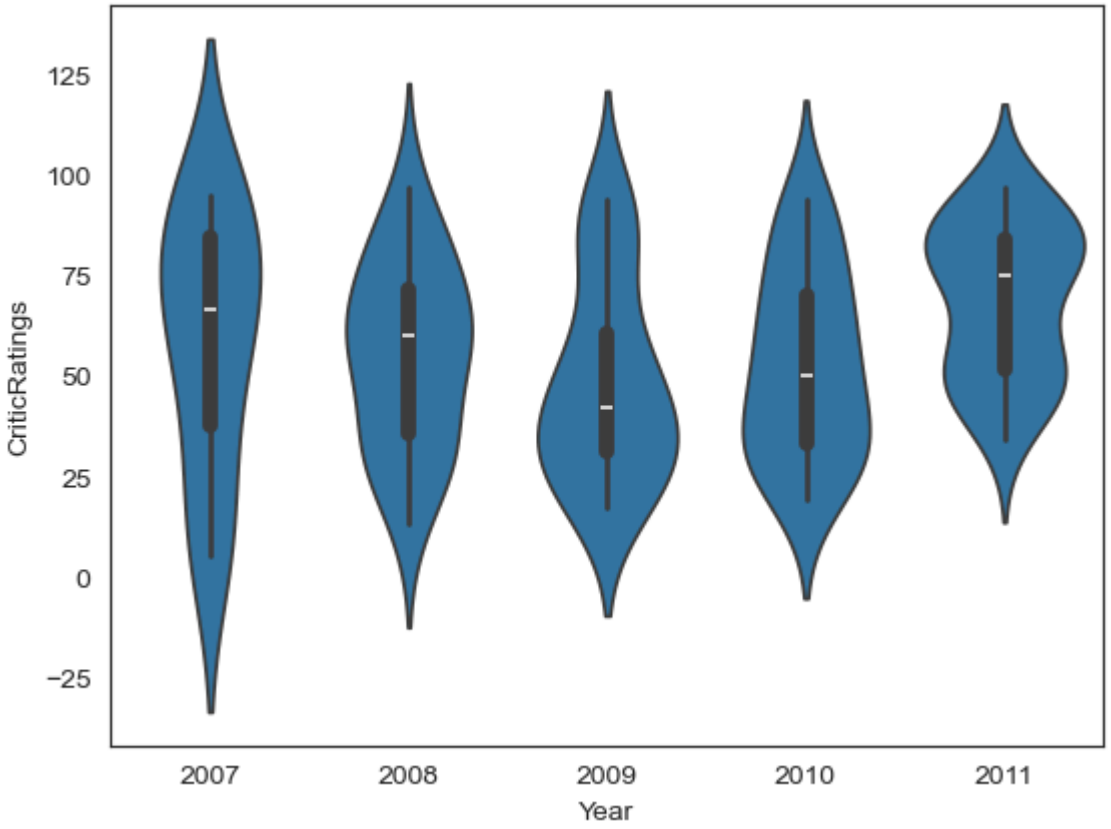
In [52]:
```
z = sns.violinplot(data=movies, x='Genre', y = 'CriticRatings')
```



In [53]:
```
w1 = sns.boxplot(data=movies[movies.Genre == 'Drama'], x='Year', y = 'CriticRatings', )
```



In [54]:
```
w1 = sns.violinplot(data=movies[movies.Genre == 'Drama'], x='Year', y = 'CriticRatings')
```



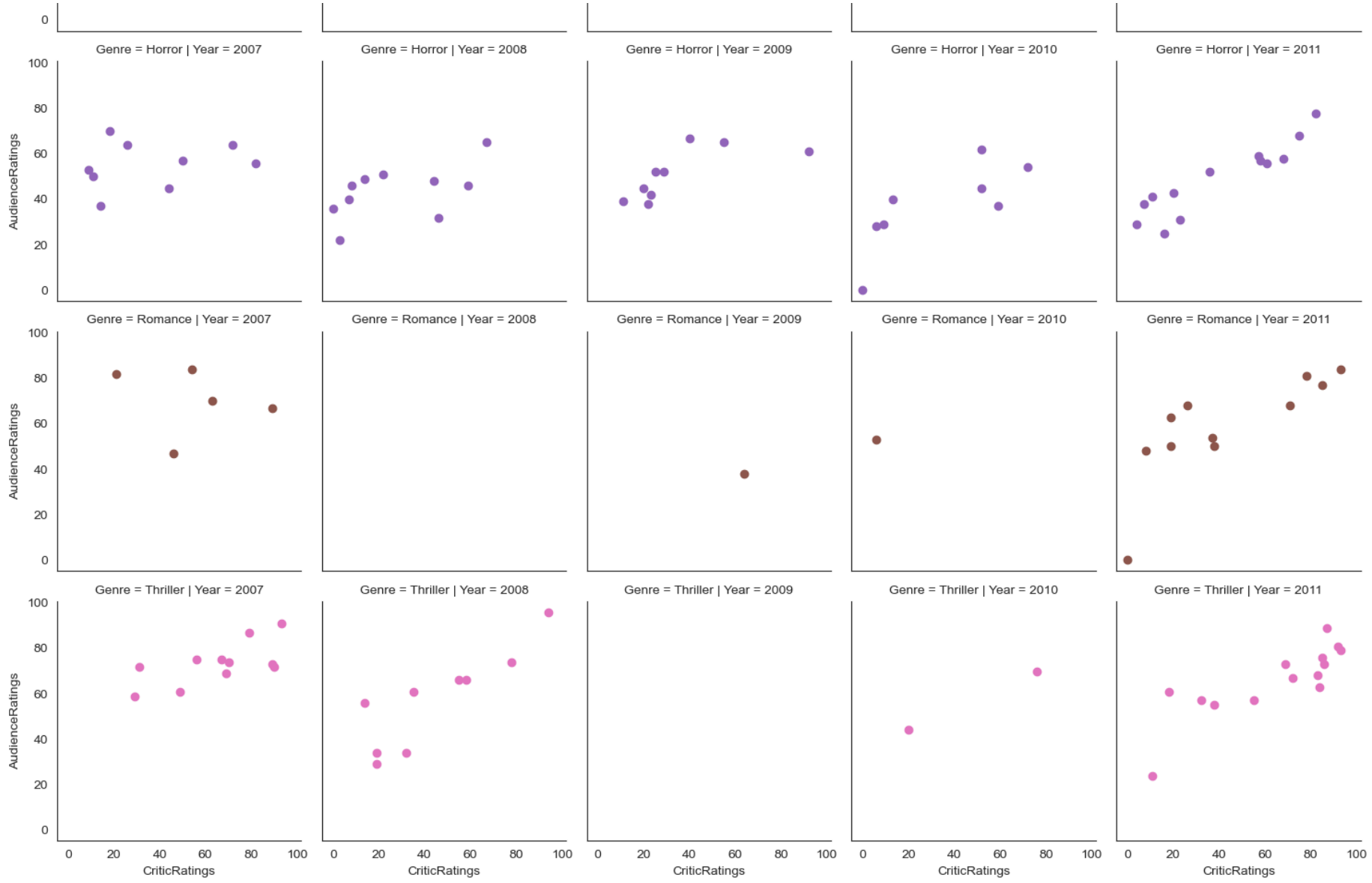
Createing a Facet grid

In [56]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre') #kind of subplots
```


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
1.0
0.8
0.6
0.4
0.2
0.0

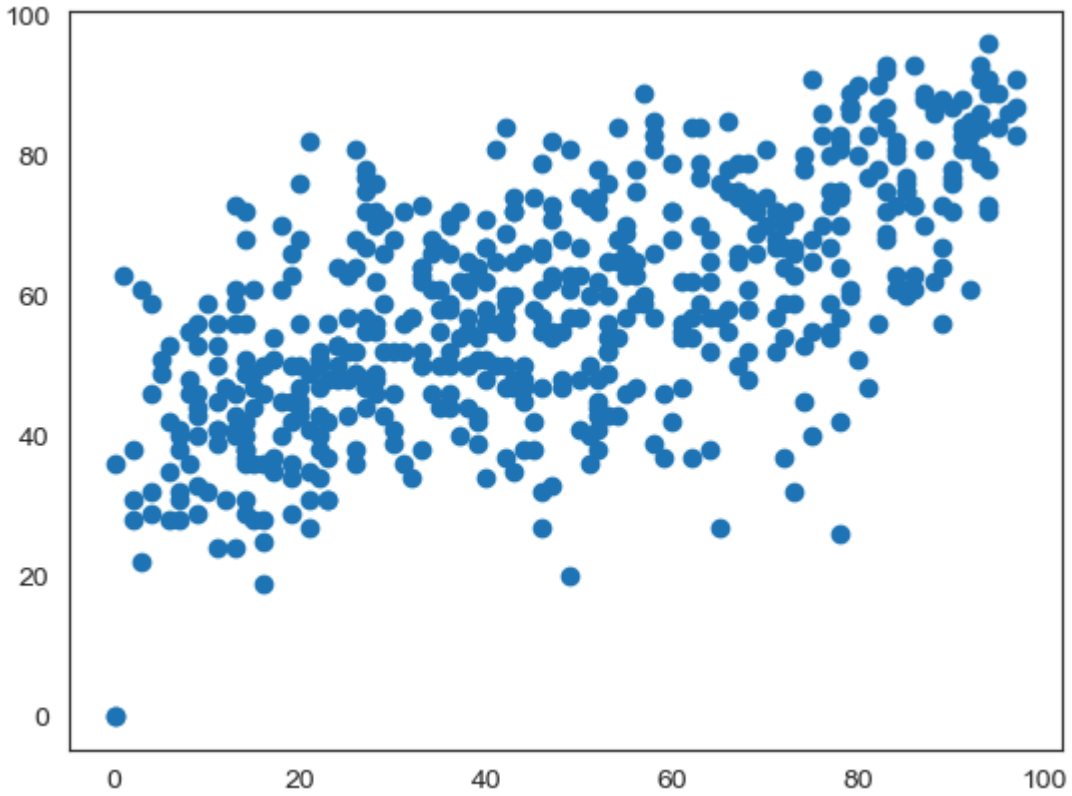
In [57]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.scatter, 'CriticRatings', 'AudienceRatings' ) #scatterplots are mapped in facetgrid
```


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
AudienceRatings
0
20
40
60
80
100



In [58]:
```
plt.scatter(movies.CriticRatings,movies.AudienceRatings)
```

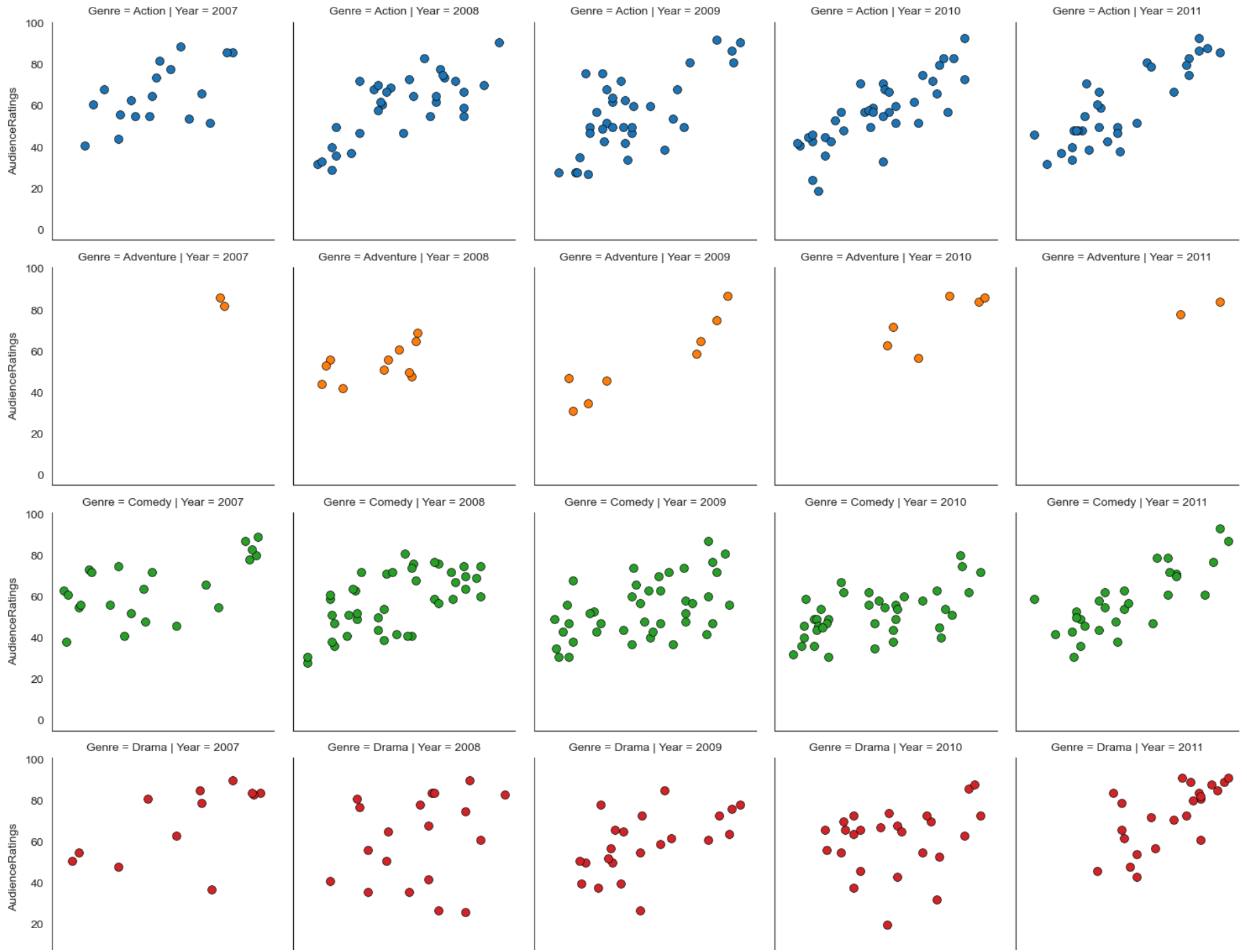
Out[58]:
```
<matplotlib.collections.PathCollection at 0x2666360d070>
```

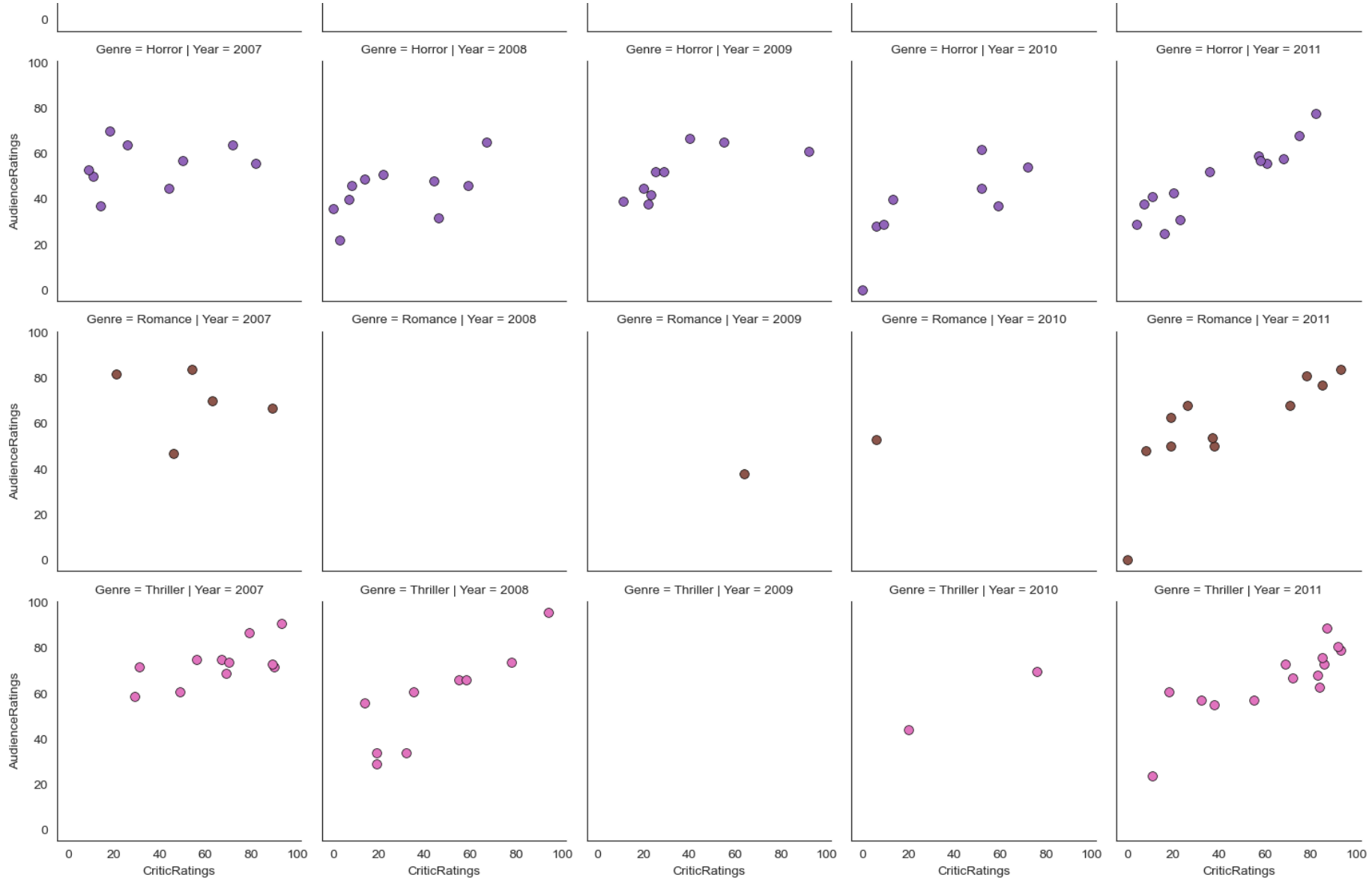


In [59]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.hist, 'Budgetmillion') #scatterplots are mapped in facetgrid
```


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
0
2
4
6
8
10
12
14
16

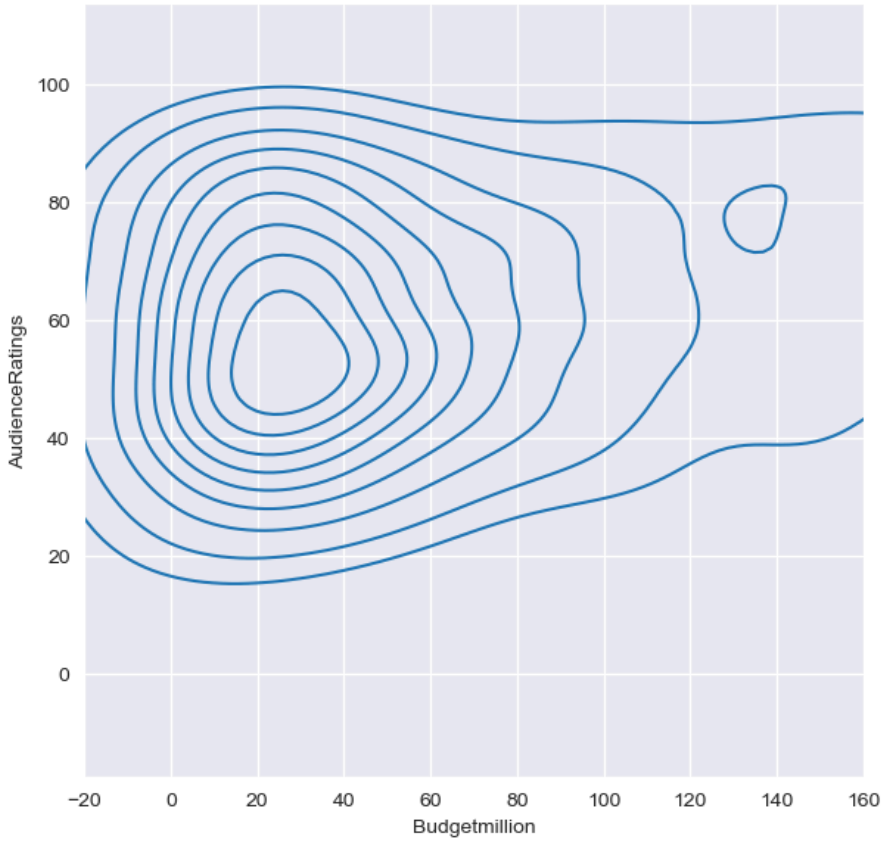
In [60]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
kws = dict(s=50, linewidth=0.5,edgecolor='black')
g = g.map(plt.scatter, 'CriticRatings', 'AudienceRatings',**kws )
```

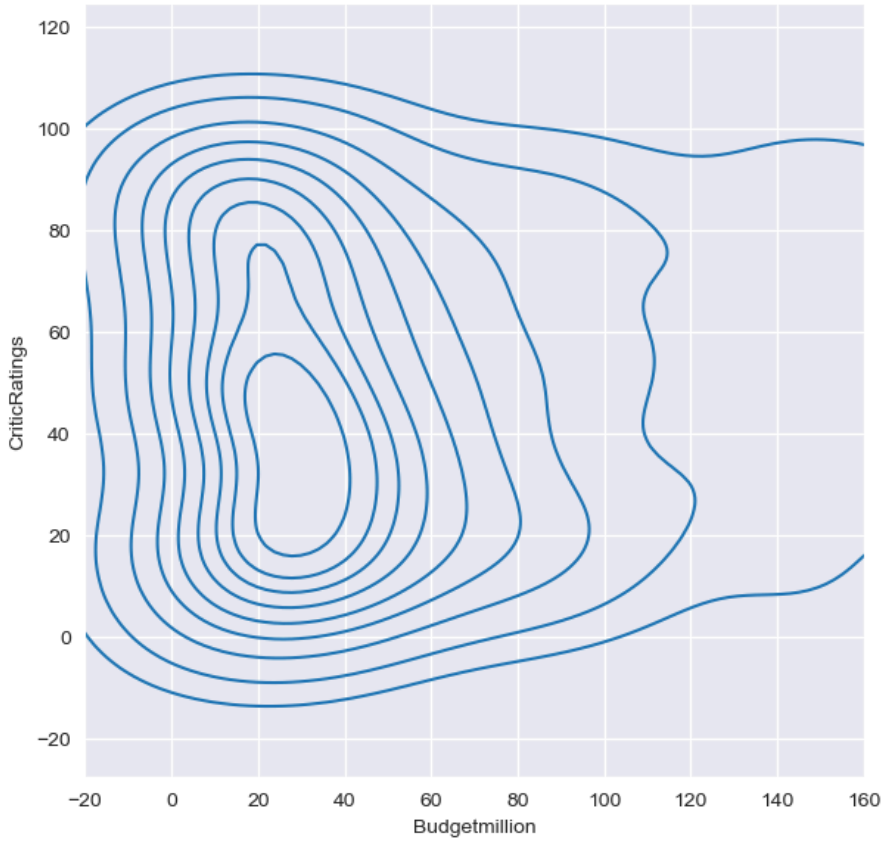

Genre = Action | Year = 2007
Genre = Action | Year = 2008
Genre = Action | Year = 2009
Genre = Action | Year = 2010
Genre = Action | Year = 2011
Genre = Adventure | Year = 2007
Genre = Adventure | Year = 2008
Genre = Adventure | Year = 2009
Genre = Adventure | Year = 2010
Genre = Adventure | Year = 2011
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
AudienceRatings
0
20
40
60
80
100

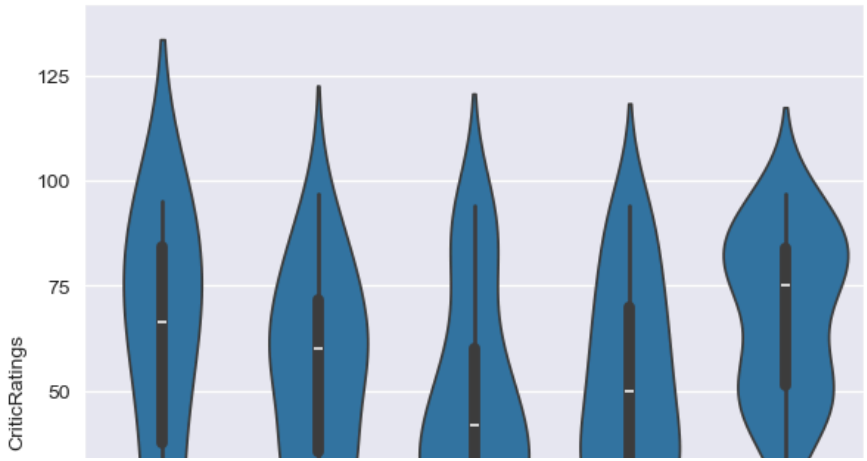


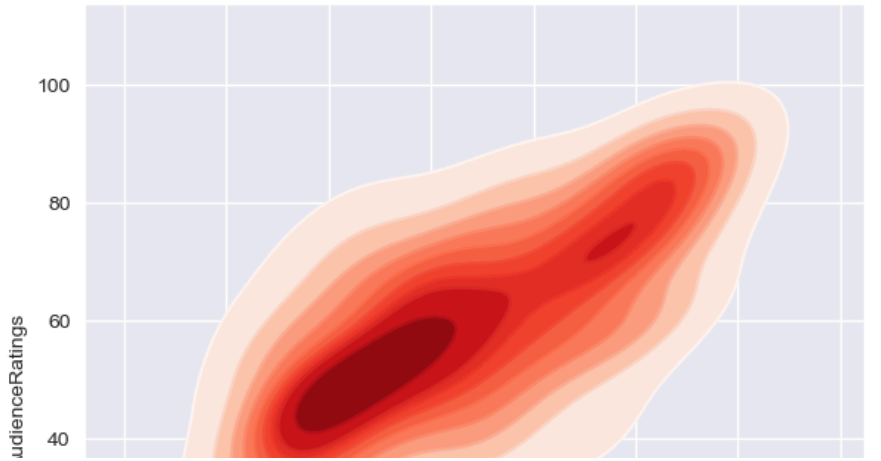
In [61]:
```
# python is not vectorize programming language
# Building dashboards (dashboard - combination of chats)


sns.set_style('darkgrid')
```

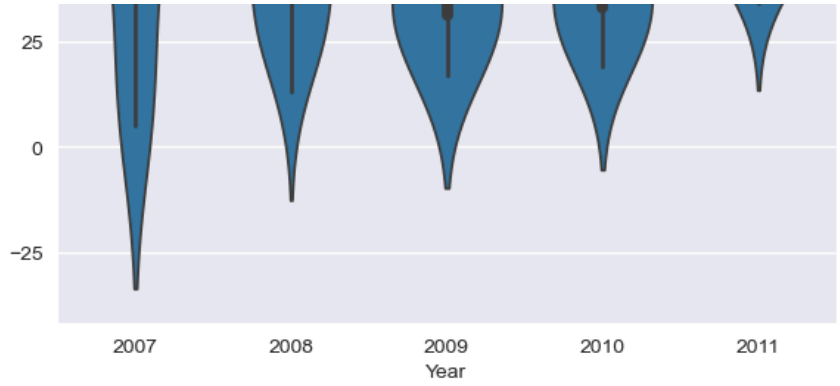
```
f, axes = plt.subplots (2,2, figsize = (15,15))

k1 = sns.kdeplot(x=movies.Budgetmillion,y=movies.AudienceRatings,ax=axes[0,0])
k2 = sns.kdeplot(x=movies.Budgetmillion,y=movies.CriticRatings,ax = axes[0,1])

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

z = sns.violinplot(data=movies[movies.Genre=='Drama'], x='Year', y = 'CriticRatings', ax=axes[1,0])

k4 = sns.kdeplot(x=movies.CriticRatings,y=movies.AudienceRatings,shade = True,shade_lowest=False,cmap='Reds',ax=axes[1,1])

k4b = sns.kdeplot(x=movies.CriticRatings,y= movies.AudienceRatings,cmap='Reds',ax = axes[1,1])

plt.show()
```

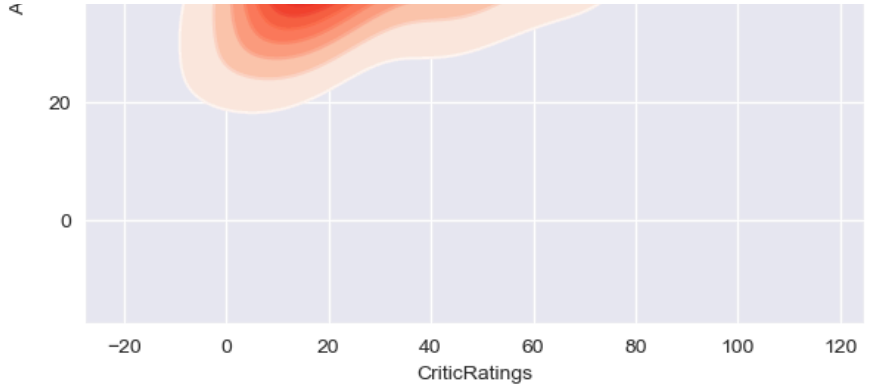

AudienceRatings
100
80
60
40
20
0
−20
0
20
40
60
80
100
120
140
160
Budgetmillion


CriticRatings
120
100
80
60
40
20
0
−20
−20
0
20
40
60
80
100
120
140
160
Budgetmillion


CriticRatings
125
100
75
50

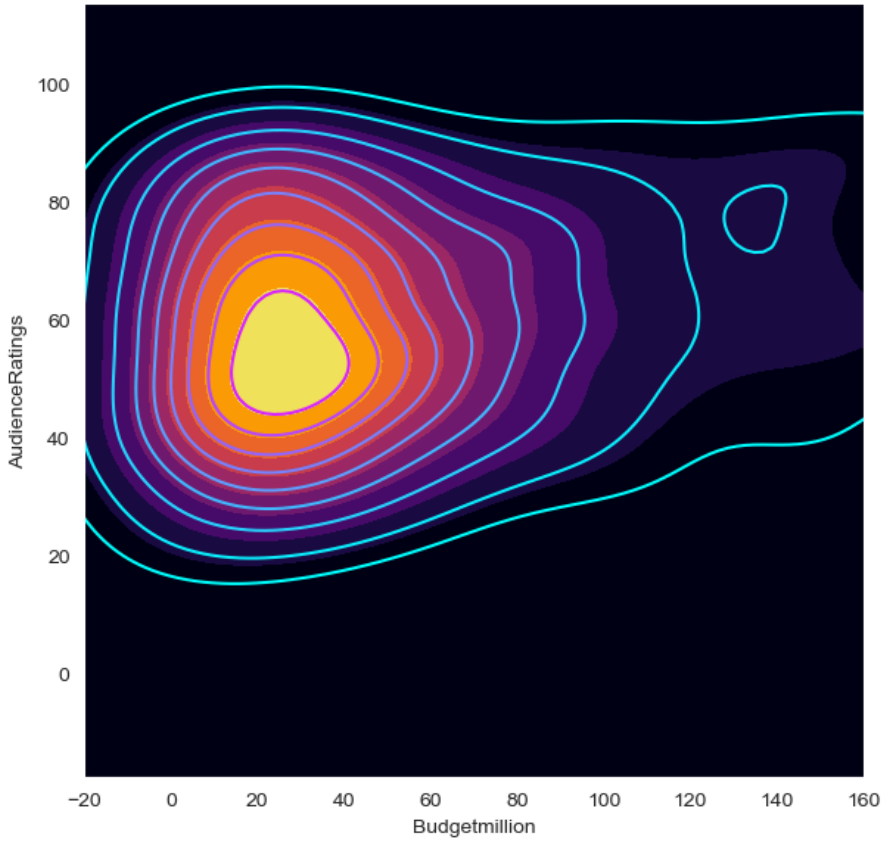

udienceRatings
100
80
60
40

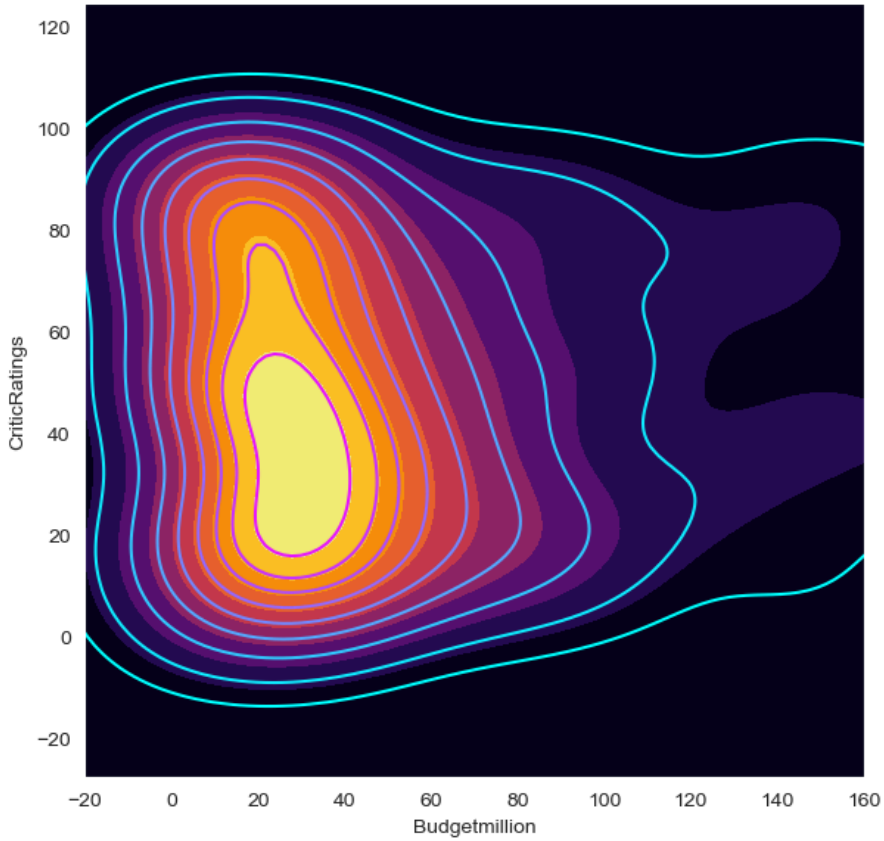


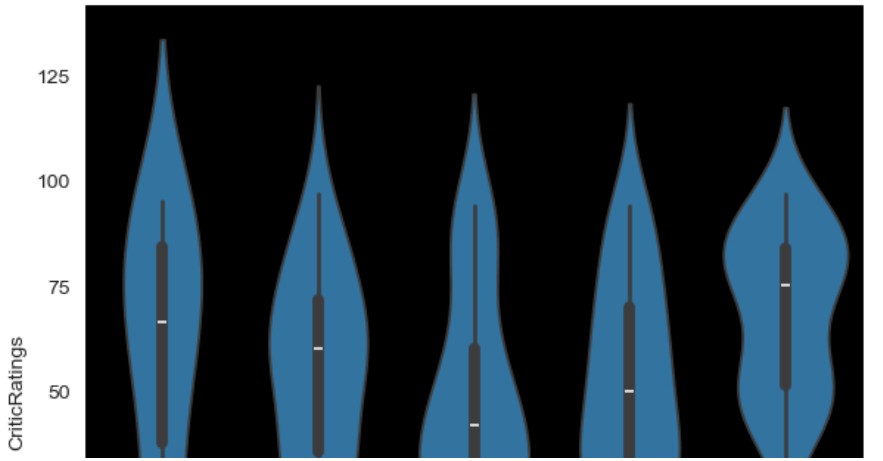


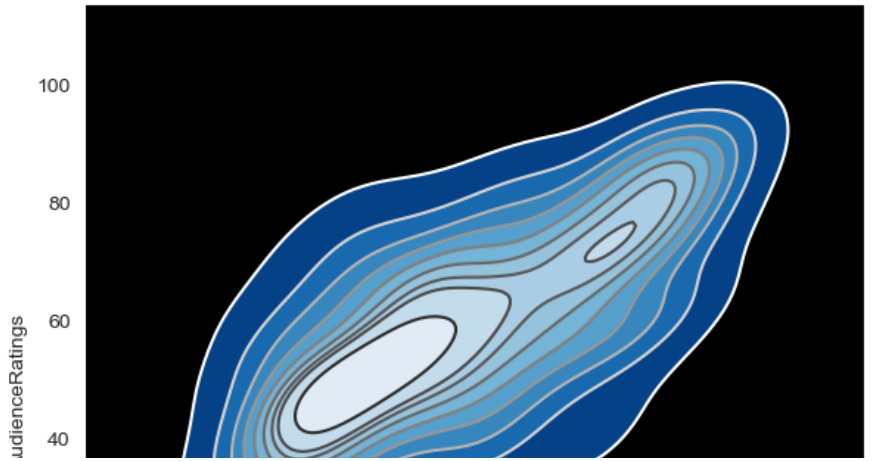
In [62]:

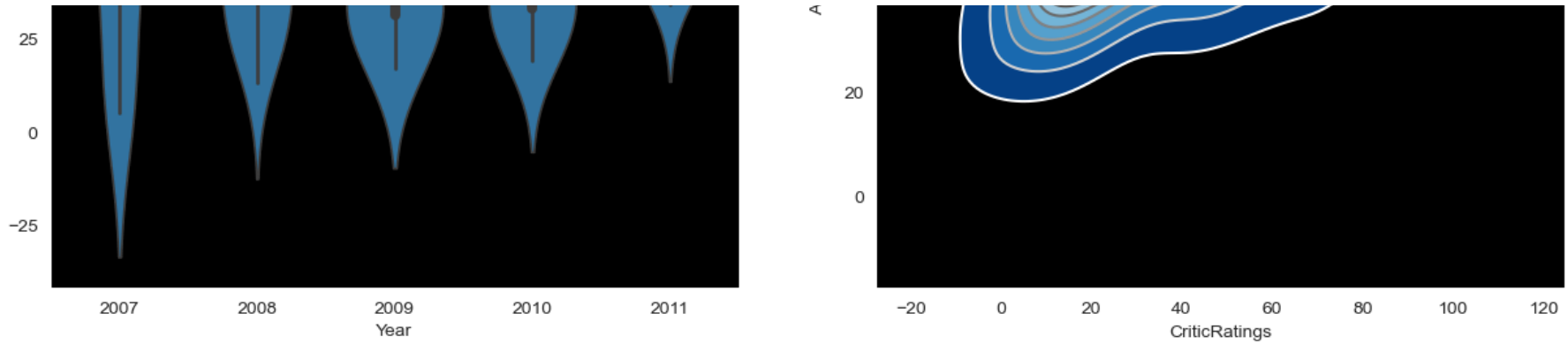
```
# How can you style your dashboard  using different color map
# python is not vectorize programming language
# Building dashboards (dashboard - combination of chats)

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

#plot [0,0]
k1 = sns.kdeplot(data=movies,x='Budgetmillion',y='AudienceRatings',shade = True, shade_lowest=True,cmap = 'inferno',ax = axes[
k1b = sns.kdeplot(data=movies,x='Budgetmillion',y='AudienceRatings',cmap = 'cool',ax = axes[0,0])

#plot [0,1]
k2 = sns.kdeplot(data=movies,x='Budgetmillion',y='CriticRatings',shade=True, shade_lowest=True, cmap='inferno',ax = axes[0,1])
k2b = sns.kdeplot(data=movies,x='Budgetmillion',y='CriticRatings',cmap = 'cool', ax = axes[0,1])

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

#plot[1,1]
k4 = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',shade = True,shade_lowest=False,cmap='Blues_r',ax=axes[1,1]
k4b = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',cmap='gist_gray_r',ax = axes[1,1])

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

plt.show()
```


AudienceRatings
100
80
60
40
20
0
−20
0
20
40
60
80
100
120
140
160
Budgetmillion


CriticRatings
120
100
80
60
40
20
0
−20
−20
0
20
40
60
80
100
120
140
160
Budgetmillion


CriticRatings
125
100
75
50


udienceRatings
100
80
60
40



Final discussion what we learn so far - 1> category datatype in python 2> jointplots 3> histogram 4> stacked histograms 5> Kde plot 6> subplot 7> violin plots 8> Factet grid 9> Building dashboards