In [1]:

```
# Movie Rating analytics

import pandas as pd
```

In [2]:

```
import os
```

In [3]:

```
os.getcwd() # if you want to change the working directory
```

Out[3]:

```
'C:\\Users\\rushi'
```

In [4]:

```
movies = pd.read_csv(r'C:\Users\rushi\OneDrive\Documents\DS training notes\10th,11th\10th
```

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','AudienceRating','BudgetMillions','Year'
```

In [11]:

```
movies.head() # Removed noise characters
```

Out[11]:

	Film	Genre	CriticRatings	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 [12]:

```
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   CriticRatings   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 [13]:

```
movies.describe() # if you look at the year the data type is int but when you look at the
# we have to change to categroy type
# also from object datatype we will convert to category datatypes
```

Out[13]:

	CriticRatings	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 [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.head()
```

Out[15]:

	Film	Genre	CriticRatings	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 [16]:

```
movies.info() # now the same thing we will change genre to category & year 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    object
 1   Genre           559 non-null    object
 2   CriticRatings   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 [17]:

```
movies.Genre = movies.Genre.astype('category')
movies.Year = movies.Year.astype('category')
```

In [18]:

```
movies.Genre
```

Out[18]:

```
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', 'Horro
r', 'Romance', 'Thriller']
```

In [19]:

```
movies.Year # is it real no. year you can take average,min,max but out come have no meani
```

Out[19]:

```
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 [20]:

```
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    category
 2   CriticRatings   559 non-null    int64   
 3   AudienceRating  559 non-null    int64   
 4   BudgetMillions  559 non-null    int64   
 5   Year            559 non-null    category
dtypes: category(2), int64(3), object(1)
memory usage: 19.2+ KB
```

In [21]:

```
movies.Film = movies.Film.astype('category')
```

In [22]:

```
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   CriticRatings   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 [23]:

```
movies.Genre.cat.categories
```

Out[23]:

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

In [24]:

```
movies.describe()
```

Out[24]:

	CriticRatings	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 [25]:

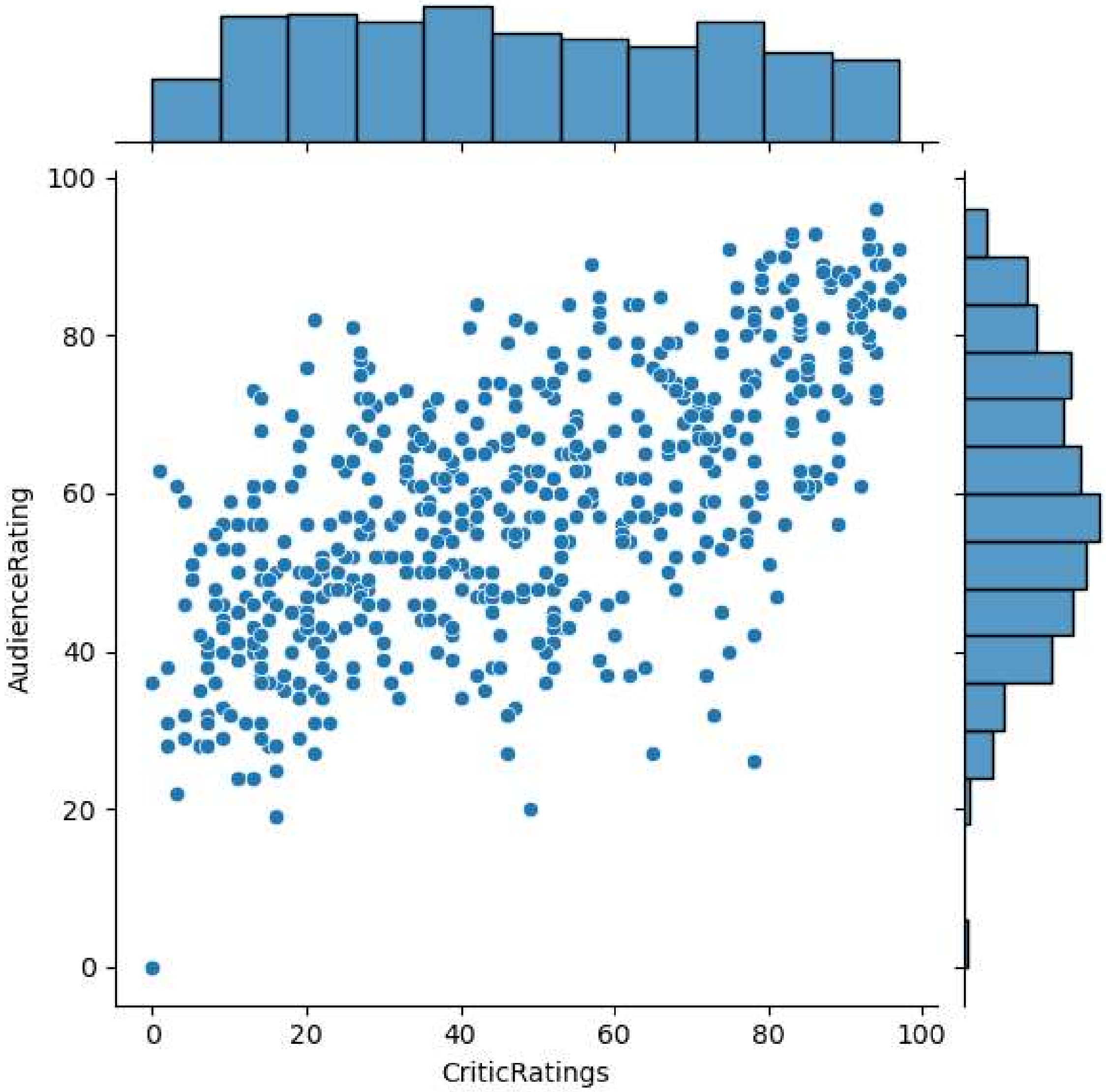
```
# How to working with joint plots

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 audiene & critics

also if you look up you can find the uniform distribution (critics)and normal distriution (audience)

In [26]:

```
j = sns.jointplot( data = movies, x = 'CriticRatings', y = 'AudienceRating')
```



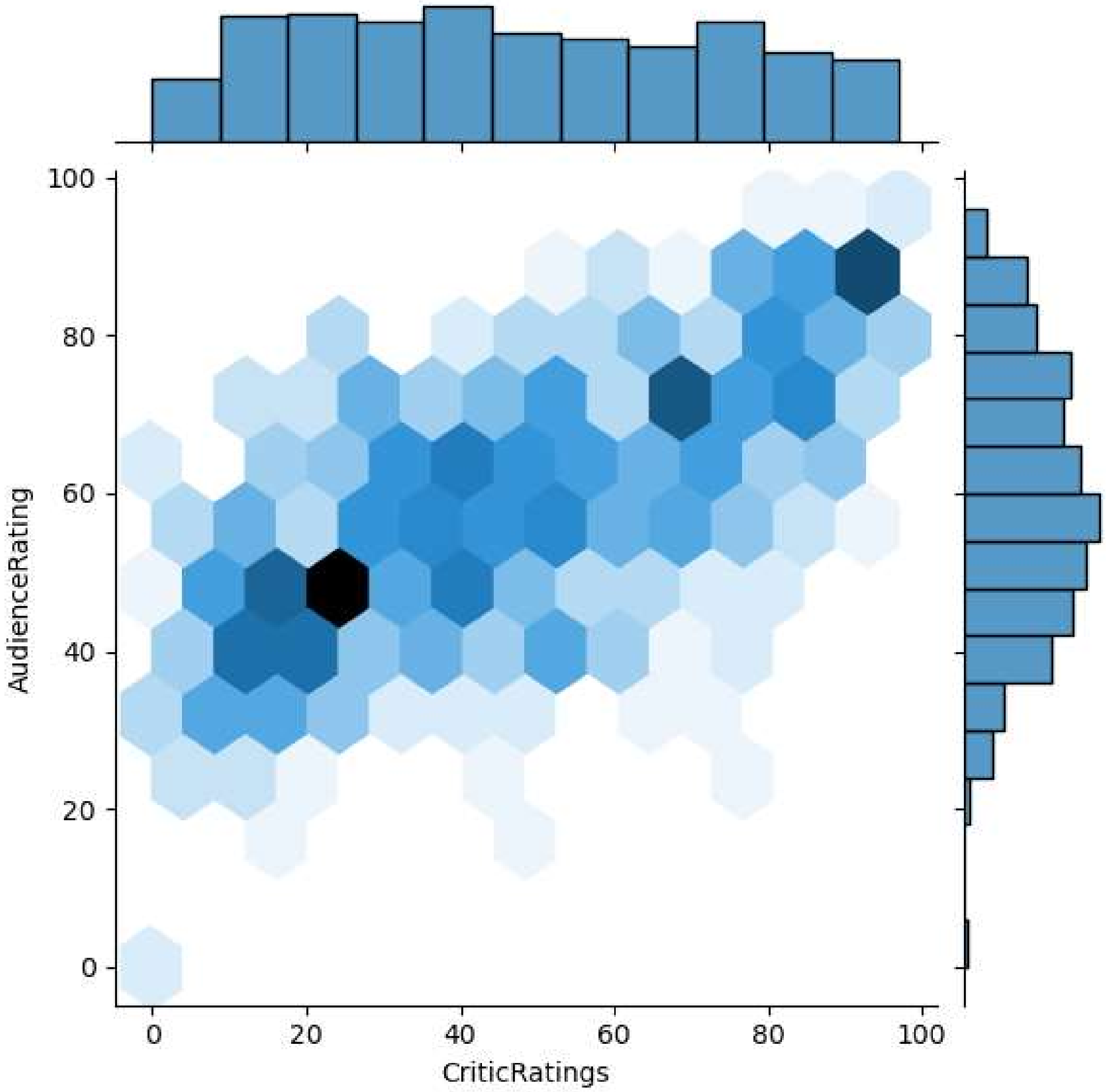
Audience rating is more dominant then critics rating

Based on this we find out as most people are most liklihood to watch audience rating & less likely to wathc critics rating

let me explain the excel - if you filter audience rating & critic rating. critic rating has very low values compare

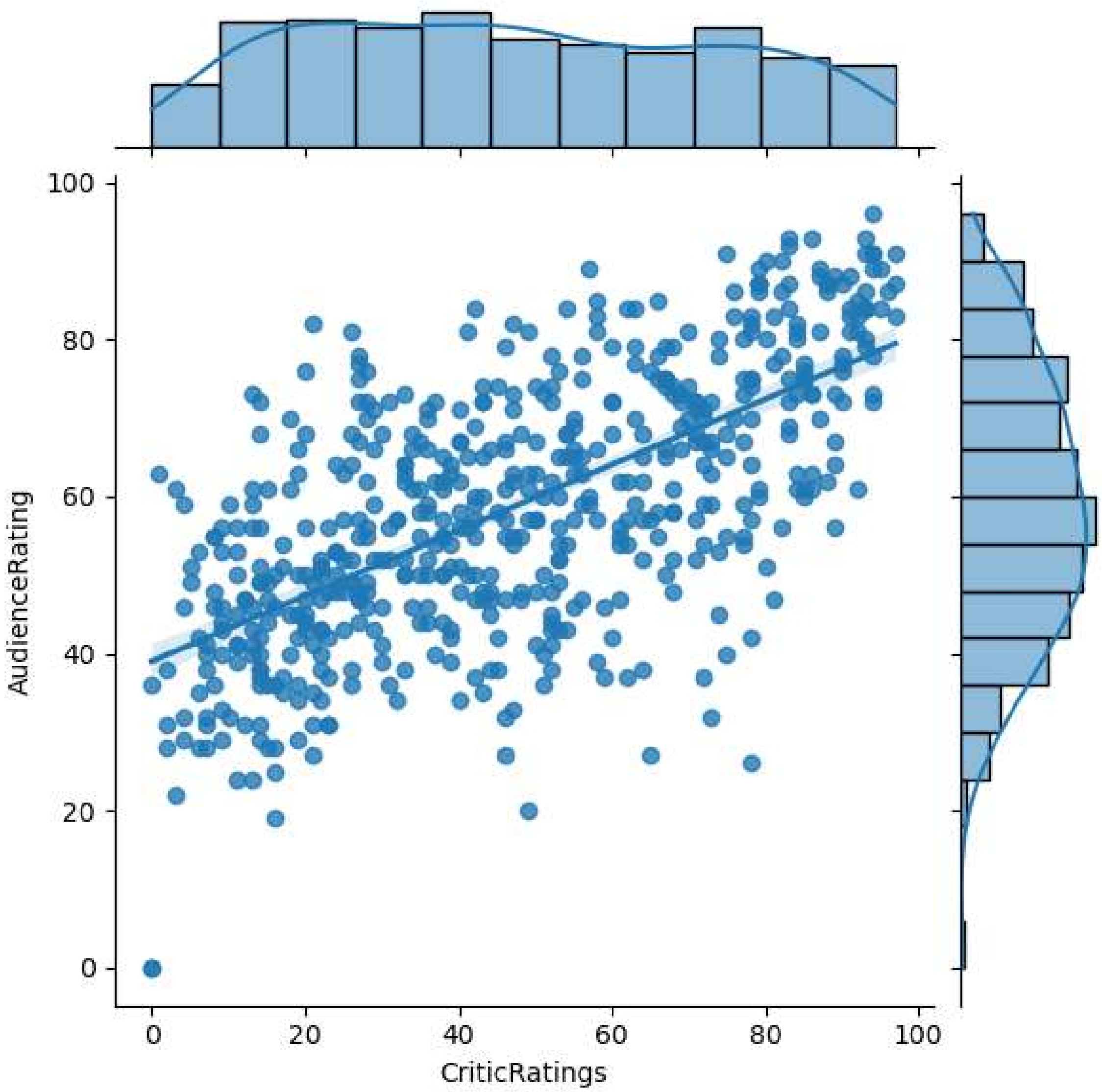
In [27]:

```
j = sns.jointplot( data = movies, x = 'CriticRatings', y = 'AudienceRating', kind = 'hex'
```



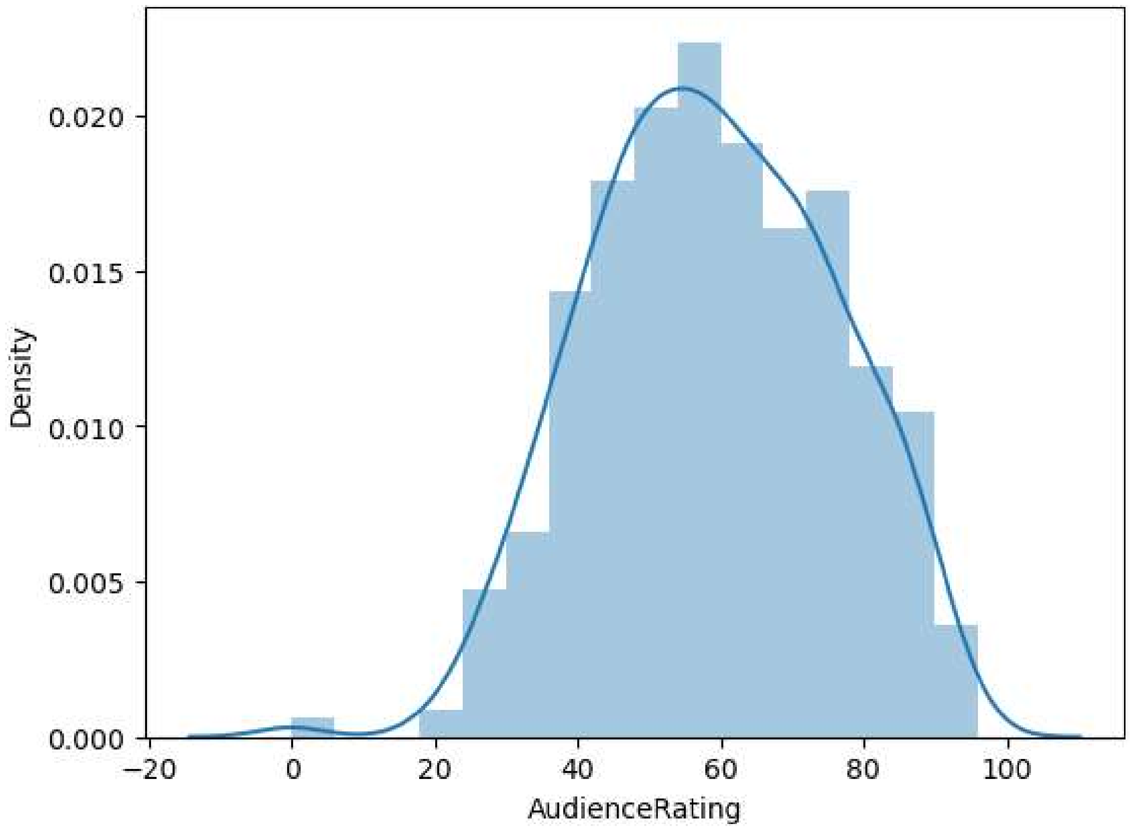
In [28]:

```
j = sns.jointplot( data = movies, x = 'CriticRatings', y = 'AudienceRating', kind = 'reg'
```



In [29]:

```
# Histograms
# <<< chat1

m1 = sns.distplot(movies.AudienceRating) # y-axis is automatically generated by seaborn g
```

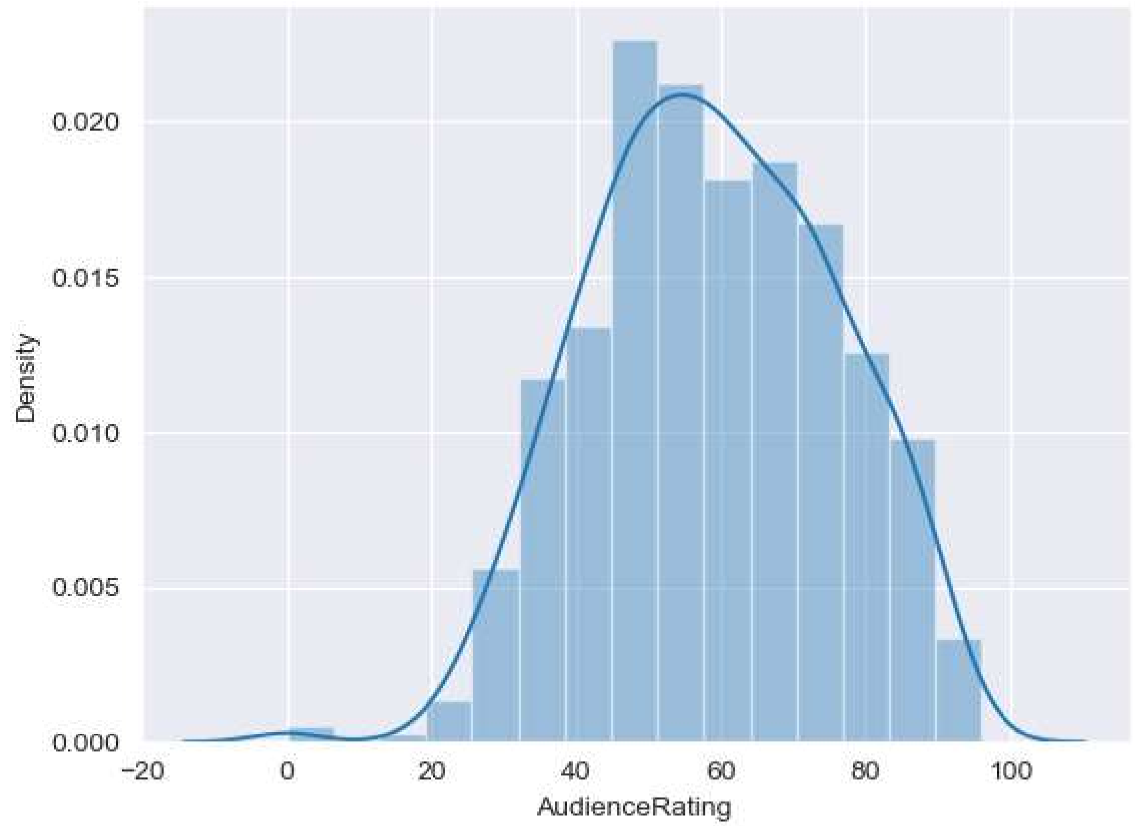


In [30]:

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

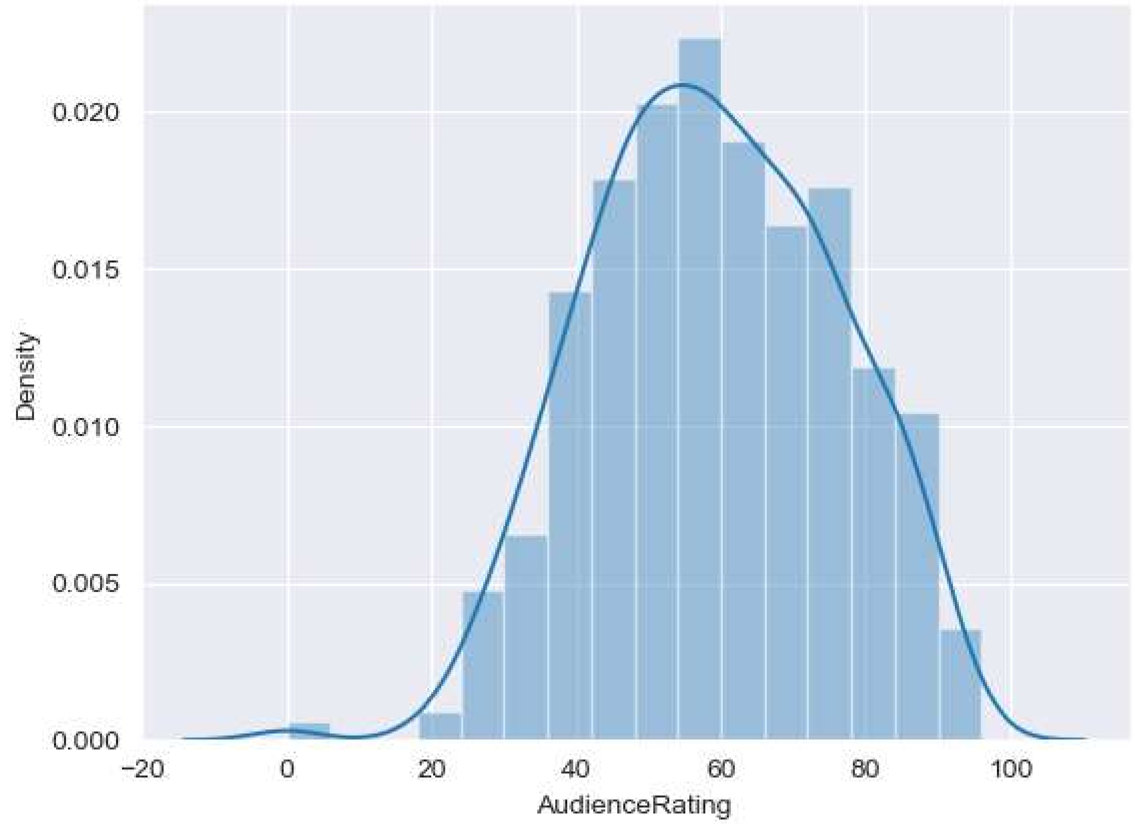
In [31]:

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



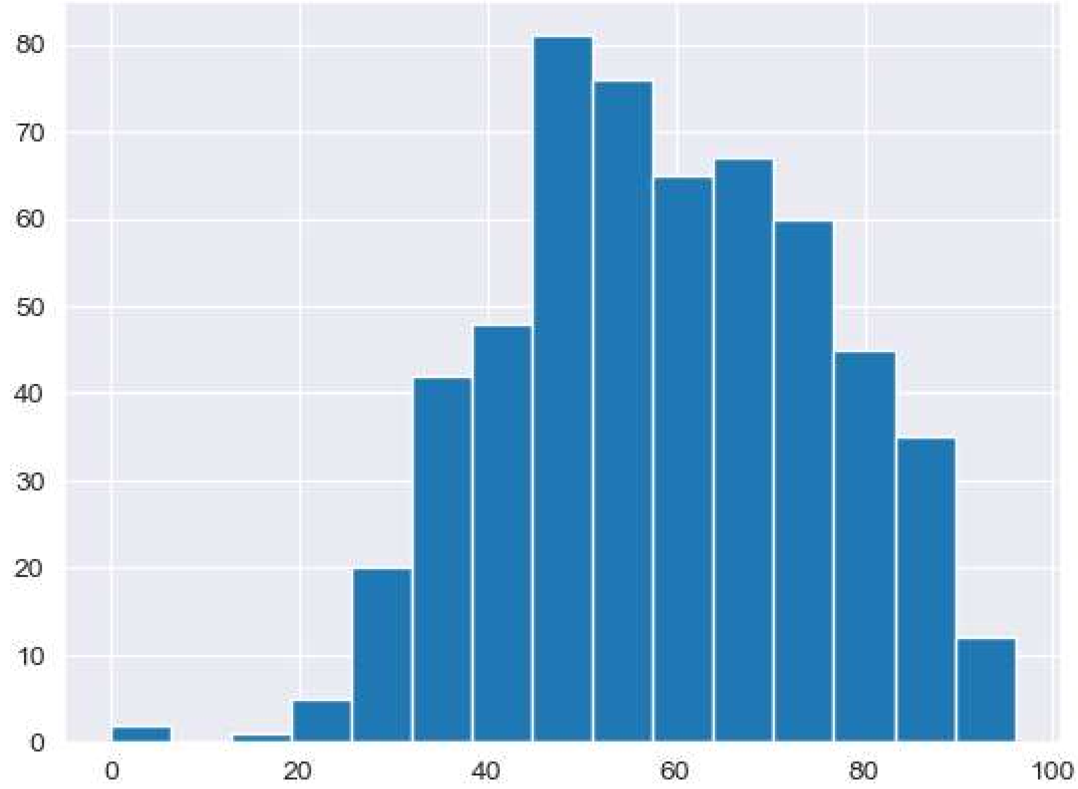
In [32]:

```
m2 = sns.distplot(movies.AudienceRating)
```



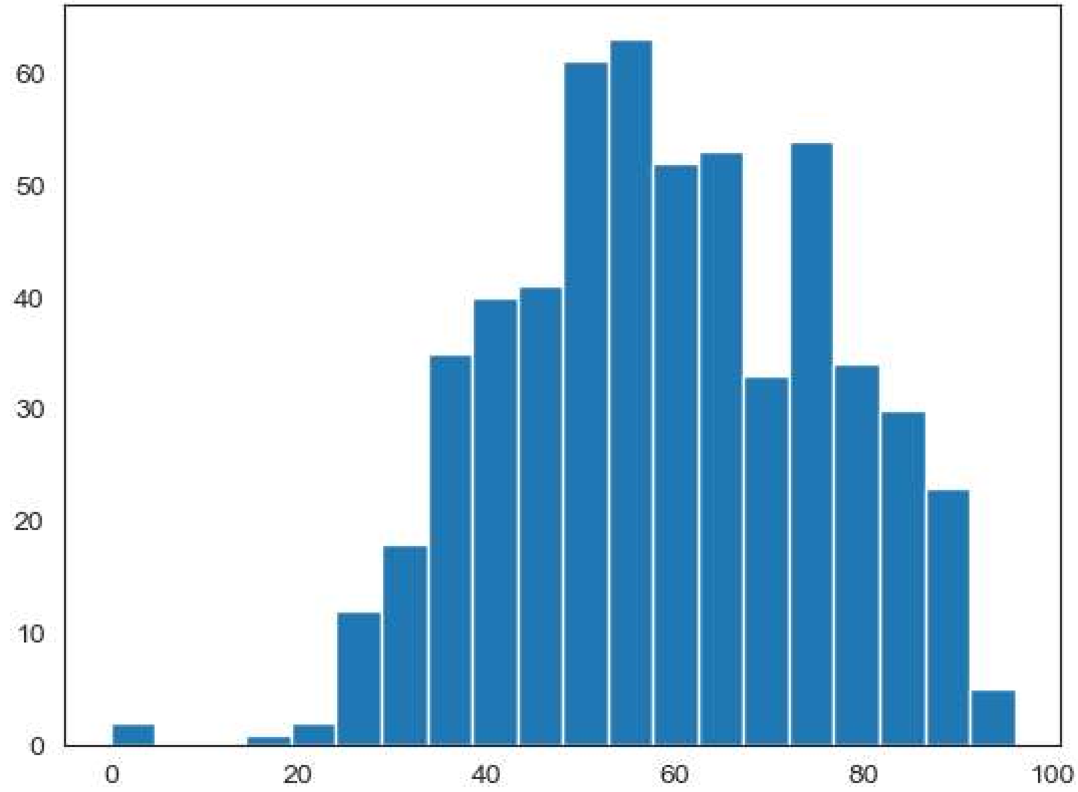
In [33]:

```
#sns.set_style('darkgrid')
n1 = plt.hist(movies.AudienceRating, bins=15)
```



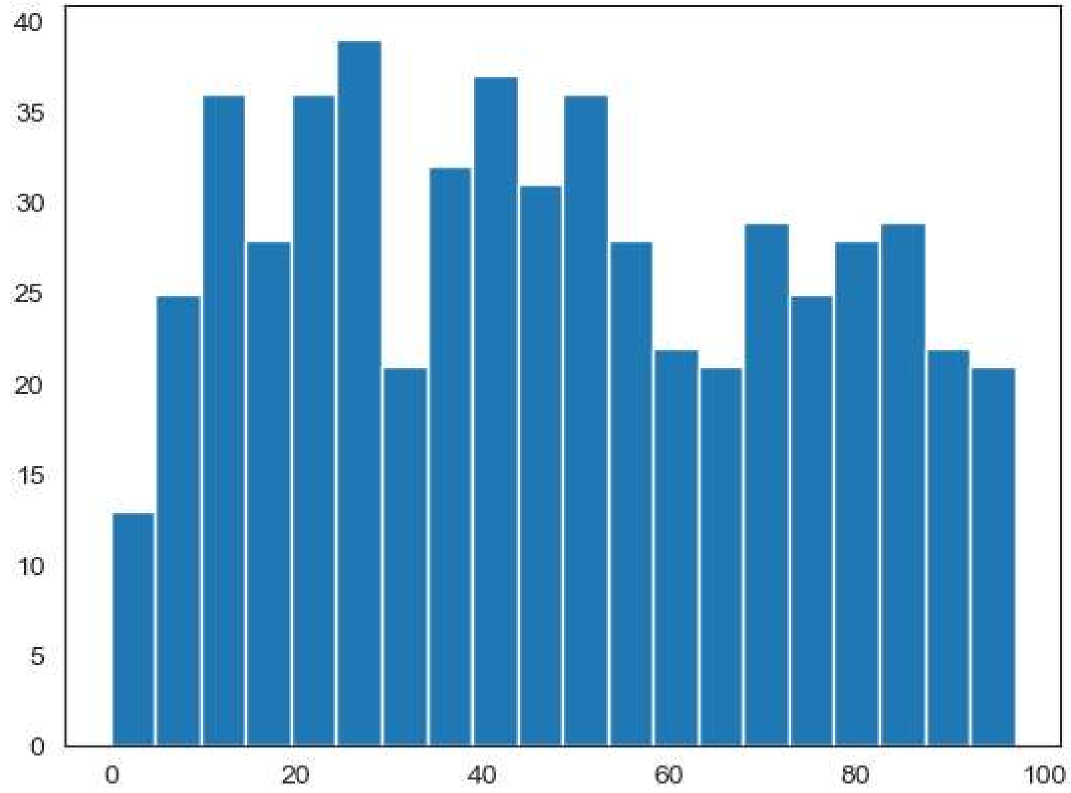
In [34]:

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



In [35]:

```
n1 = plt.hist(movies.CriticRatings, bins=20) #uniform distribution
```

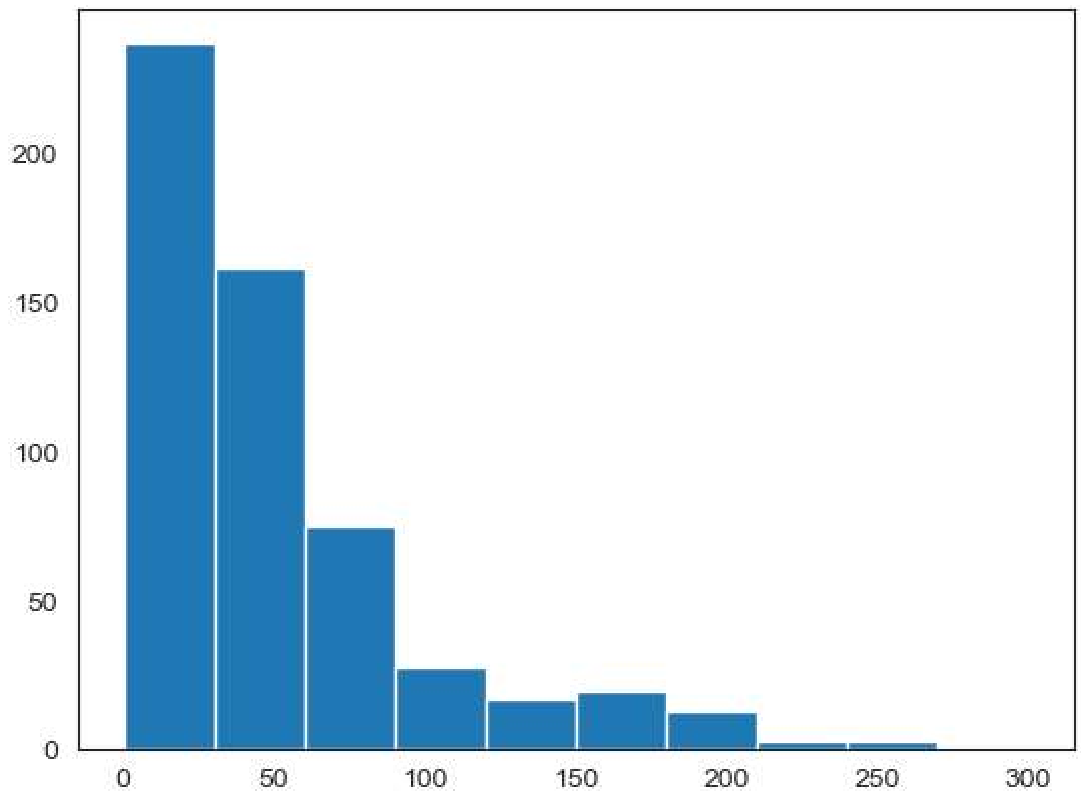


In [36]:

```
# <<< chat - 2
# creating stacked histograms & this is bit tough too understand
```

In [37]:

```
# h1 = plt.hist(movies.BudgrtMillions)

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



In [38]:

```
movies
```

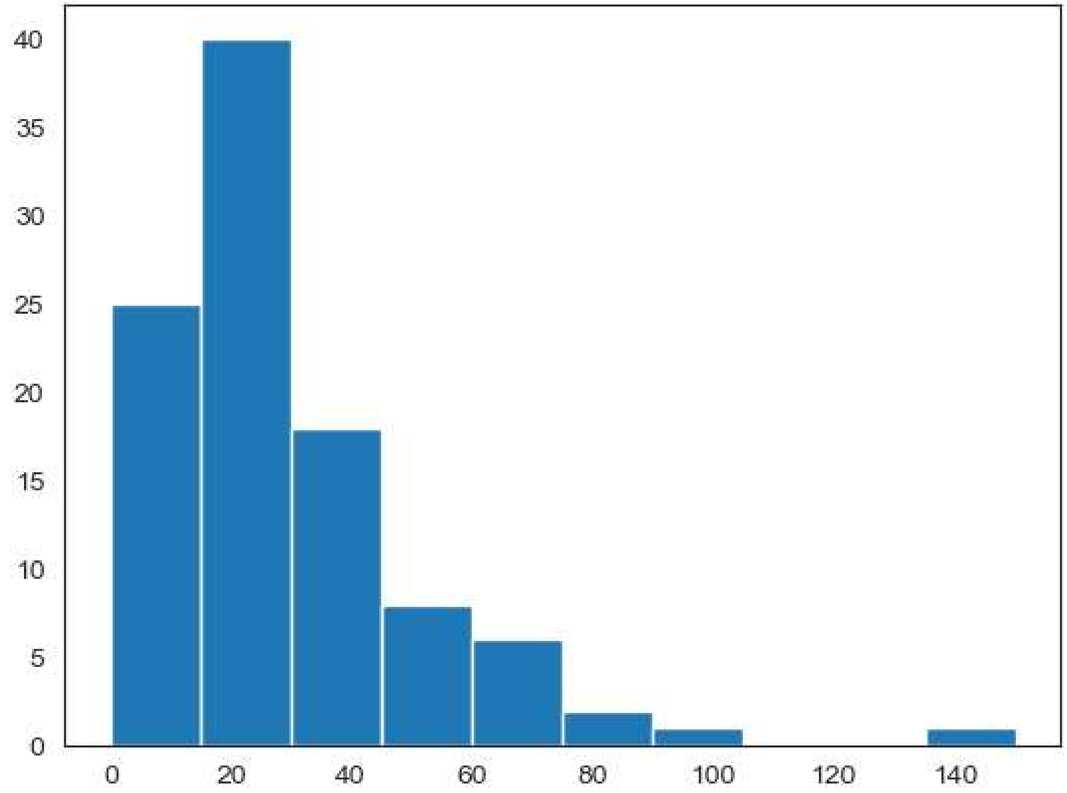
Out[38]:

	Film	Genre	CriticRatings	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 [39]:

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



In [40]:

```
movies.head()
```

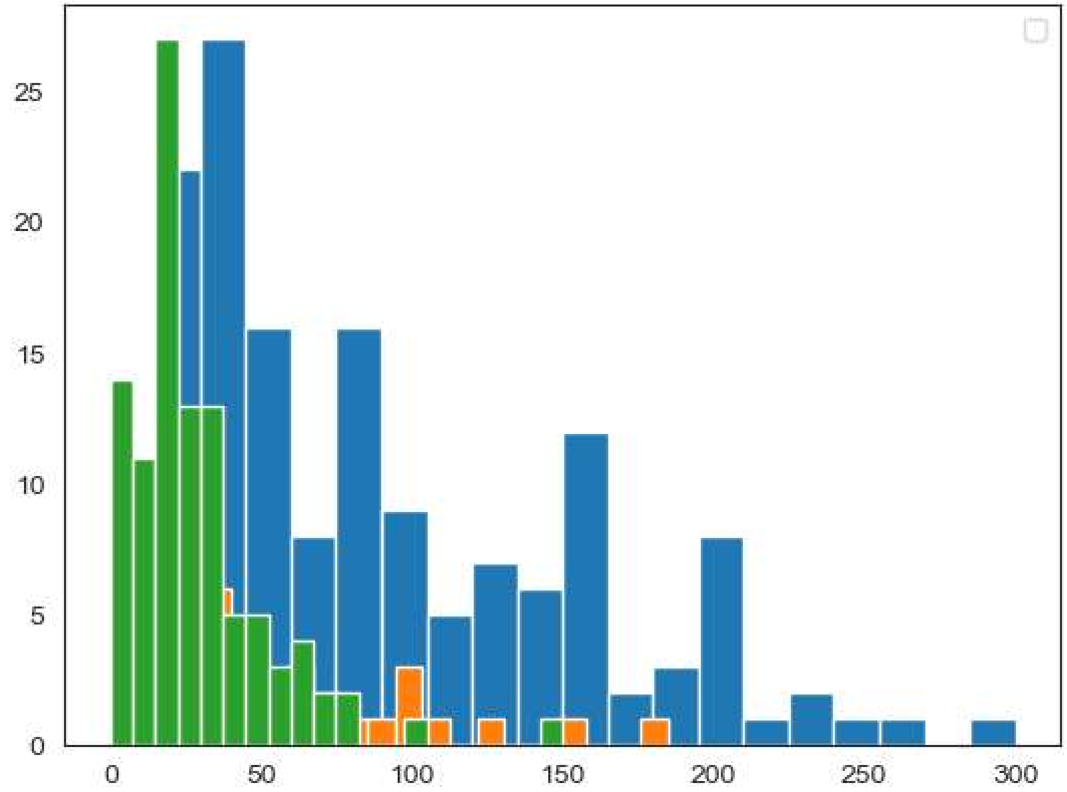
Out[40]:

	Film	Genre	CriticRatings	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 [41]:

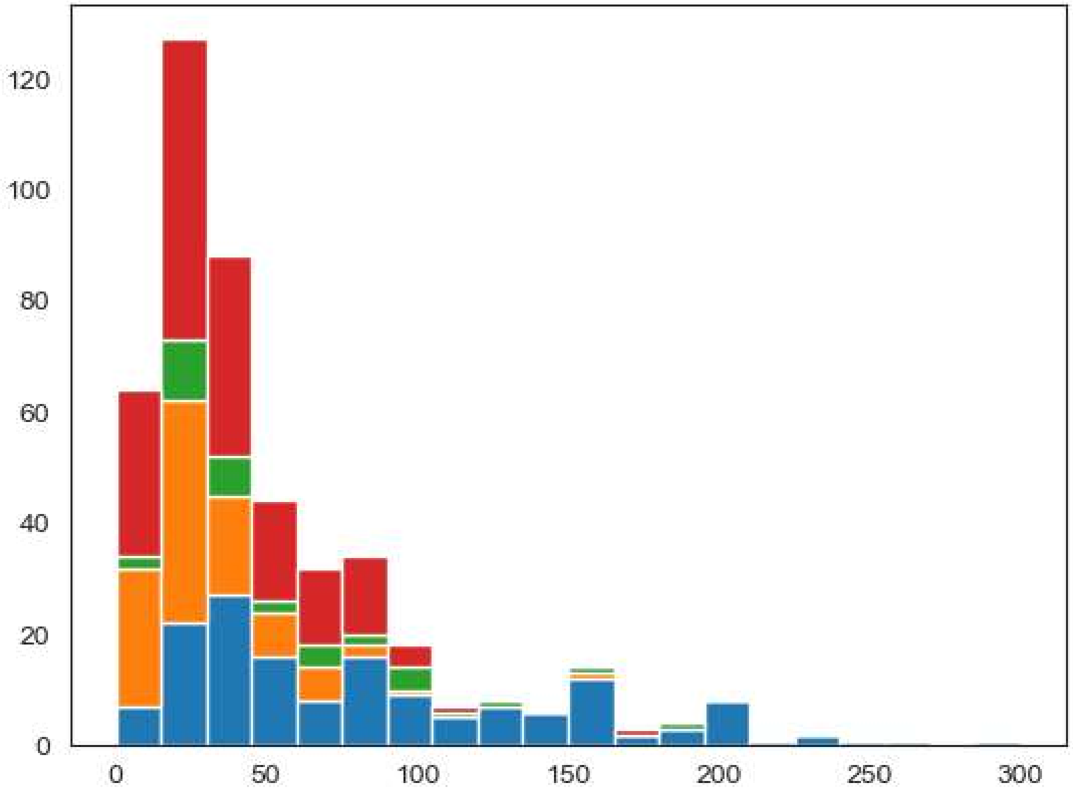
```
#movies.Genre.unique()
# Below plots are stacked histogram becuase overlaped

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

```
No artists with labels found to put in legend.  Note that artists whose la
bel start with an underscore are ignored when legend() is called with no a
rgument.
```



In [42]:

```
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()
```



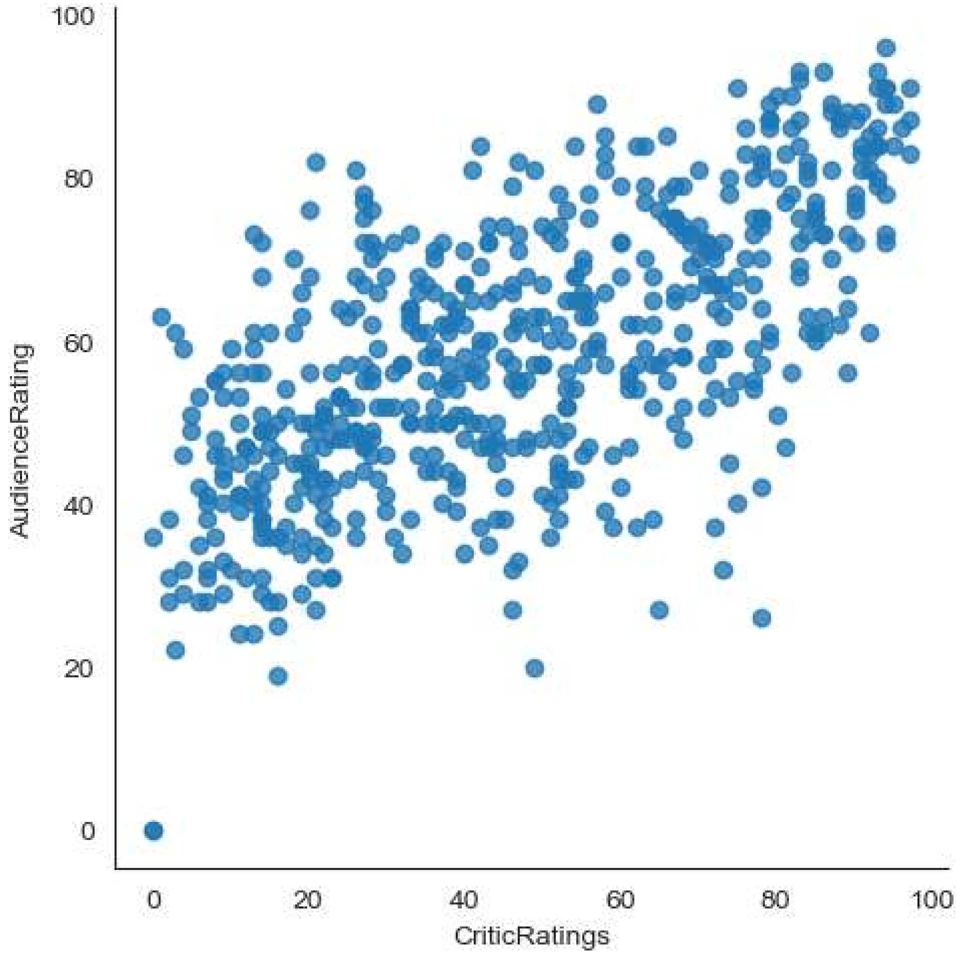
In [43]:

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

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

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

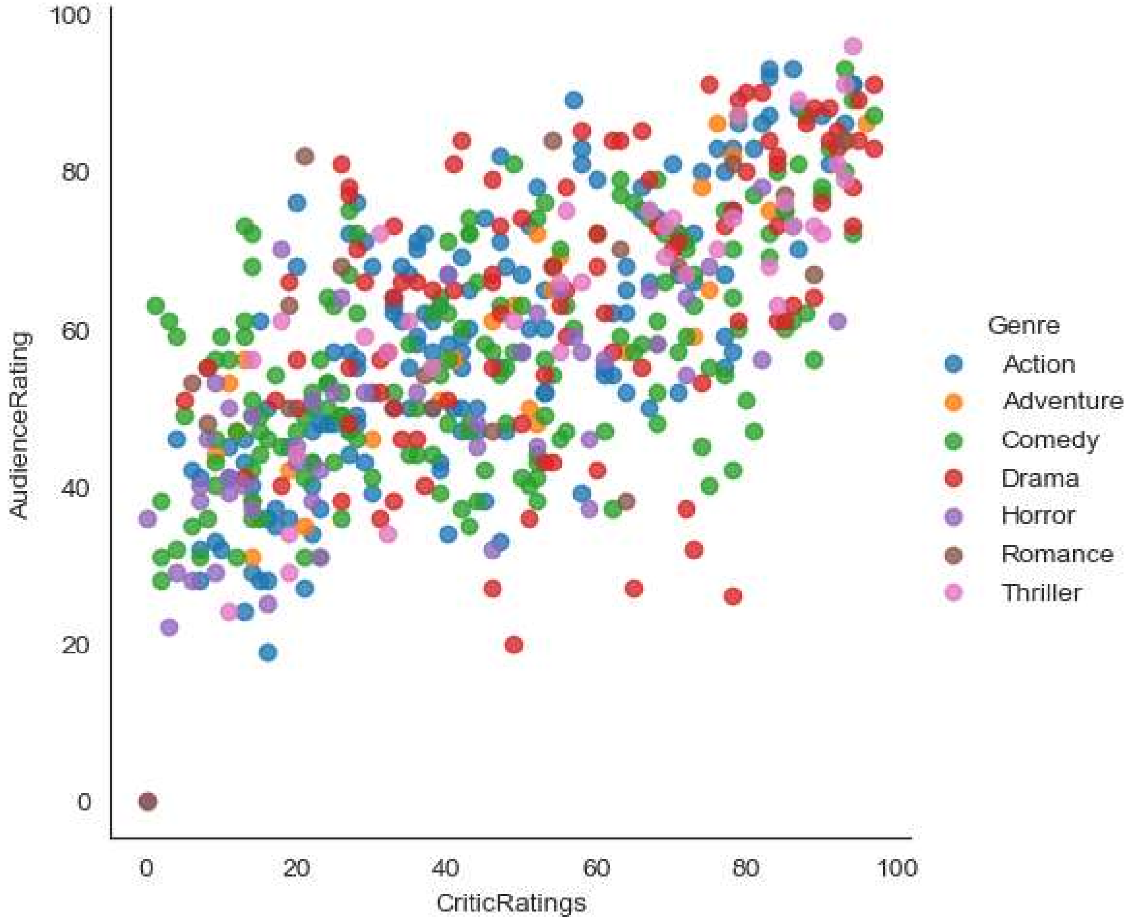
In [44]:

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



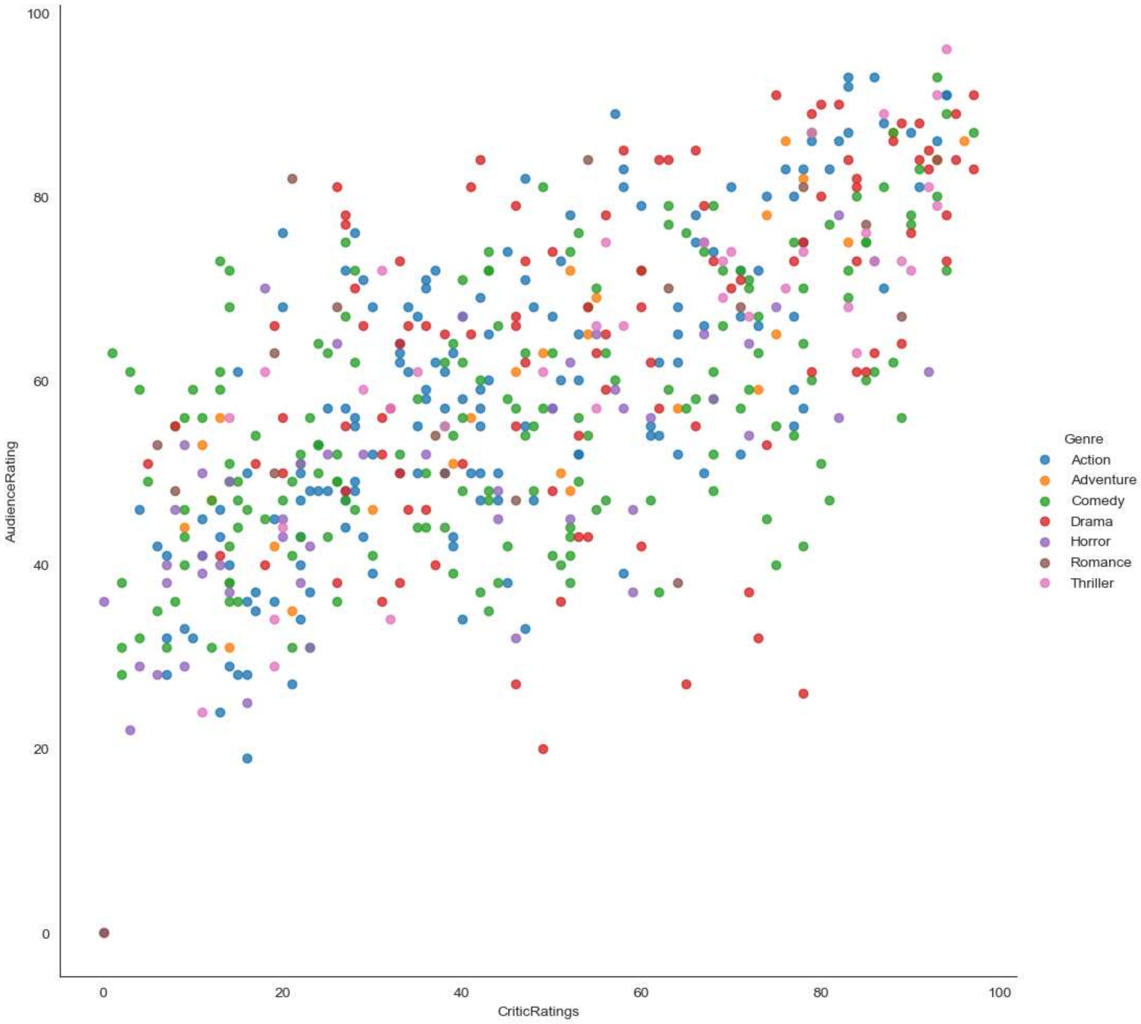
In [45]:

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



In [46]:

```
vis1 = sns.lmplot(data=movies, x='CriticRatings', y='AudienceRating',\
                  fit_reg=False, hue = 'Genre', height = 10,aspect=1 )
```



In [47]:

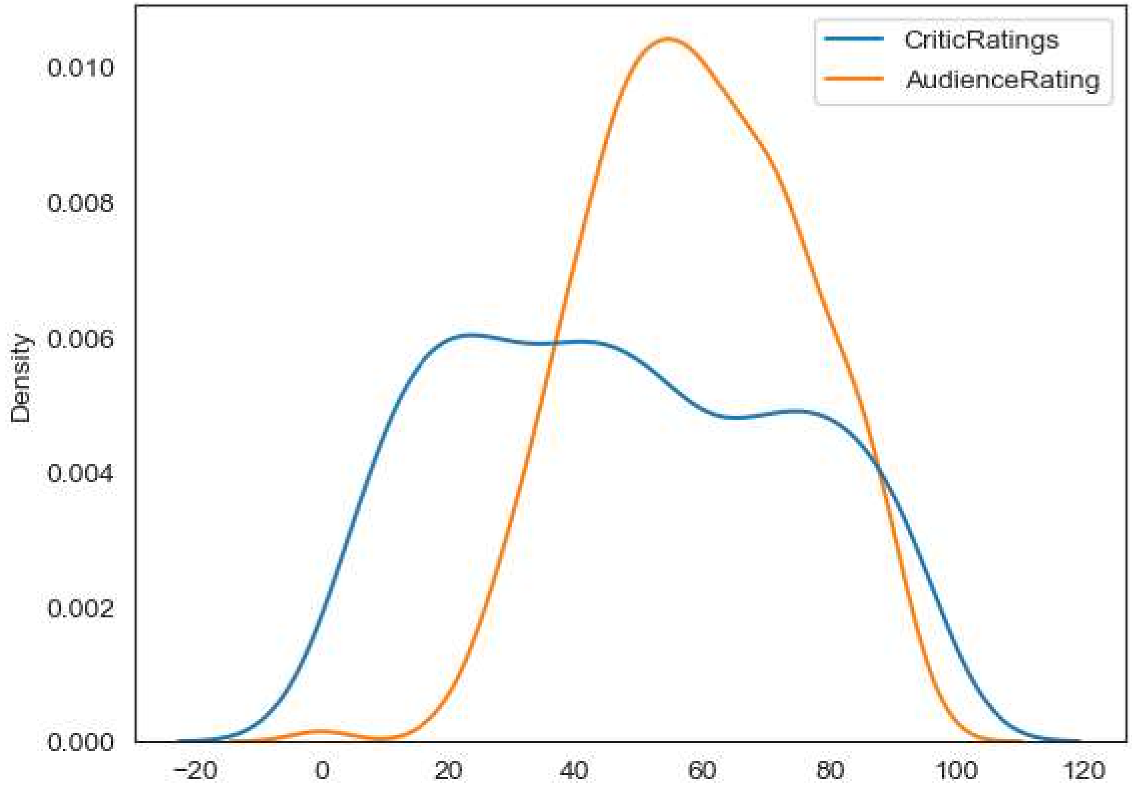
```
# Kernal Density Estimate plot ( KDE PLOT)
# how can i visulize audience rating & critics rating . using scatterplot
```

In [53]:

```

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

In [49]:

```
import pandas as pd
combined_ratings = pd.concat([movies['CriticRatings'], movies['AudienceRating']], axis=1)
k1 = sns.kdeplot(data=combined_ratings)
```



In [50]:

```
movies
```

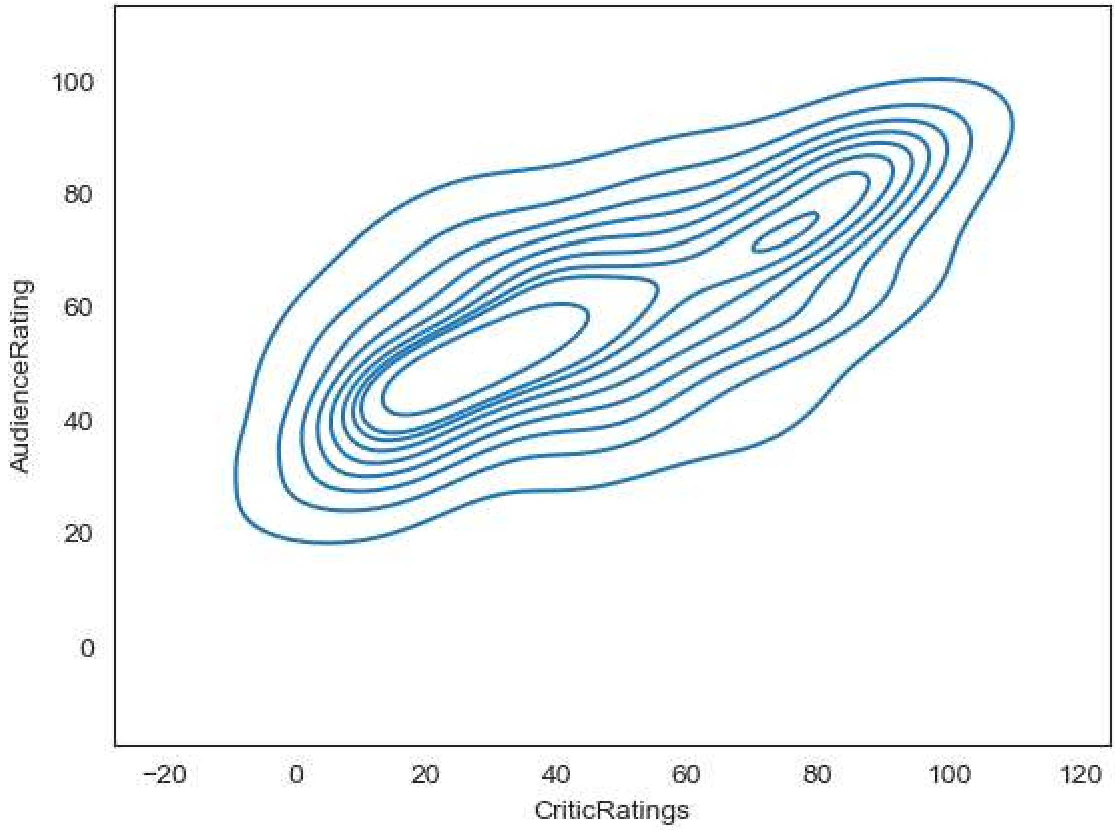
Out[50]:

	Film	Genre	CriticRatings	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

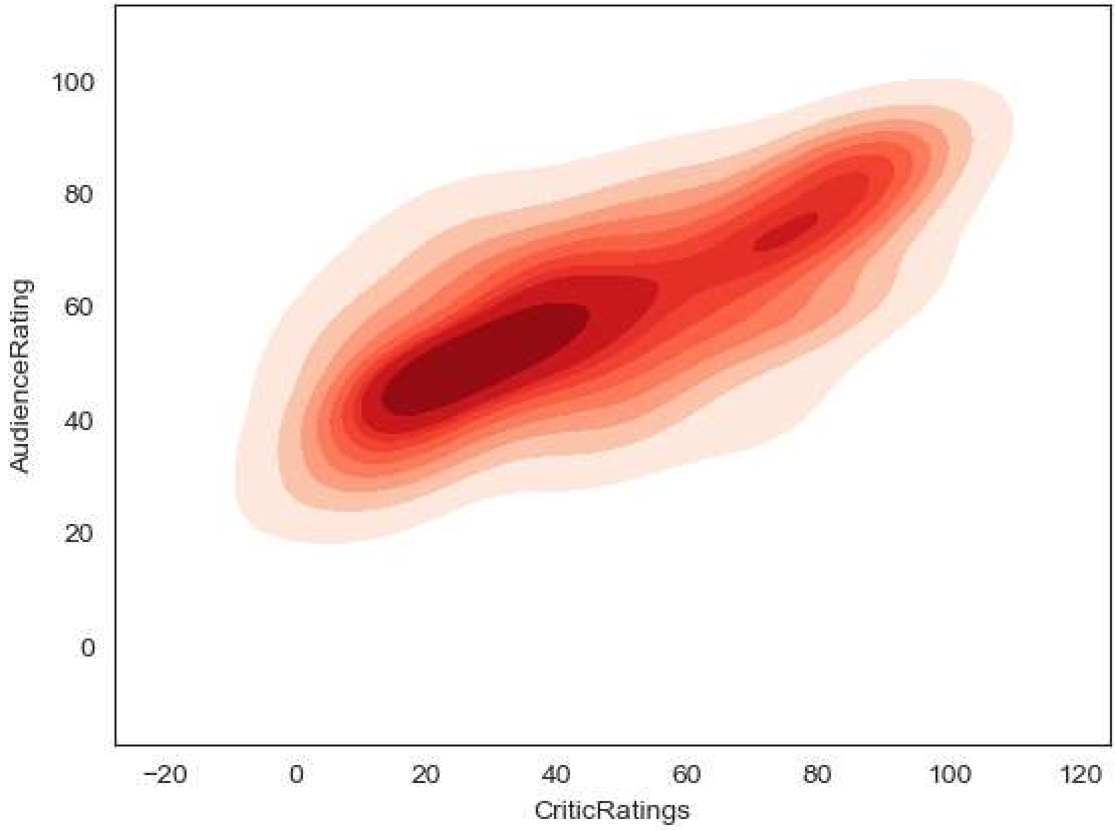
In [54]:

```
k1 = sns.kdeplot(data=movies,x='CriticRatings', y='AudienceRating')
```



In [55]:

```
k1 = sns.kdeplot(data=movies,x='CriticRatings',y ='AudienceRating',shade = True,shade_low
```

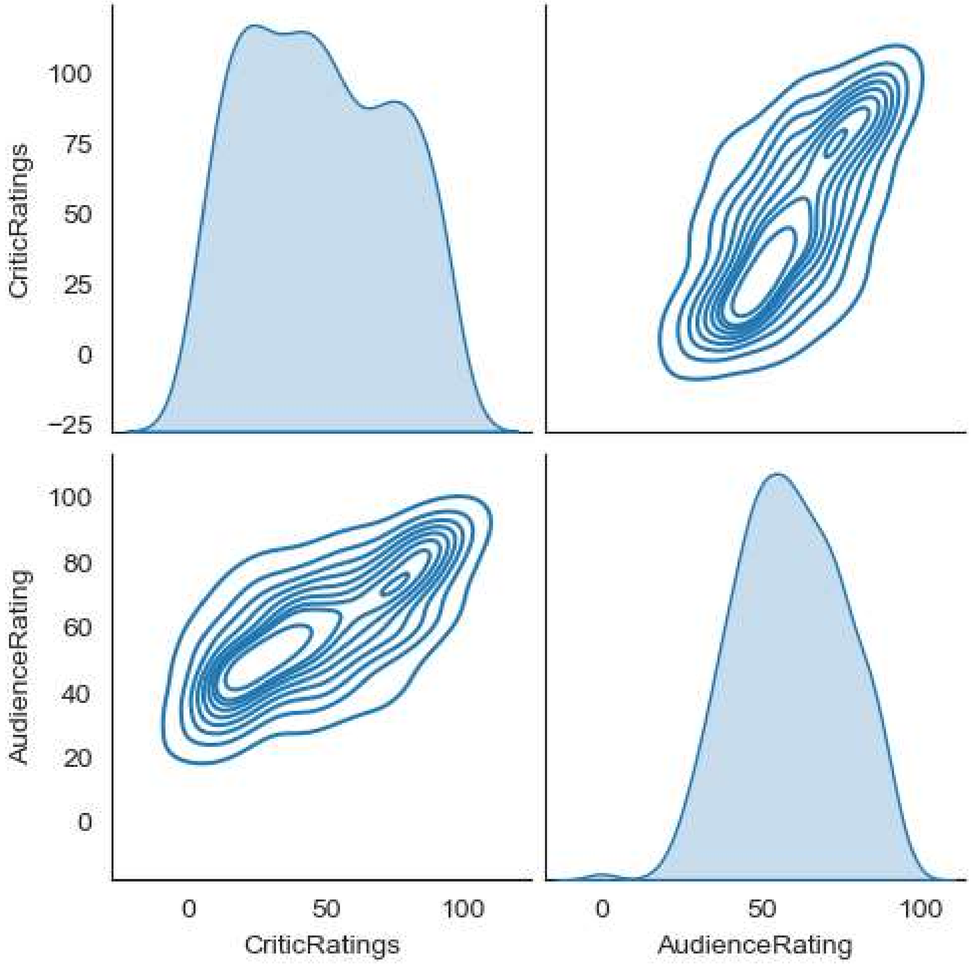


In [56]:

```
sns.pairplot(data=movies, vars=['CriticRatings', 'AudienceRating'], kind='kde')
```

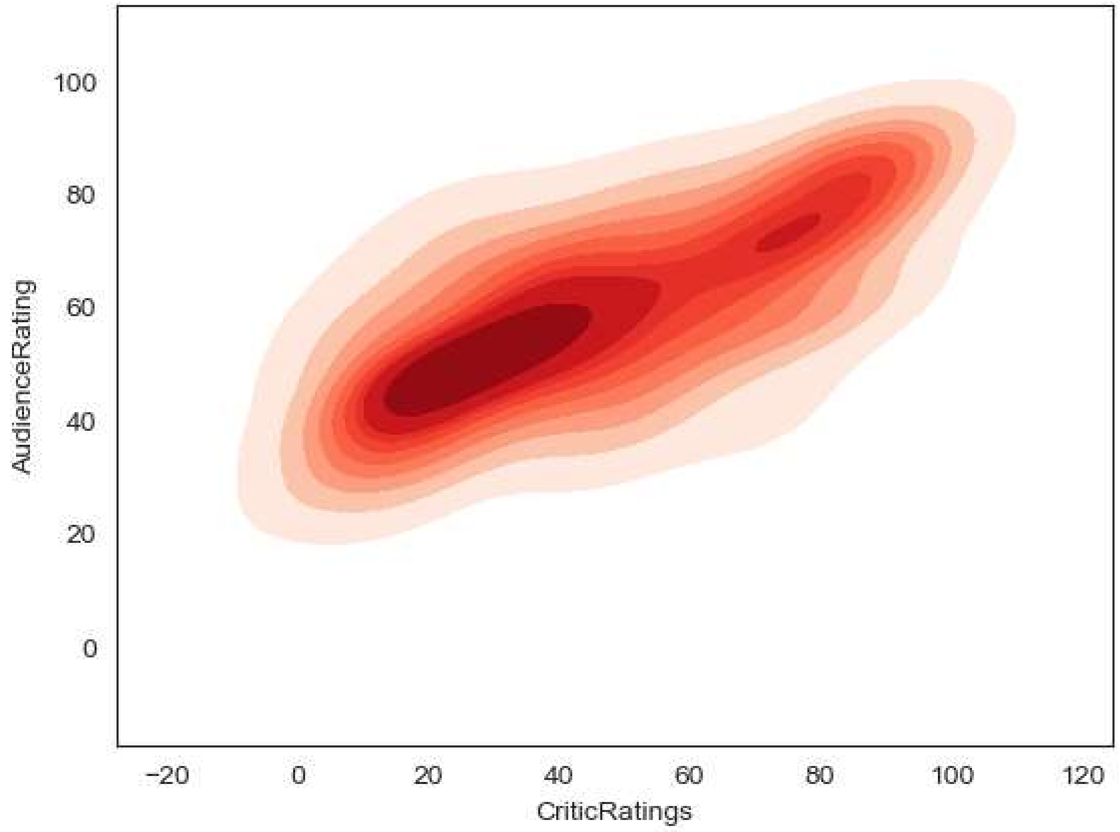
Out[56]:

```
<seaborn.axisgrid.PairGrid at 0x1dc9aa96e30>
```



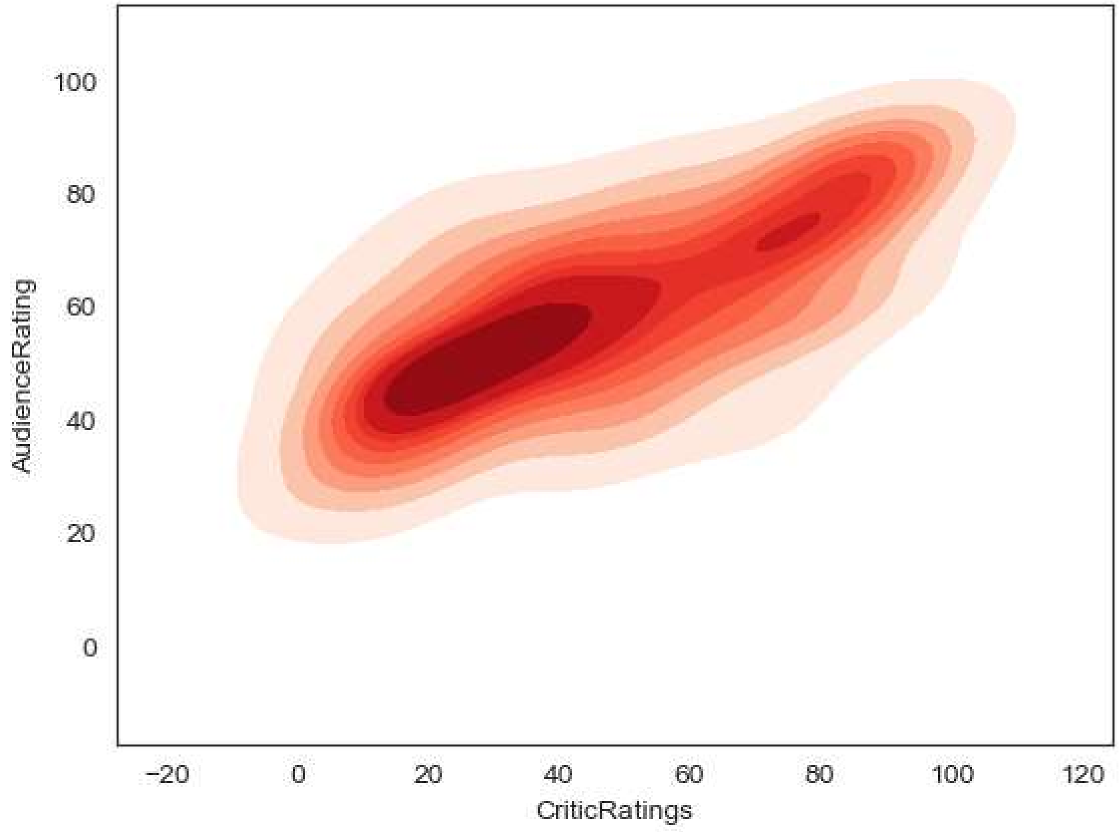
In [58]:

```
k1 = sns.kdeplot(data=movies, x='CriticRatings',y='AudienceRating',shade = True,shade_low
```



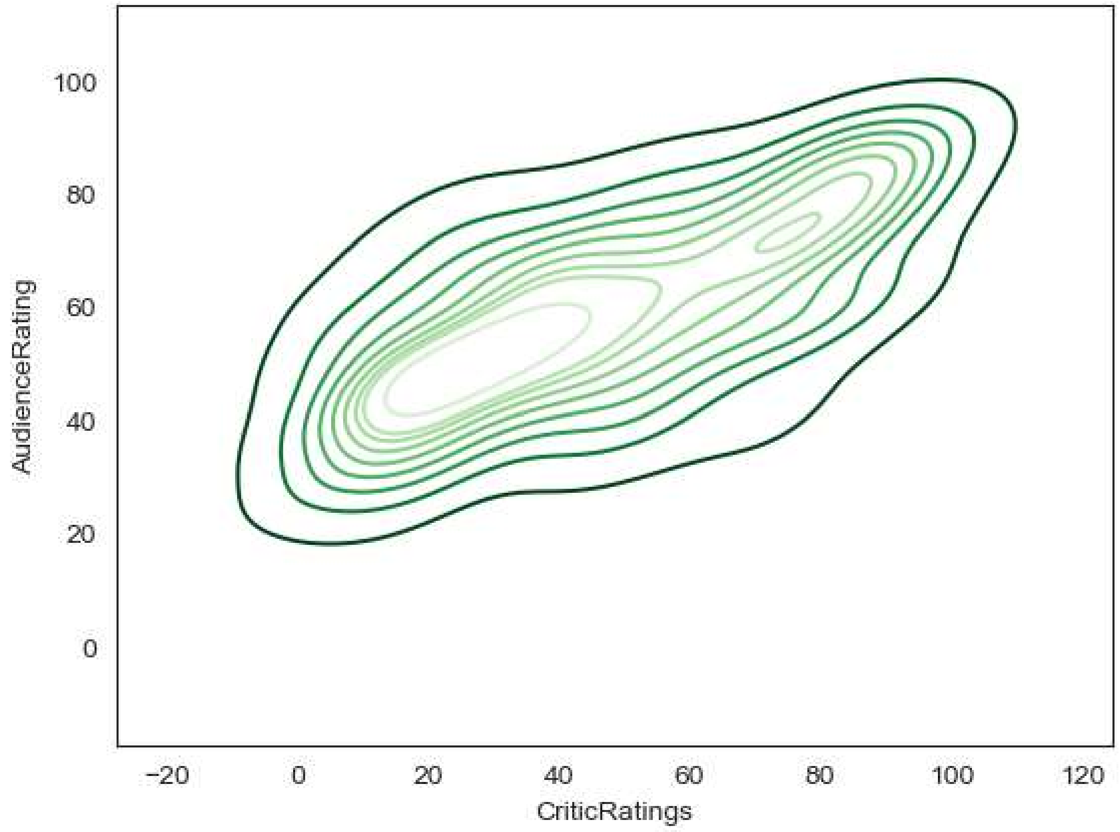
In [59]:

```
k1 = sns.kdeplot(data=movies, x='CriticRatings', y='AudienceRating', shade=True, shade_lo
```



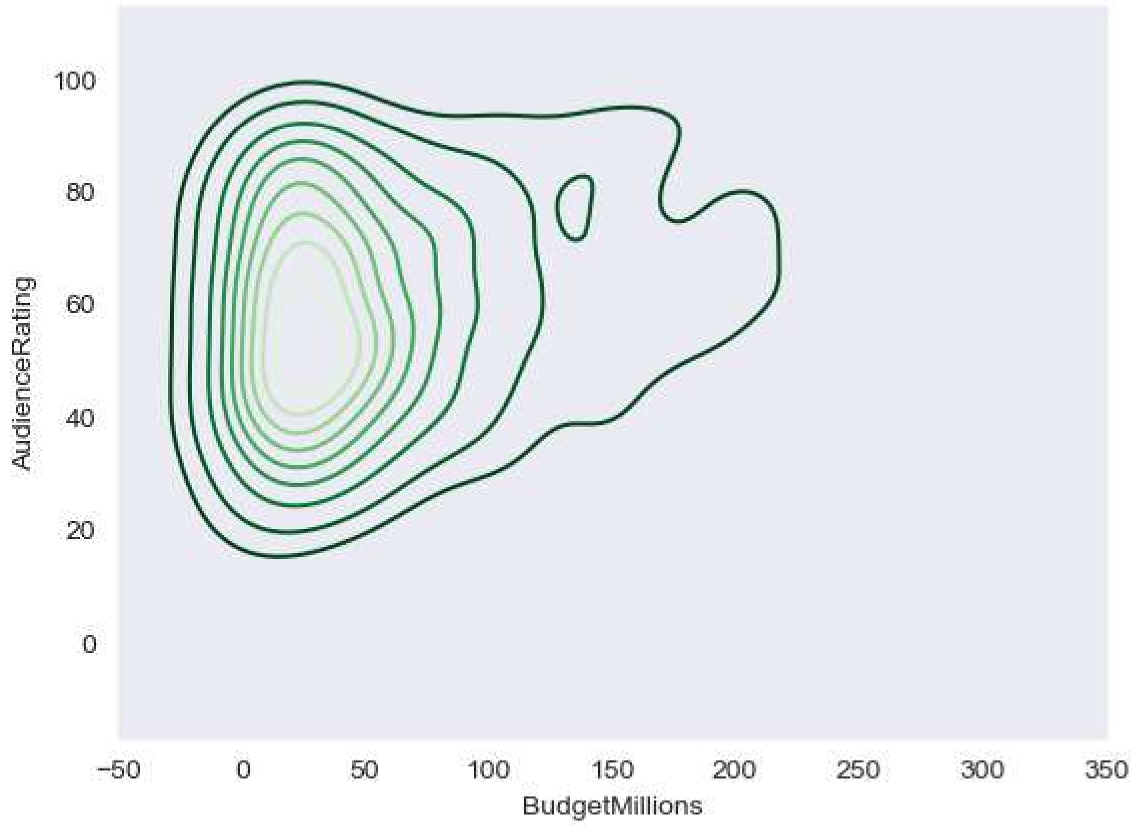
In [60]:

```
k2 = sns.kdeplot(data=movies,x='CriticRatings', y='AudienceRating',shade_lowest=False,cma
```



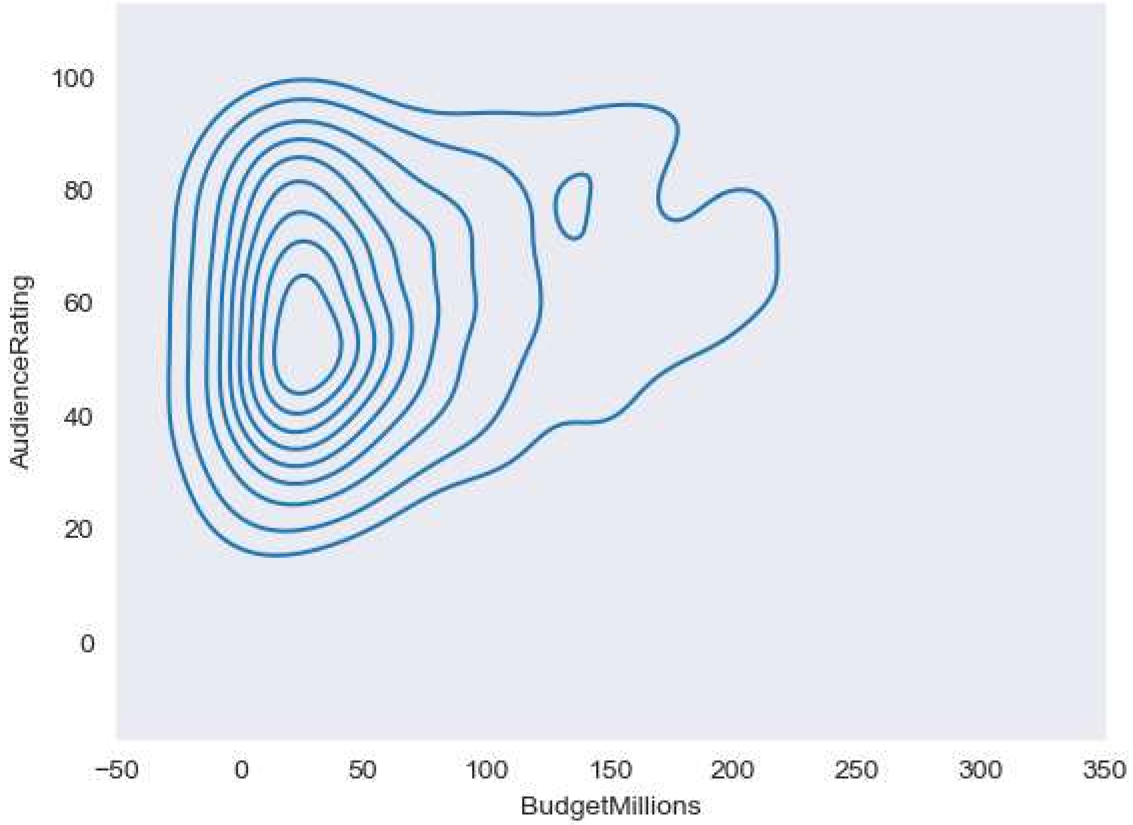
In [61]:

```
sns.set_style('dark')
k1 = sns.kdeplot(data=movies, x='BudgetMillions',y='AudienceRating',shade_lowest=False,cm
```



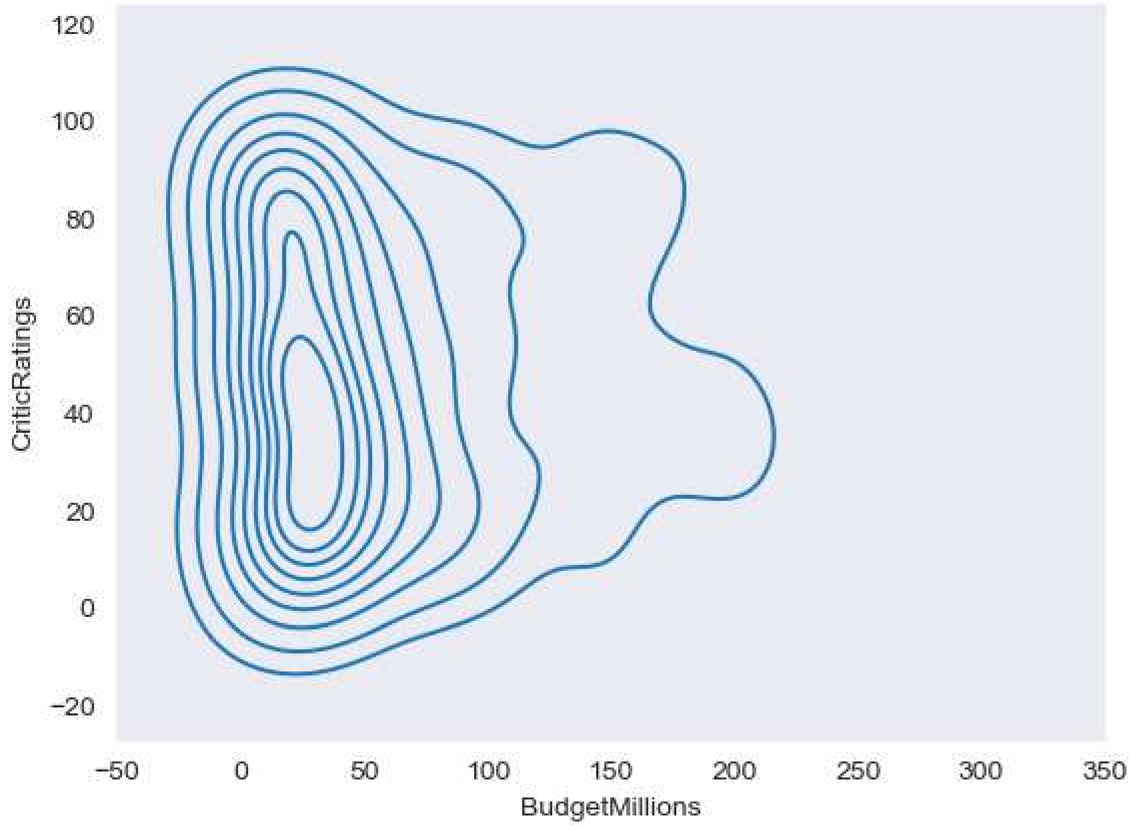
In [62]:

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



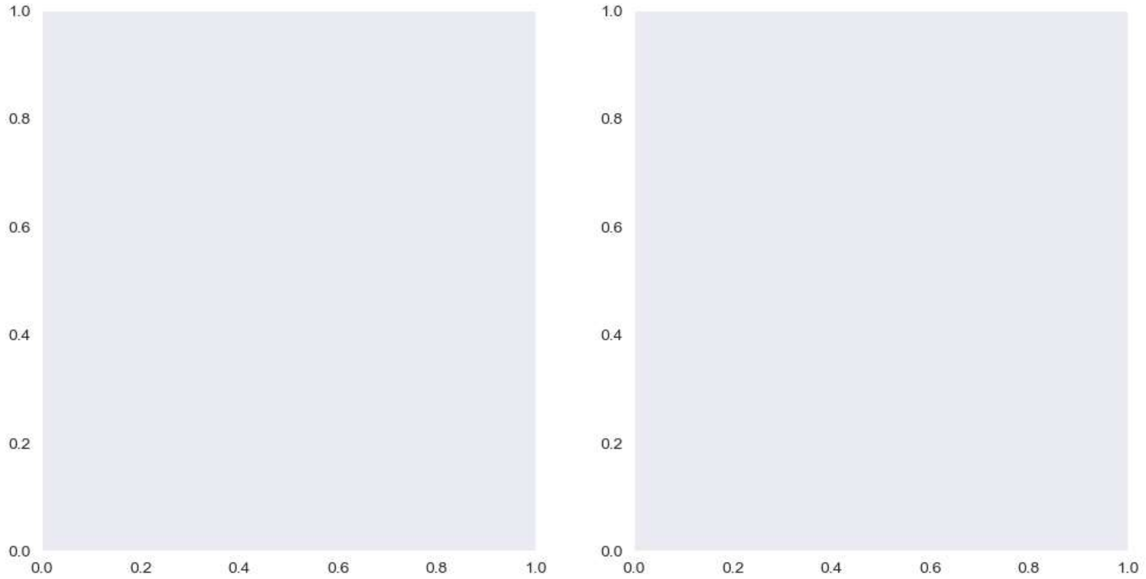
In [63]:

```
k2 = sns.kdeplot(data=movies, x='BudgetMillions',y='CriticRatings')
```



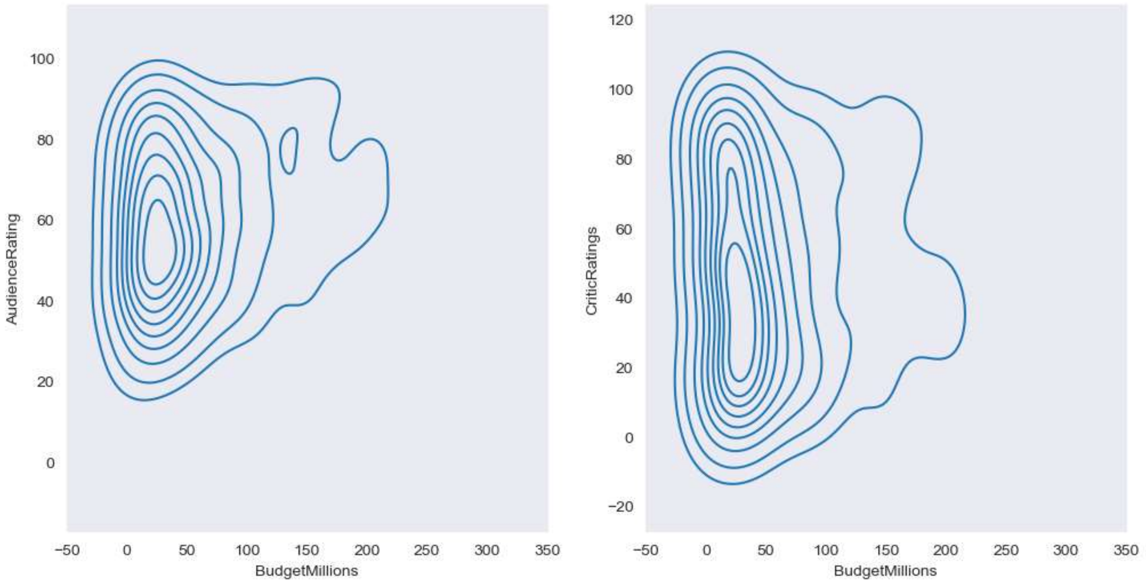
In [64]:

```
# subplots

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



In [65]:

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

k1 = sns.kdeplot(data=movies, x='BudgetMillions',y = 'AudienceRating',ax=axes[0])
k2 = sns.kdeplot(data=movies, x='BudgetMillions',y = 'CriticRatings',ax=axes[1])
```



In [66]:

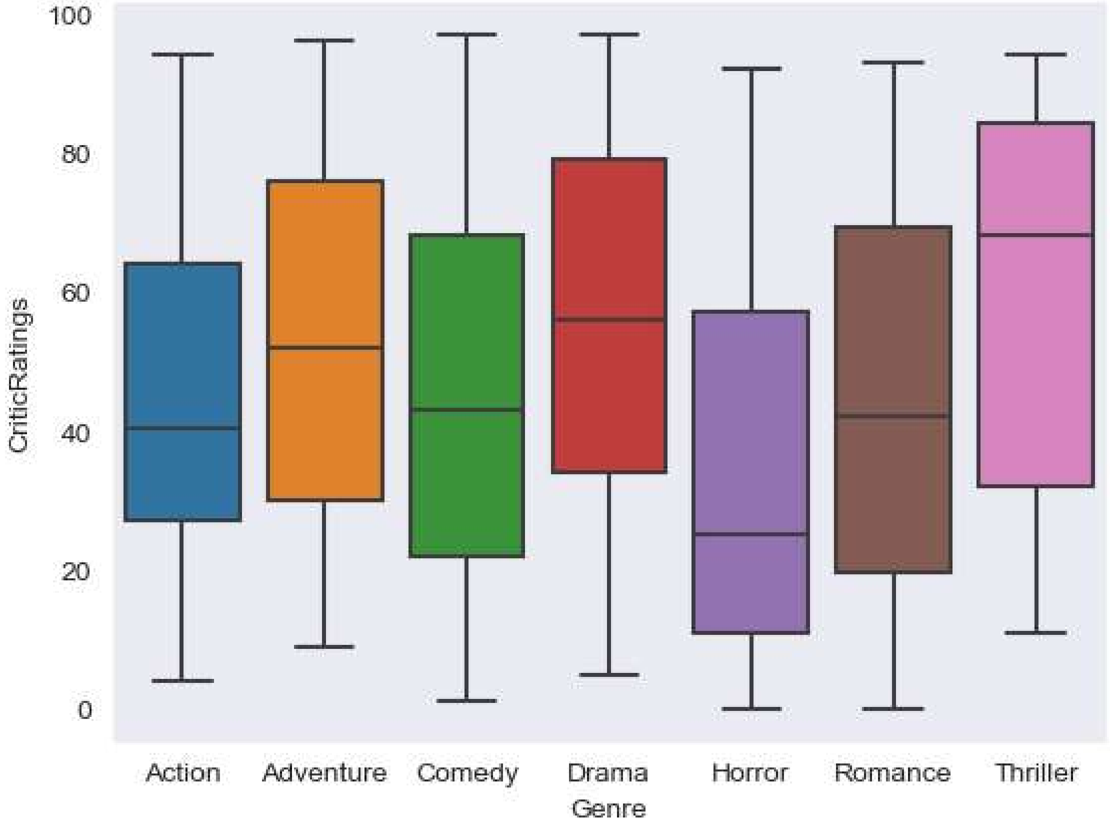
```
axes
```

Out[66]:

```
array([<Axes: xlabel='BudgetMillions', ylabel='AudienceRating'>,
       <Axes: xlabel='BudgetMillions', ylabel='CriticRatings'>],
      dtype=object)
```

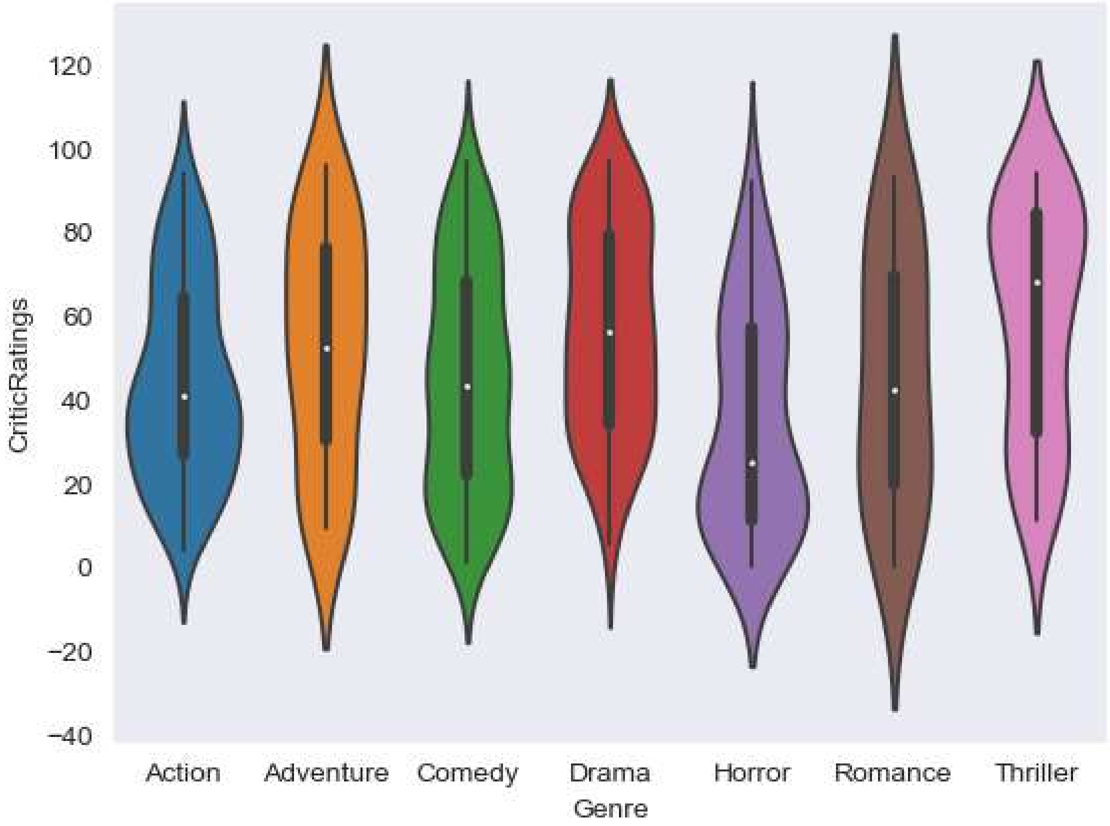
In [67]:

```
#Box plots -

w = sns.boxplot(data=movies, x='Genre', y = 'CriticRatings')
```



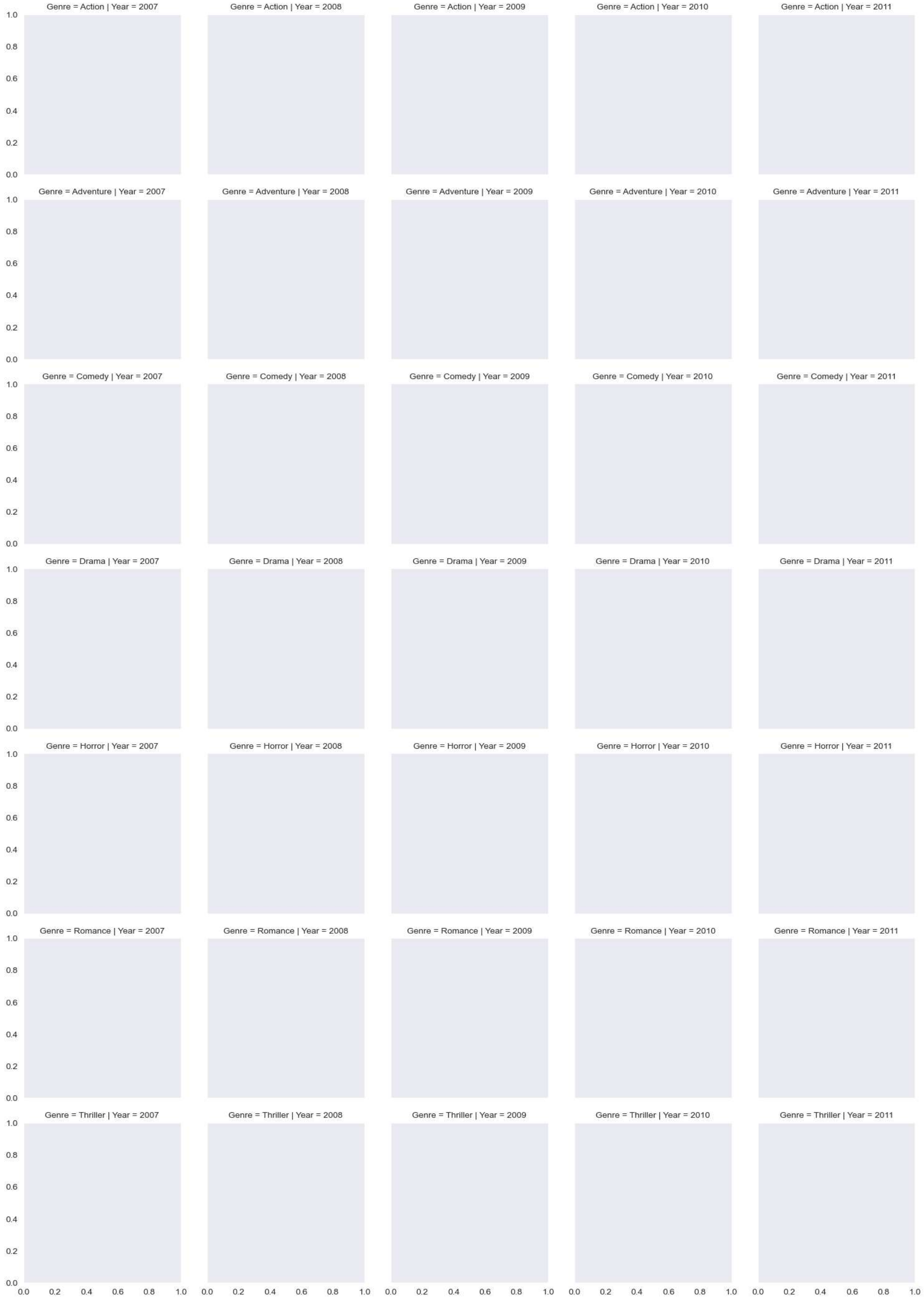
In [68]:

```
#violin plot

z = sns.violinplot(data=movies, x='Genre', y = 'CriticRatings')
```



In [70]:

```
# Createing a Facet grid

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

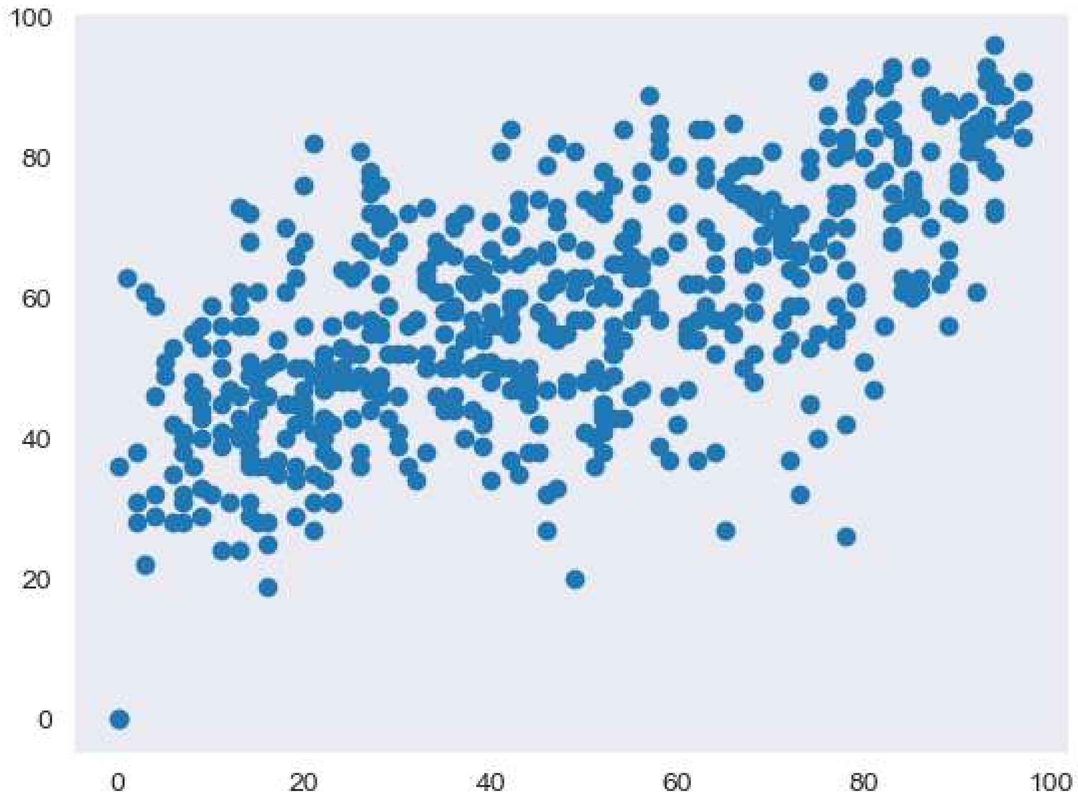


In [71]:

```
plt.scatter(movies.CriticRatings,movies.AudienceRating)
```

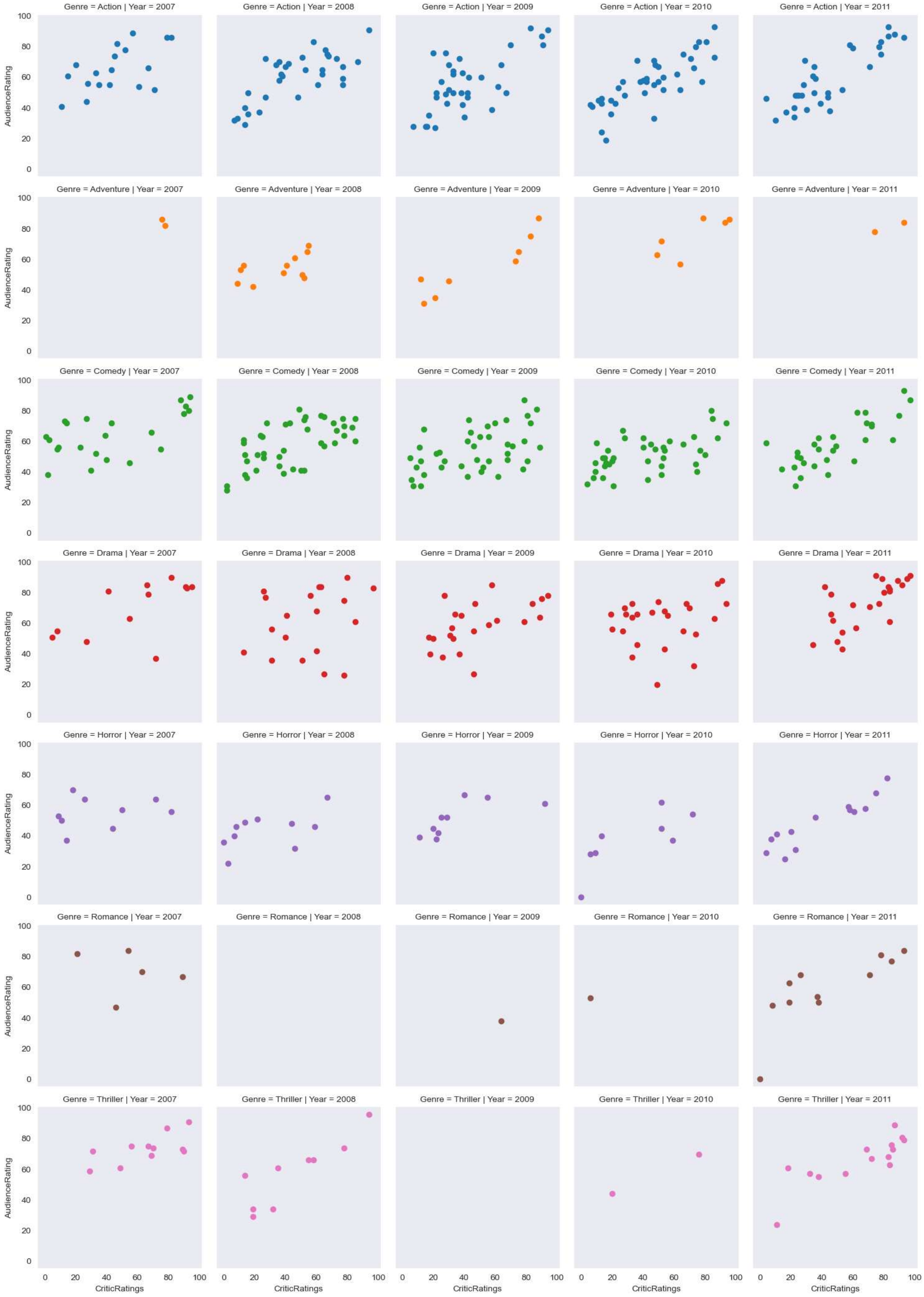
Out[71]:

```
<matplotlib.collections.PathCollection at 0x1dca24273d0>
```



In [72]:

```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.scatter, 'CriticRatings', 'AudienceRating' ) #scatterplots are mapped in fo
```



In [ ]: