Movies Ratings

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

In [2]:
```
import numpy as np
```

In [3]:
```
movies = pd.read_csv(r'E:\Data Science & AI\Dataset files\Movie-Rating.csv')
```

In [4]:
```
movies
```

Out[4]:

	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 [5]:
```
len(movies)
```

Out[5]:
```
559
```

In [6]:
```
movies.columns
```

Out[6]:
```
Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
       'Budget (million $)', 'Year of release'],
      dtype='object')
```

In [7]:
```
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   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 [8]:
```
movies.shape
```

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

In [9]:
```
movies.head
```

Out[9]:
```
<bound method NDFrame.head of                          Film      Genre  Rotten Tom
atoes Ratings %  \
0    (500) Days of Summer      Comedy                        87
1              10,000 B.C.  Adventure                         9
2               12 Rounds      Action                        30
3                127 Hours  Adventure                        93
4                17 Again      Comedy                        55
..                    ...         ...                       ...
554          Your Highness     Comedy                        26
555        Youth in Revolt     Comedy                        68
556                 Zodiac   Thriller                        89
557            Zombieland      Action                        90
558              Zookeeper     Comedy                        14

     Audience Ratings %  Budget (million $)  Year of release
0                    81                   8             2009
1                    44                 105             2008
2                    52                  20             2009
3                    84                  18             2010
4                    70                  20             2009
..                  ...                 ...              ...
554                  36                  50             2011
555                  52                  18             2009
556                  73                  65             2007
557                  87                  24             2009
558                  42                  80             2011

[559 rows x 6 columns]>
```

In [10]:
```
movies.tail
```

Out[10]:
```
<bound method NDFrame.tail of                          Film      Genre  Rotten Tom
atoes Ratings %  \
0    (500) Days of Summer      Comedy                      87
1              10,000 B.C.  Adventure                       9
2               12 Rounds      Action                      30
3                127 Hours  Adventure                      93
4                17 Again      Comedy                      55
..                    ...         ...                     ...
554          Your Highness     Comedy                      26
555        Youth in Revolt     Comedy                      68
556                 Zodiac   Thriller                      89
557            Zombieland      Action                      90
558              Zookeeper     Comedy                      14

     Audience Ratings %  Budget (million $)  Year of release
0                    81                   8             2009
1                    44                 105             2008
2                    52                  20             2009
3                    84                  18             2010
4                    70                  20             2009
..                  ...                 ...              ...
554                  36                  50             2011
555                  52                  18             2009
556                  73                  65             2007
557                  87                  24             2009
558                  42                  80             2011

[559 rows x 6 columns]>
```

In [14]:
```
movies.columns = ['Film', 'Genre', 'CriticRatings', 'AudienceRatings',
       'BudgetMillions', 'Year']
```

In [22]:
```
movies.head() # Removed spaces & % removed noise characters
```

Out[22]:

	Film	Genre	CriticRatings	AudienceRatings	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 [25]:
```
movies.describe()
# if you look at the year the data type is int but when you look at the mean val
# we have to change to categroy type
# also from object datatype we will convert to category datatypes
```

Out[25]:

	CriticRatings	AudienceRatings	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 [27]:
```
movies['Film']
#movies['Audience Ratings %']
```

Out[27]:
```
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 [29]:
```
movies.Film
```

Out[29]:
```
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 [31]:
```
movies.Film = movies.Film.astype('category')
movies.Genre = movies.Genre.astype('category')
movies.Year = movies.Year.astype('category')
```

In [33]:
```
movies.Film
```

Out[33]:
```
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 [35]:
```
movies.head()
```

Out[35]:

	Film	Genre	CriticRatings	AudienceRatings	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 [37]:
```
movies.info()
# now the same thing we will change genra 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    category
 1   Genre            559 non-null    category
 2   CriticRatings    559 non-null    int64
 3   AudienceRatings  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 [39]:
```
movies.Genre
movies.Year # is it real no. year you can take average,min,max but out come have
```

Out[39]:
```
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 [41]:
```
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   AudienceRatings  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 [43]:
```
movies.Genre.cat.categories
```

Out[43]:
```
Index(['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'Romance',
       'Thriller'],
      dtype='object')
```

In [45]:
```
movies.describe()#now when you see the describt you will get only integer value
```

Out[45]:

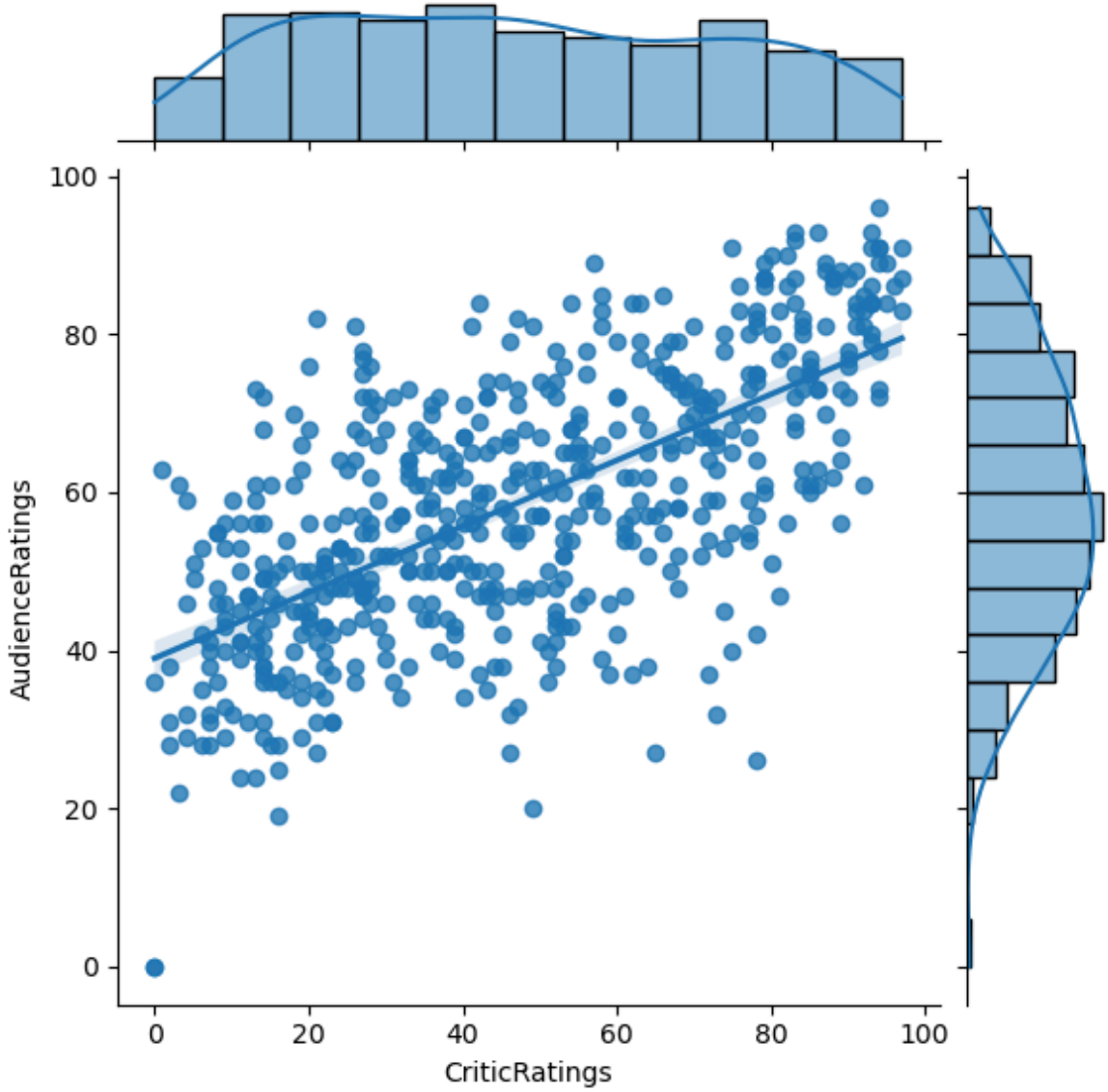
	CriticRatings	AudienceRatings	BudgetMillions
count	559.000000	559.000000	559.000000
mean	47.309481	58.744186	50.236136
std	26.413091	16.826887	48.731817
min	0.000000	0.000000	0.000000
25%	25.000000	47.000000	20.000000
50%	46.000000	58.000000	35.000000
75%	70.000000	72.000000	65.000000
max	97.000000	96.000000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

In [47]:
```
# How to working with joint plots
from matplotlib import pyplot as plt
import seaborn as sns

%matplotlib inline
```

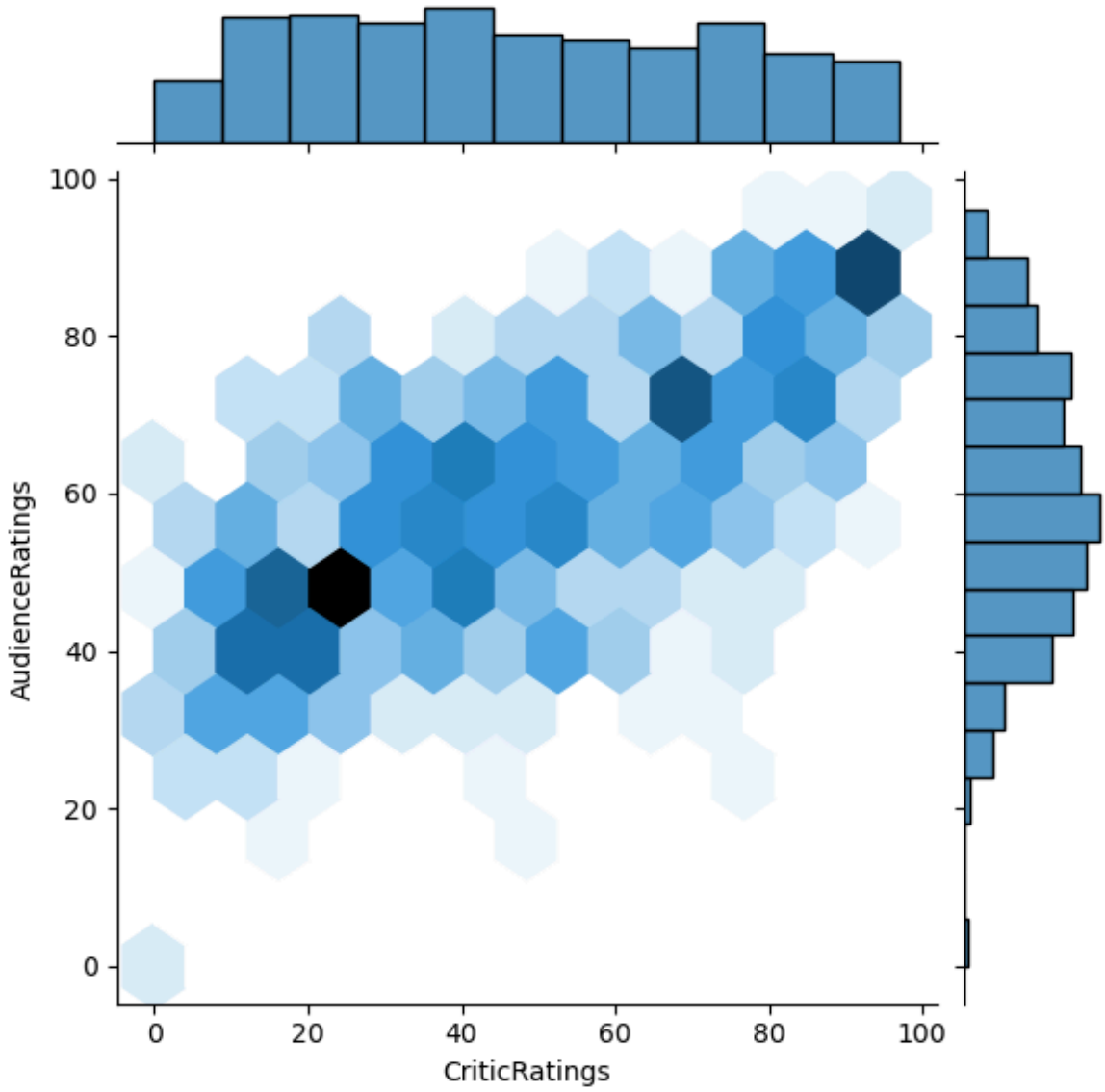
```
import warnings
warnings.filterwarnings('ignore')
```

* 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)

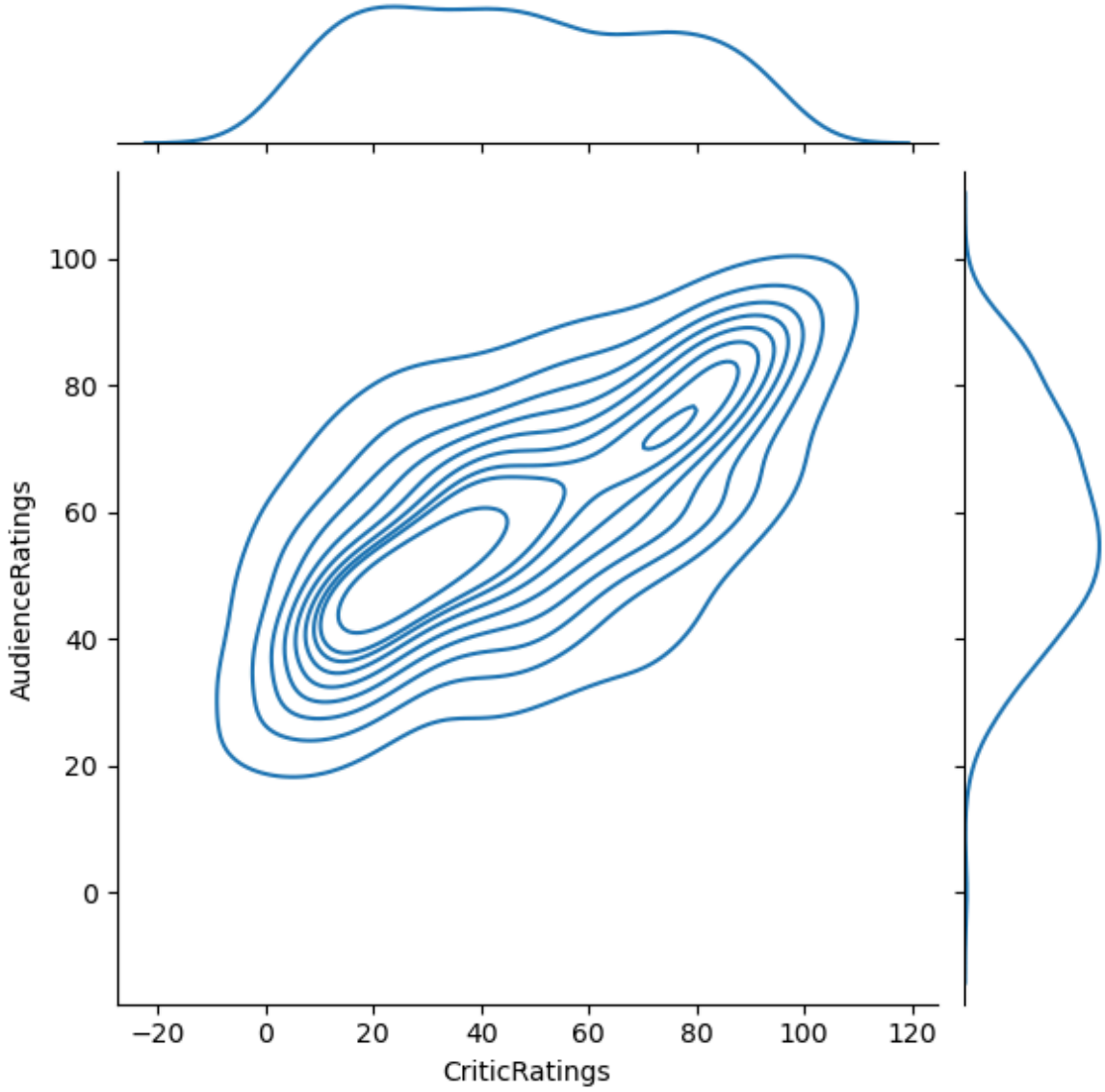
In [48]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind =
# Audience rating is more dominant then critics rating
# Based on this we find out as most people are most liklihood to watch audience
# let me explain the excel - if you filter audience rating & critic rating. crit
```



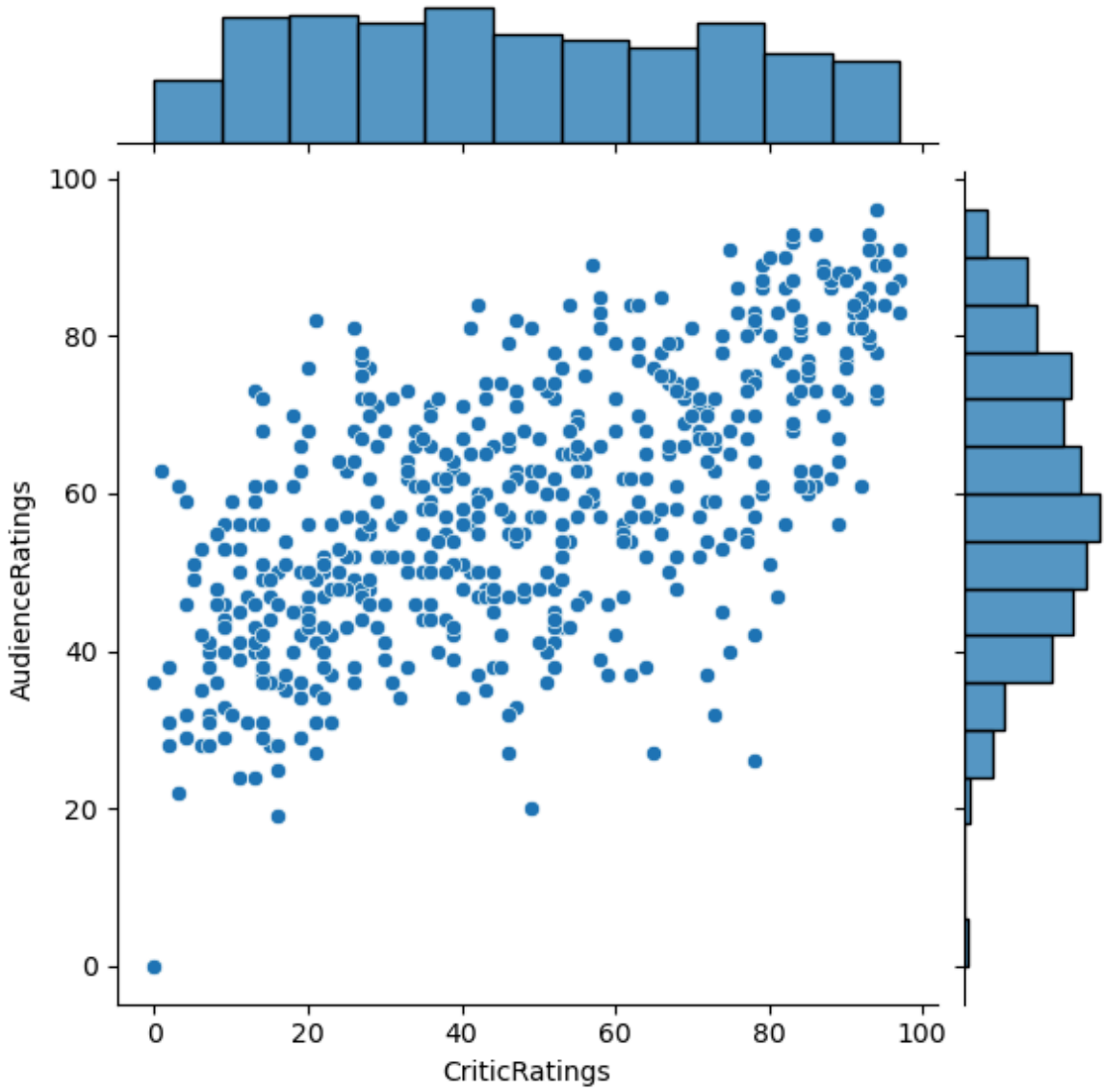
In [49]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind =
#j = sns.jointplot( data = movies, x = 'CriticRating', y = 'AudienceRating', kin
```



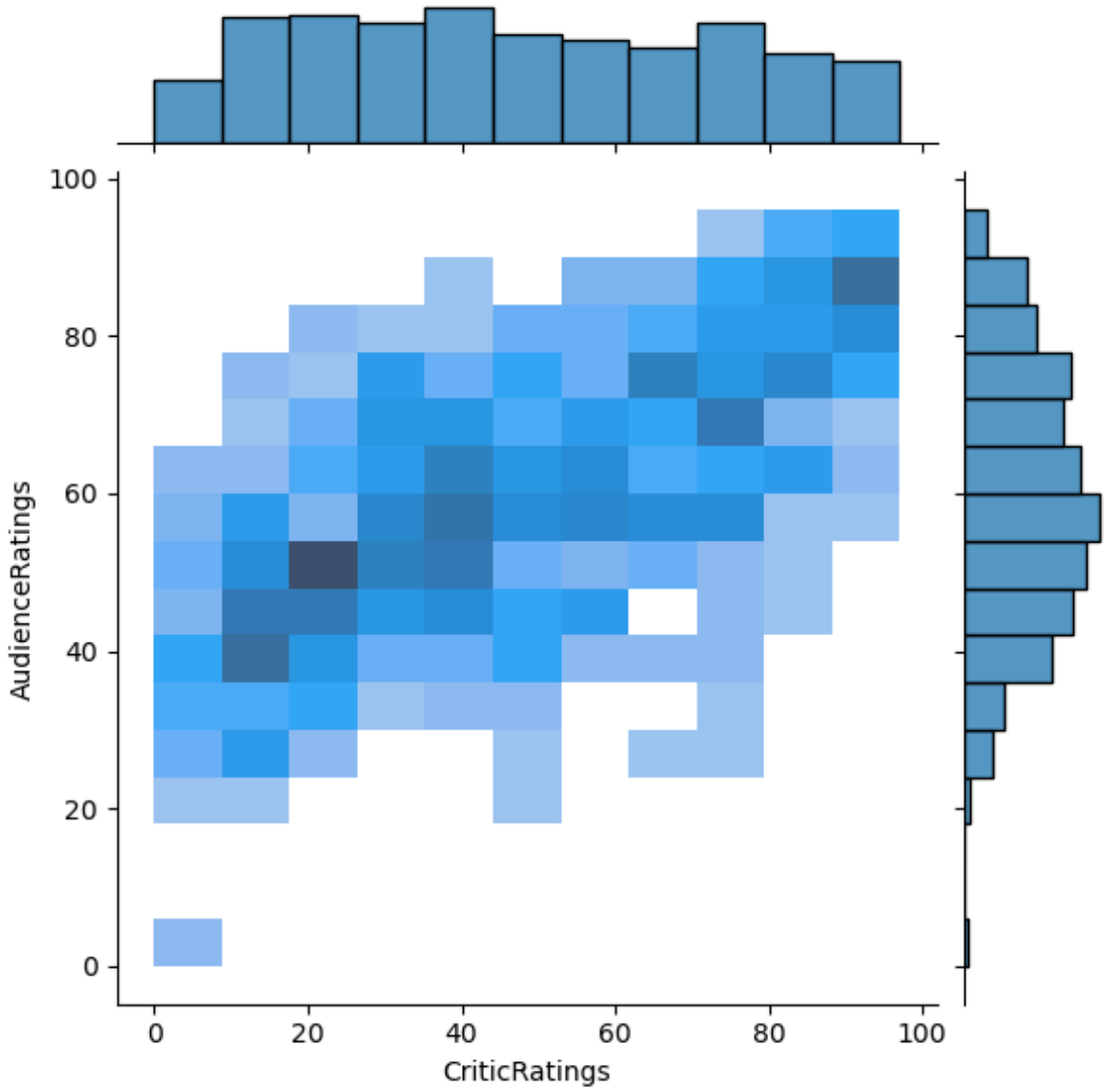
In [50]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind =
```



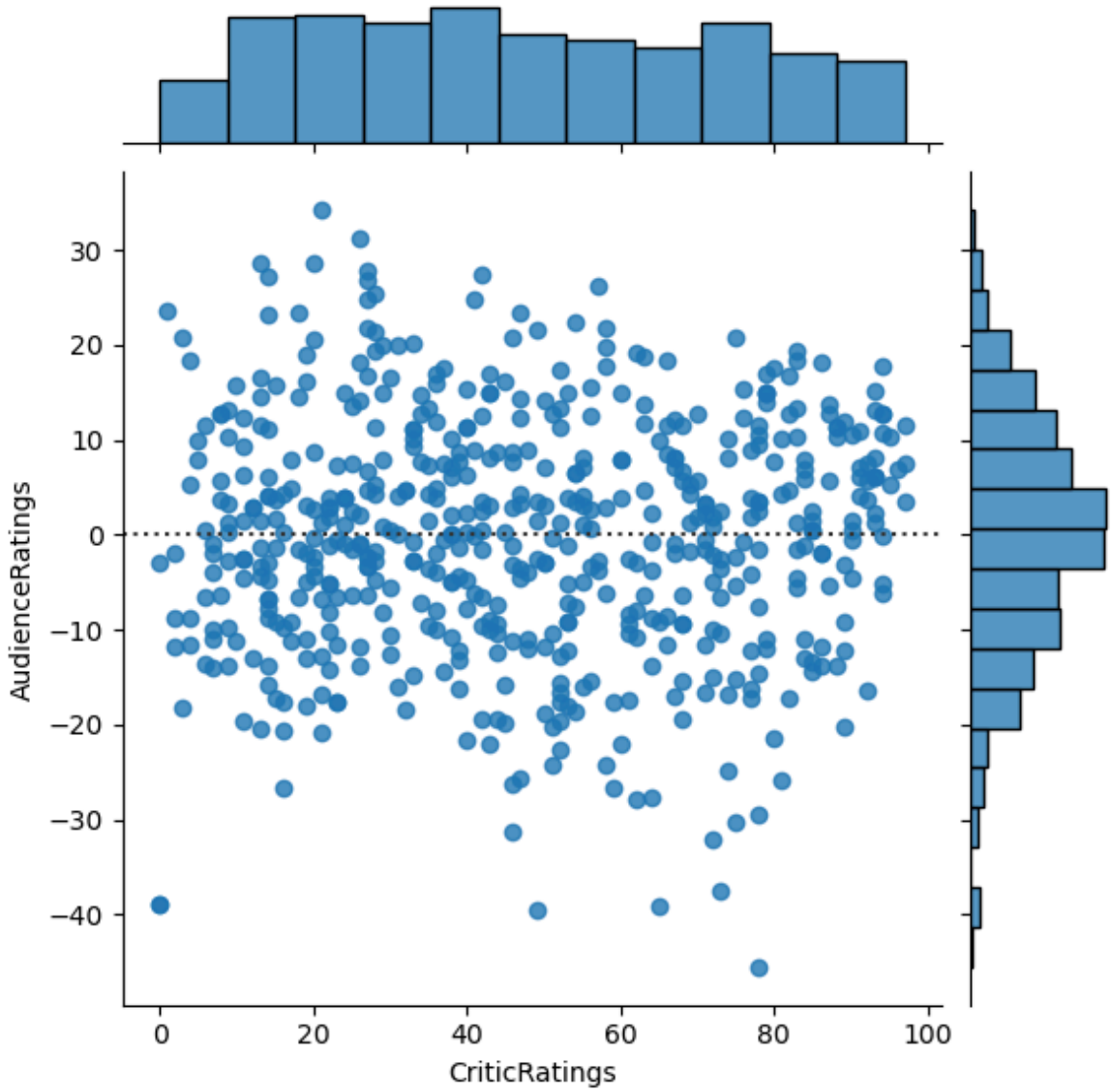
In [51]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind =
```



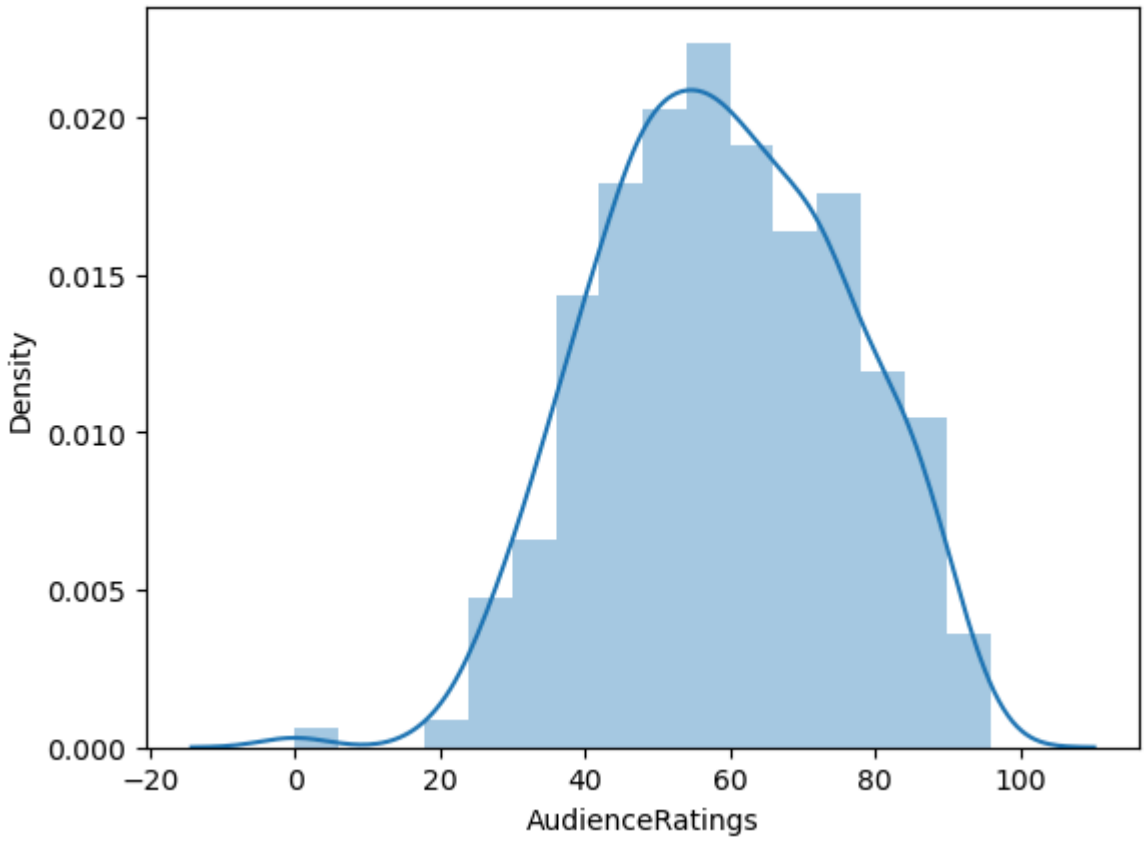
In [52]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind =
```



In [53]:
```
j = sns.jointplot(data = movies,x ='CriticRatings', y ='AudienceRatings',kind =
```

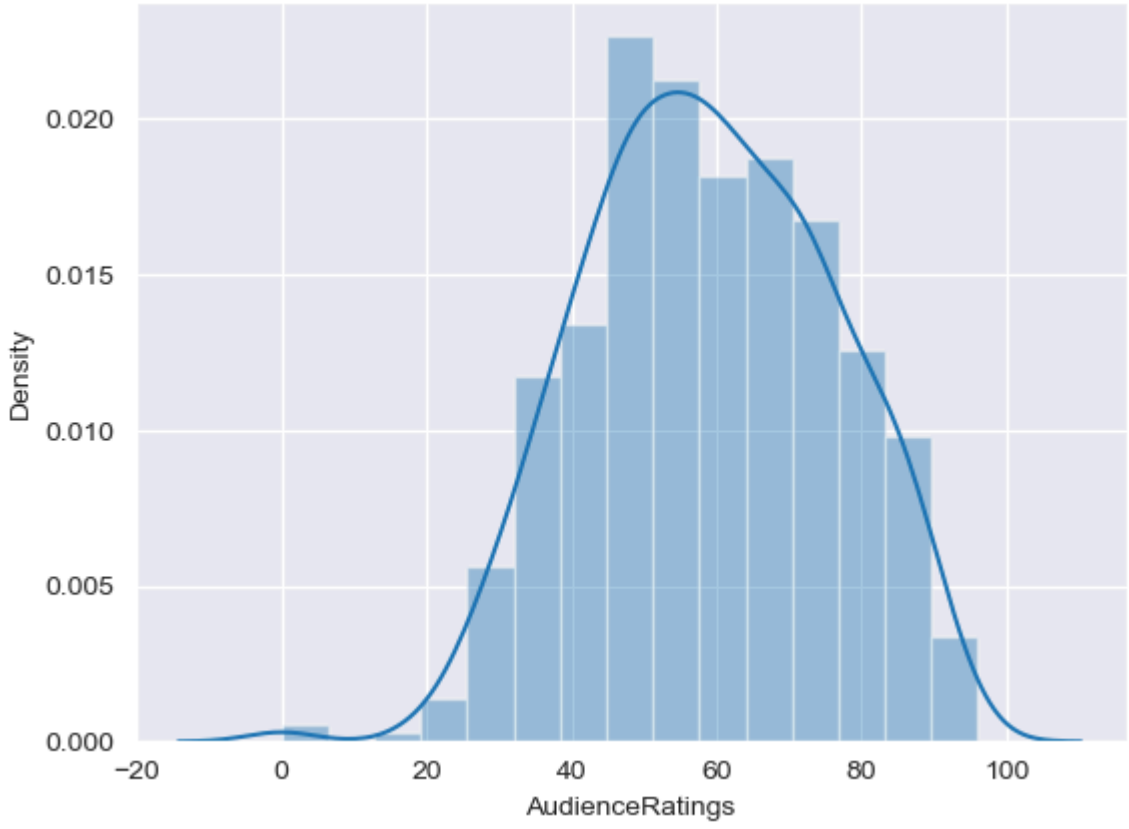


In [54]:
```
#Histograms

# <<< chat1

m1 = sns.distplot(movies.AudienceRatings)

#y - axis generated by seaborn automatically that is the powefull of seaborn gal
```

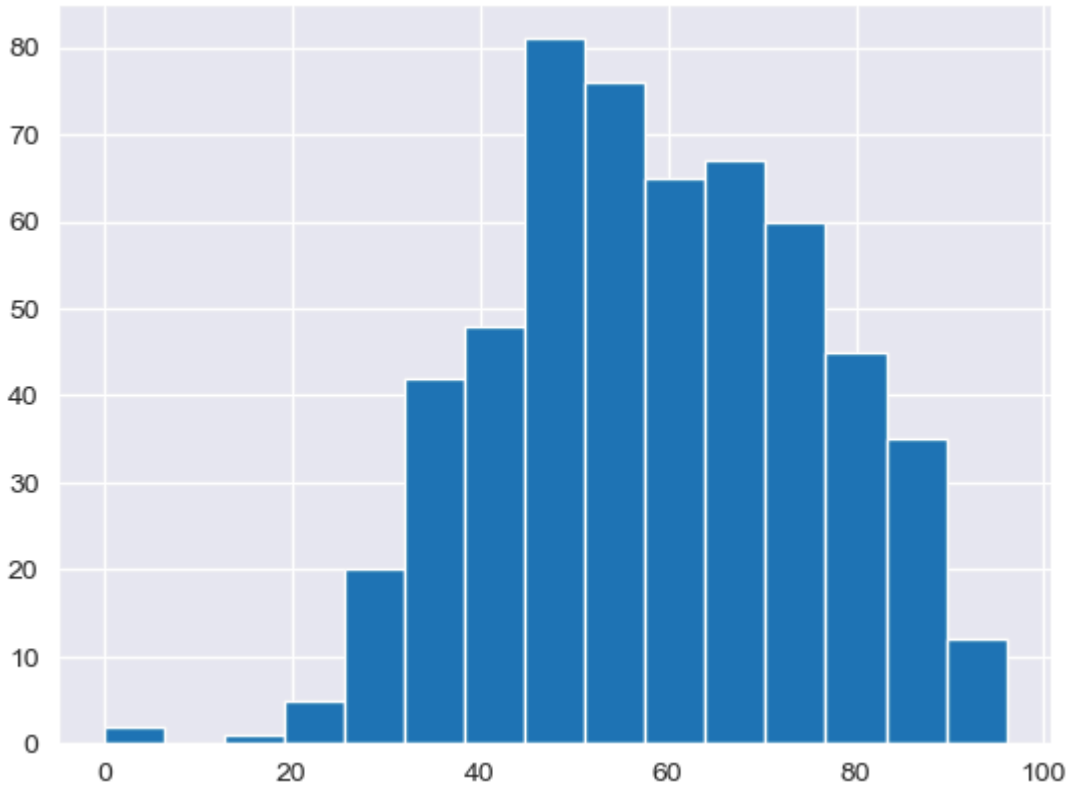


In [55]:
```
sns.set_style('darkgrid')
```

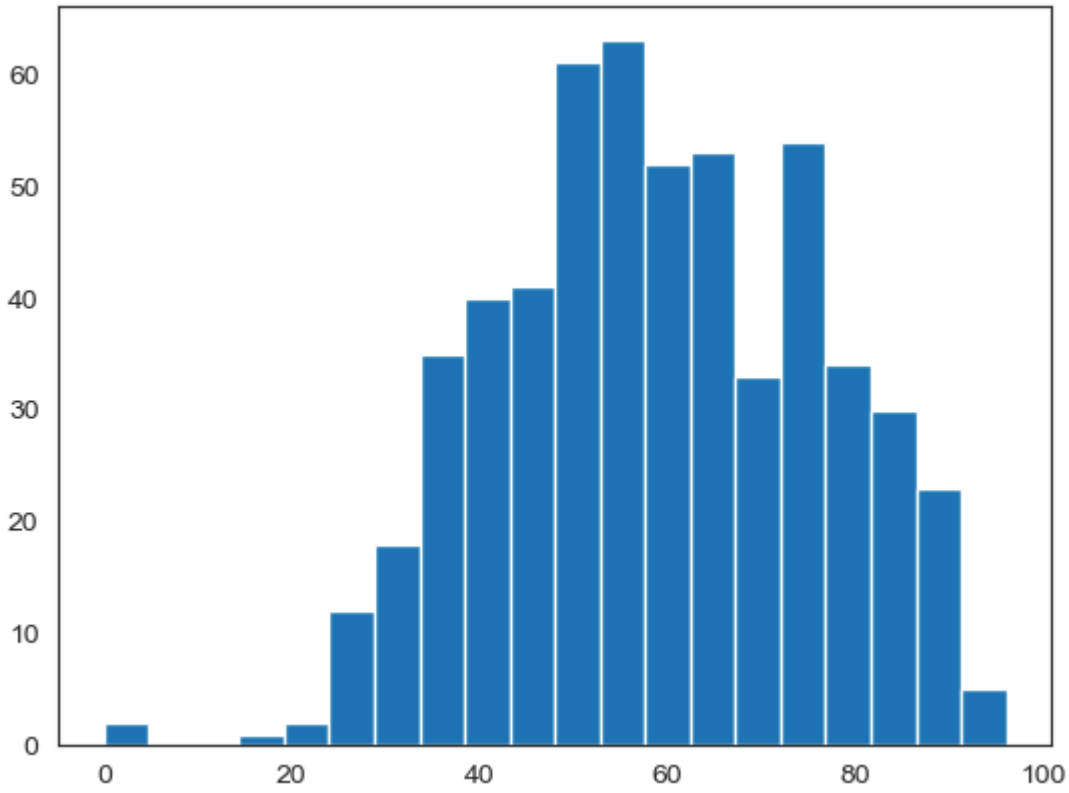
In [56]:
```
m2 = sns.distplot(movies.AudienceRatings, bins = 15)
```



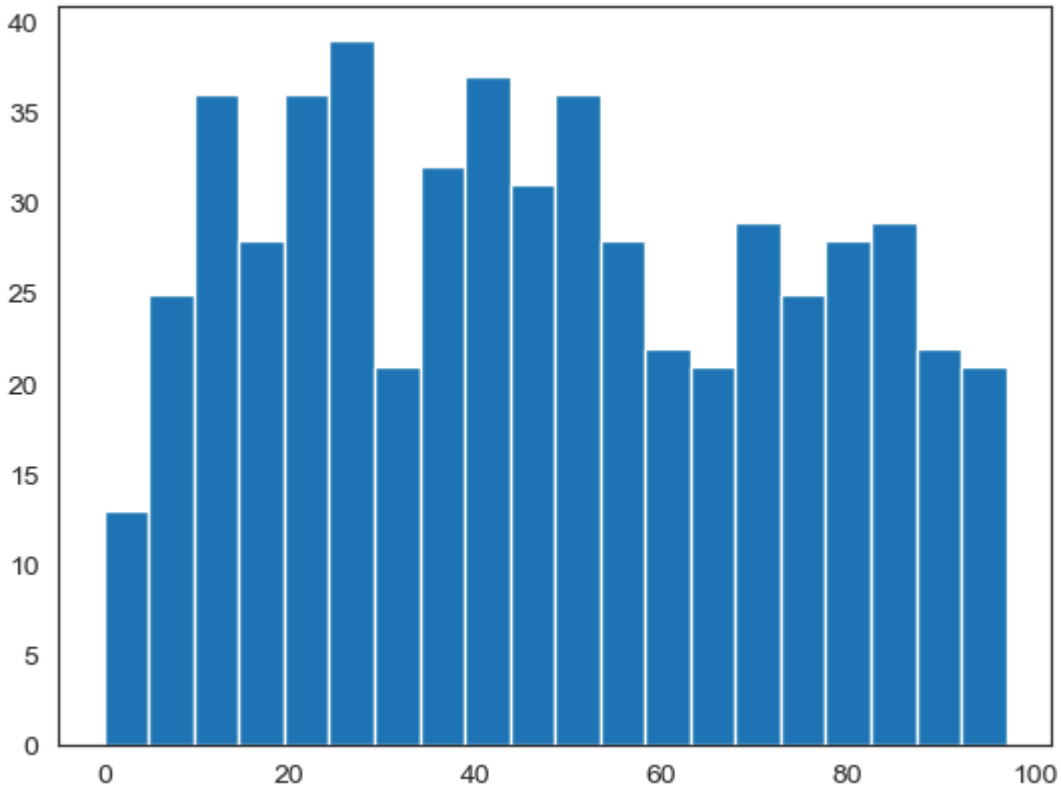
In [57]:
```
#sns.set_style('darkgrid')
n1 = plt.hist(movies.AudienceRatings, bins=15)
```



In [58]:
```
sns.set_style('white') #normal distribution & called as bell curve
n1 = plt.hist(movies.AudienceRatings, bins=20)
```

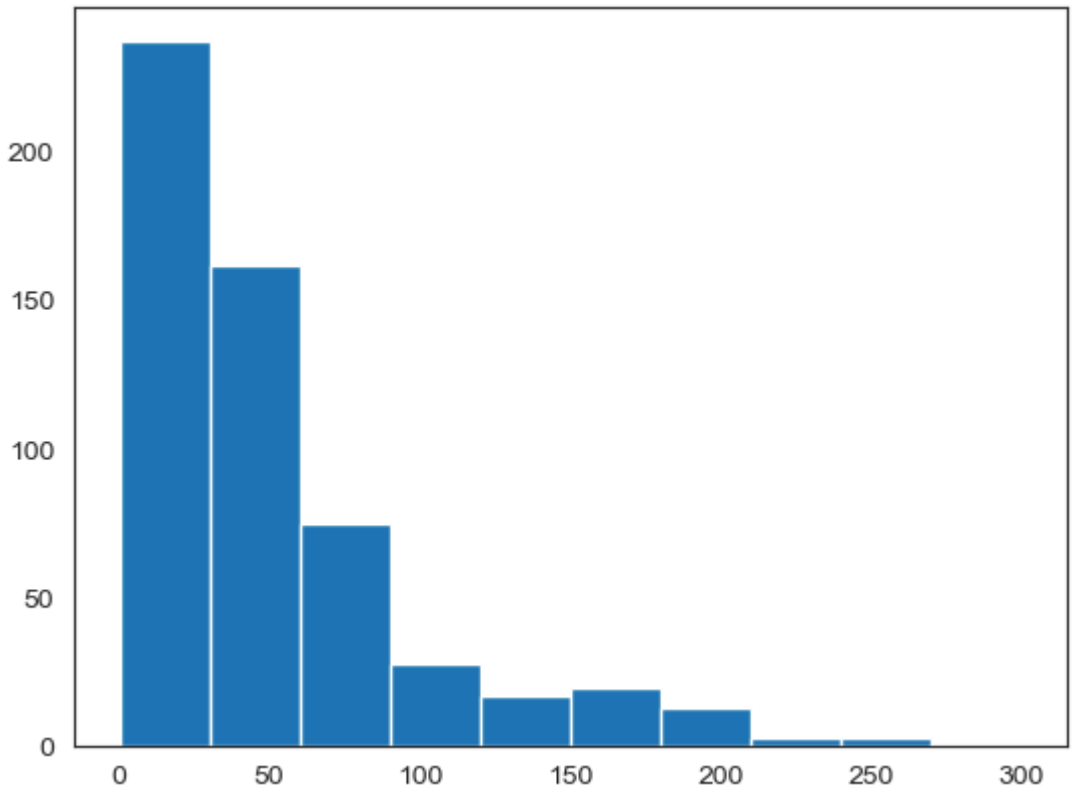


In [59]:
```
n1 = plt.hist(movies.CriticRatings, bins=20) #uniform distribution
```

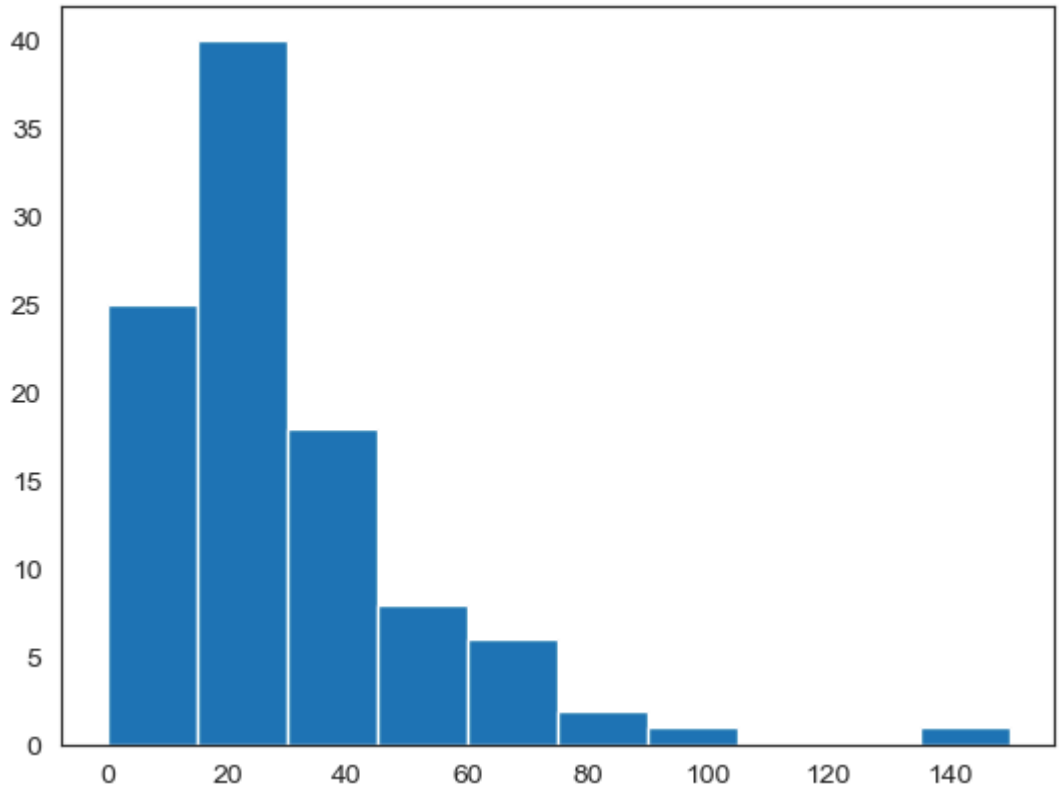


<<< chat - 2 # Creating stacked histograms & this is bit tough to understand

In [60]:
```
#h1 = plt.hist(movies.BudgetMillions)

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



In [62]:
```
plt.hist(movies[movies.Genre == 'Drama'].BudgetMillions)
plt.show()
```



In [63]:
```
movies.head()
```

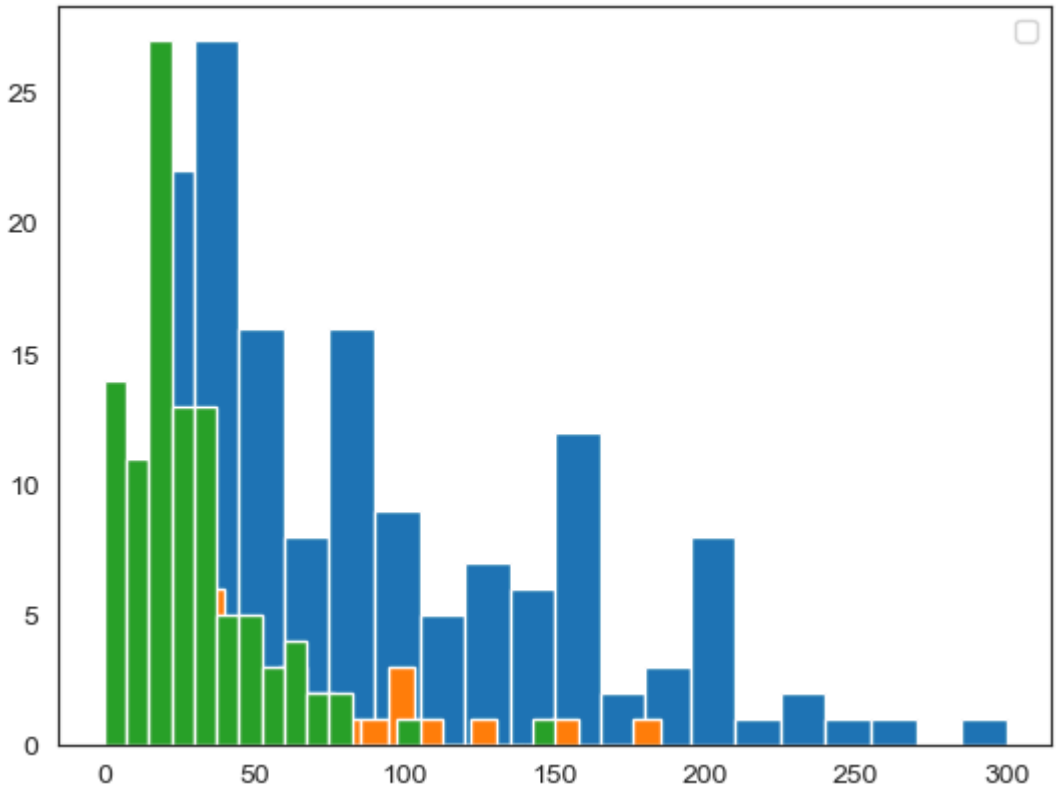
Out[63]:

	Film	Genre	CriticRatings	AudienceRatings	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

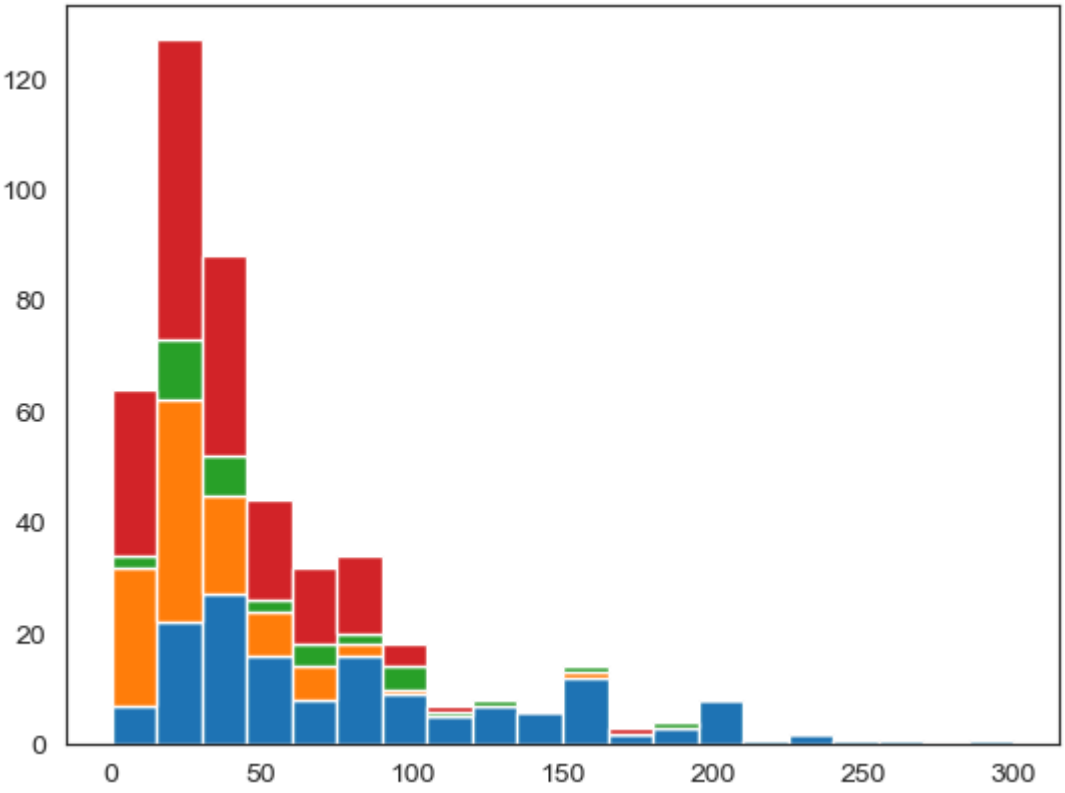
#movies.Genre.unique()

In [64]:
```
# 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 label sta
rt with an underscore are ignored when legend() is called with no argument.
```



In [65]:
```
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()
```

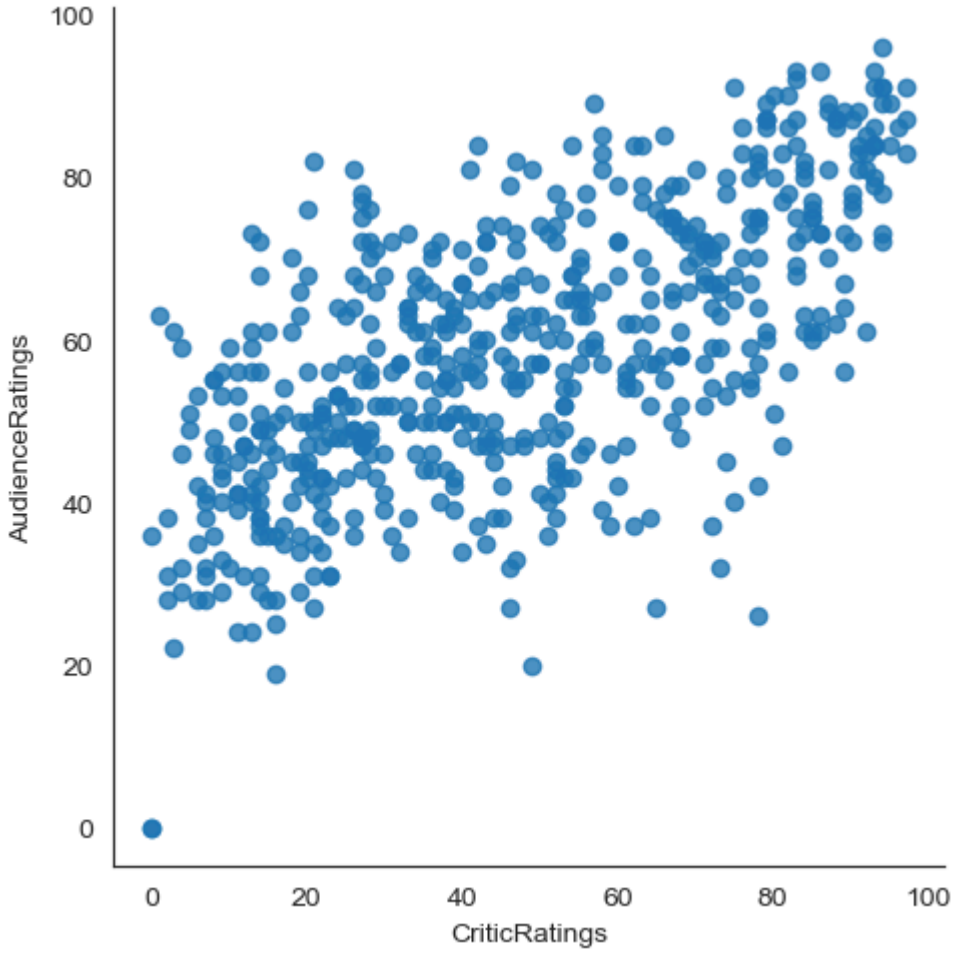


In [67]:
```
# if you have 100 categories you cannot copy & paste all the things
```

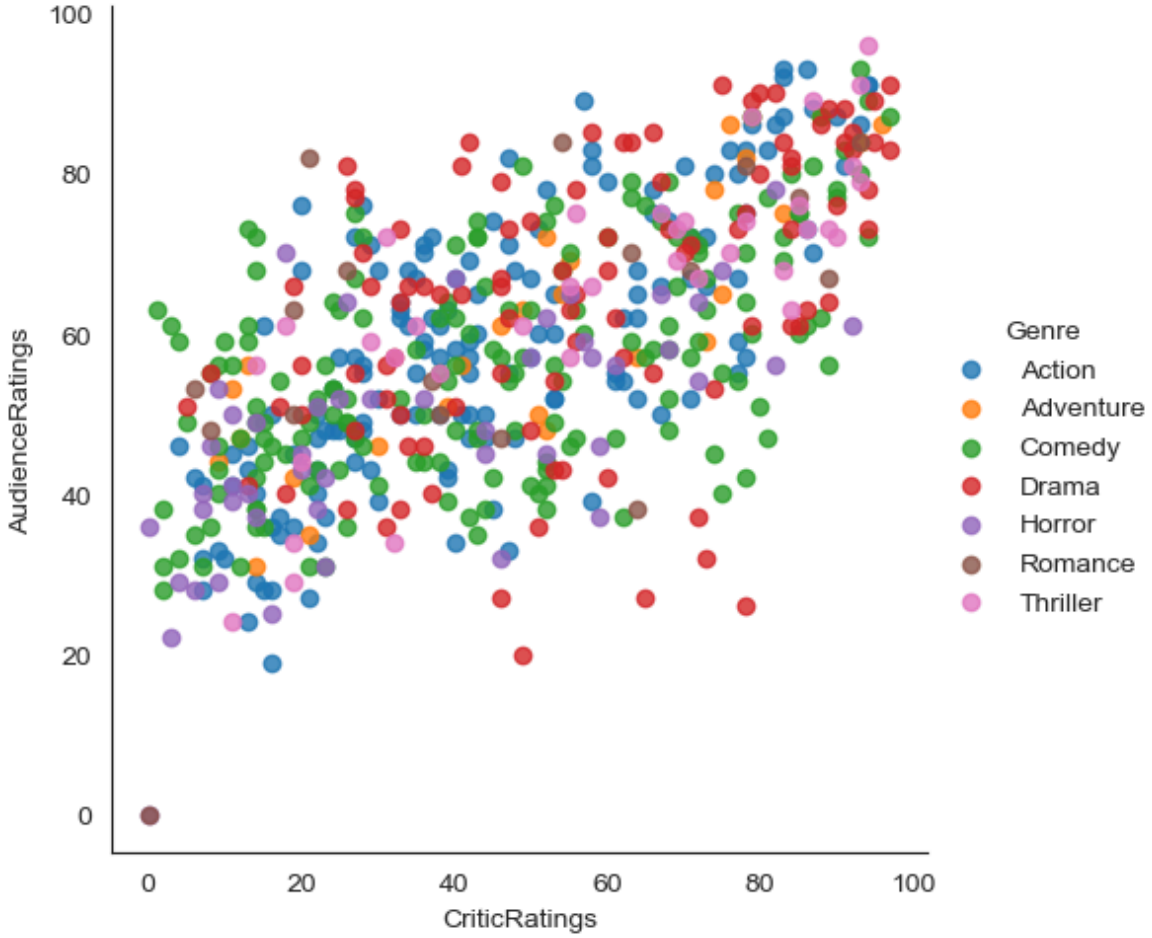
```
for gen in movies.Genre.cat.categories:
    print(gen)
```

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

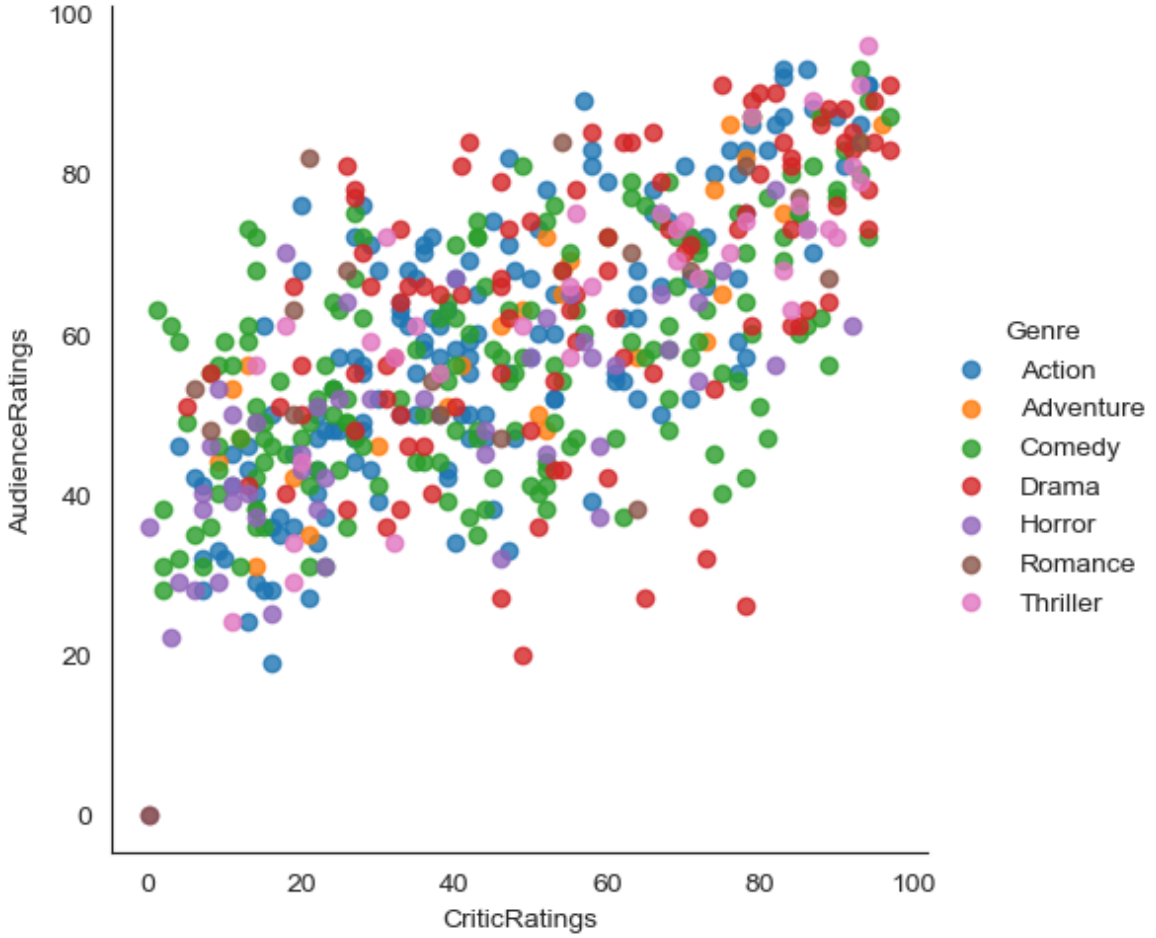
In [68]:
```
vis1 = sns.lmplot(data=movies, x='CriticRatings', y='AudienceRatings',\
                  fit_reg=False)
```



In [69]:
```
vis1 = sns.lmplot(data=movies, x='CriticRatings', y='AudienceRatings',\
                  fit_reg=False, hue = 'Genre')
```

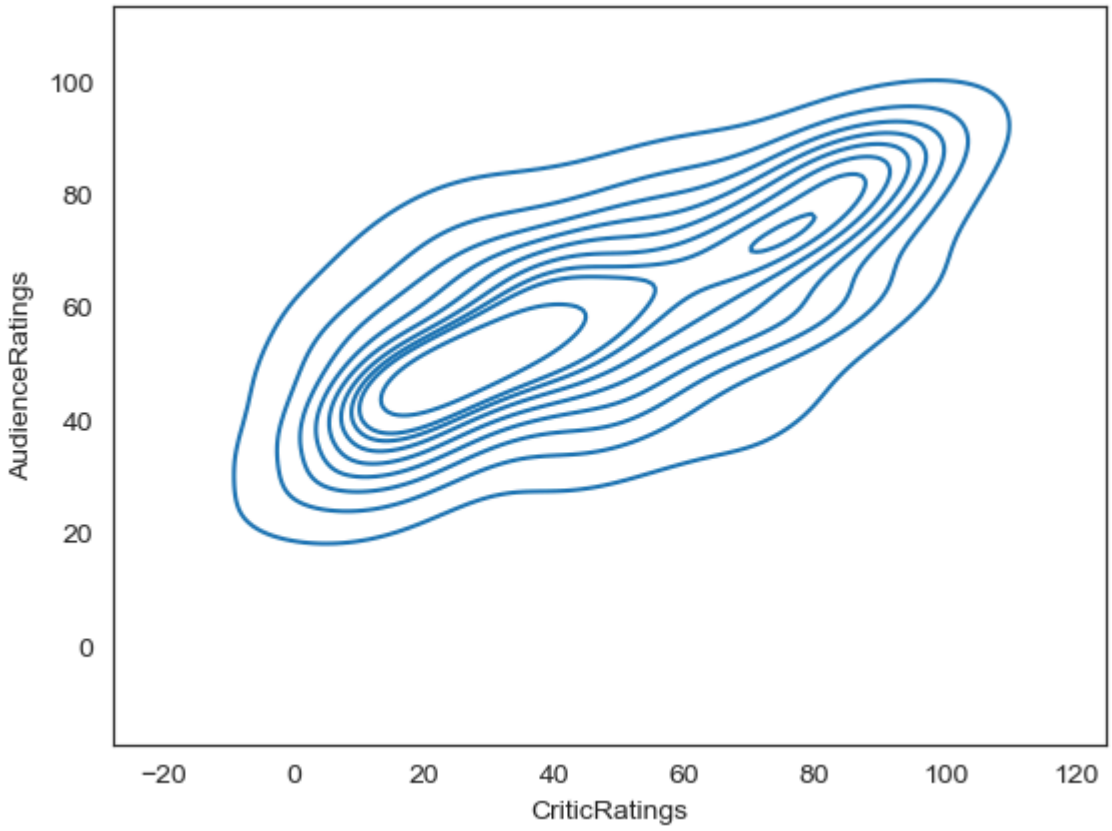


In [71]:
```
vis1 = sns.lmplot(data=movies, x='CriticRatings', y='AudienceRatings',\
                  fit_reg=False, hue = 'Genre',aspect=1)
```

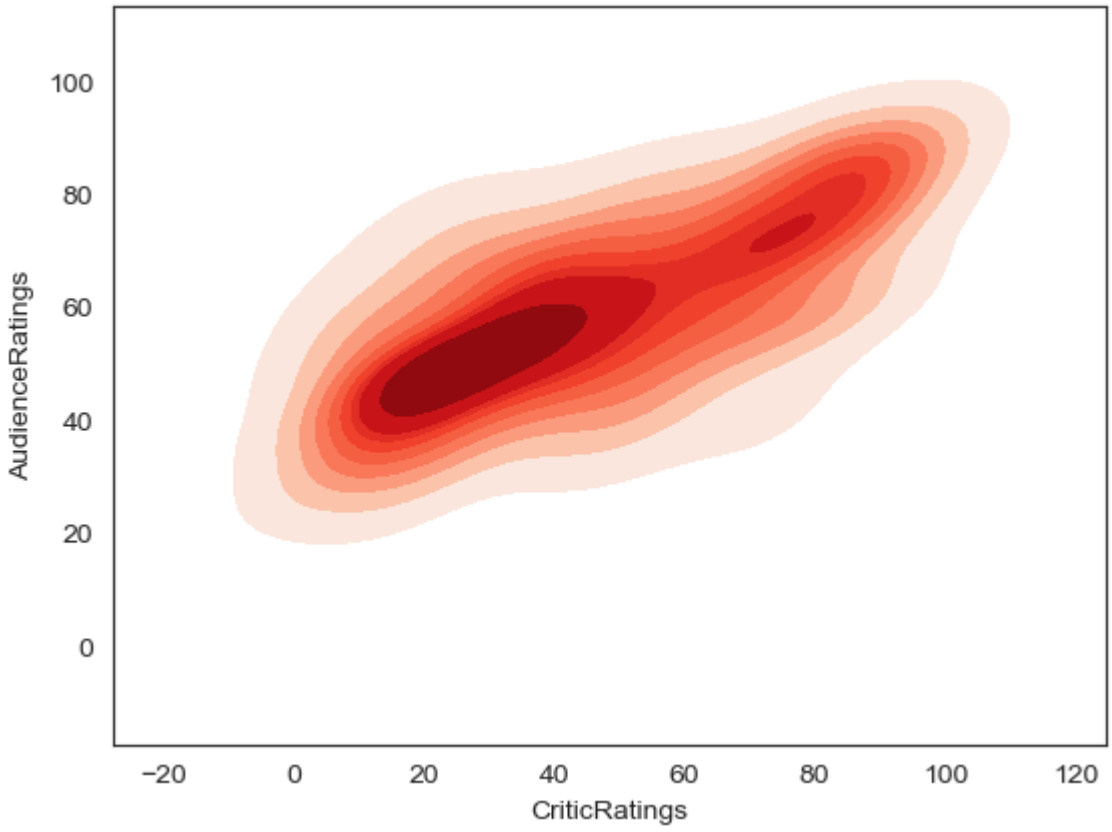


Kernal Density Estimate plot (KDE PLOT) # how can i visulize audience rating & critics rating . using scatterplot

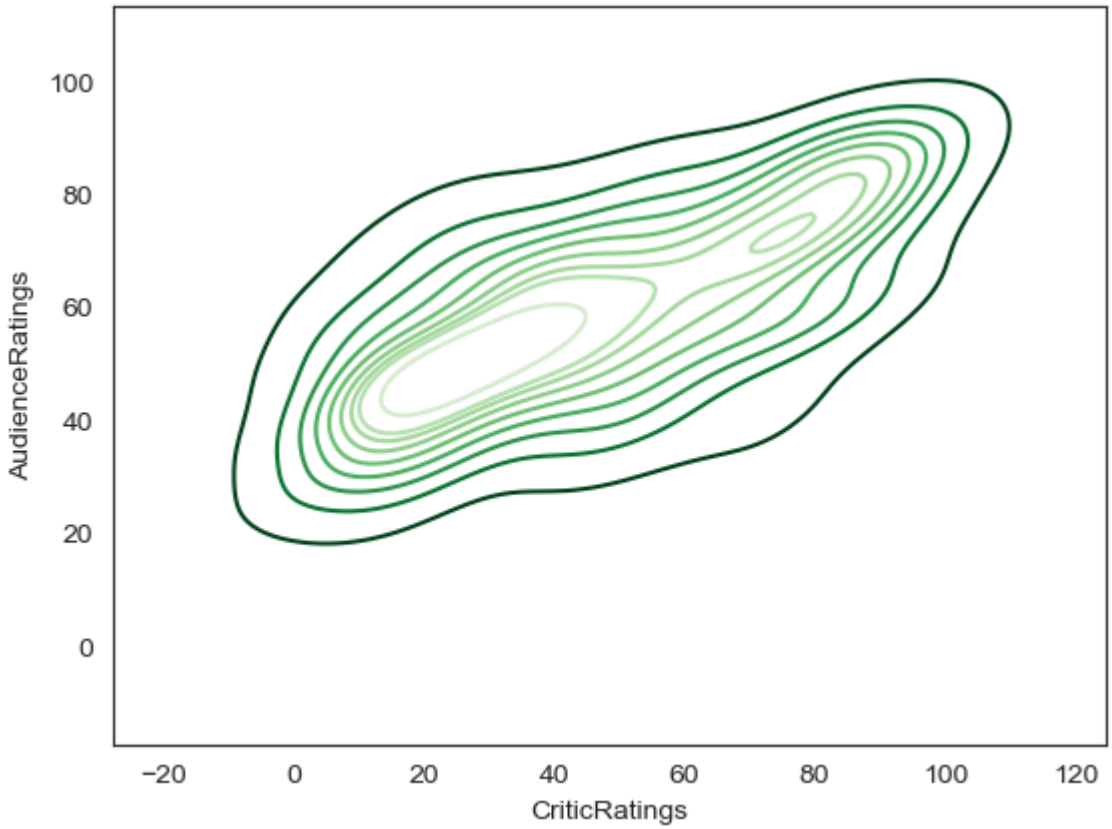
In [72]:
```
k1 = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings')
# where do u find more density and how density is distibuted across from the the
# center point is kernal this is calld KDE & insteade of dots it visualize like
# we can able to clearly see the spread at the audience ratings
```



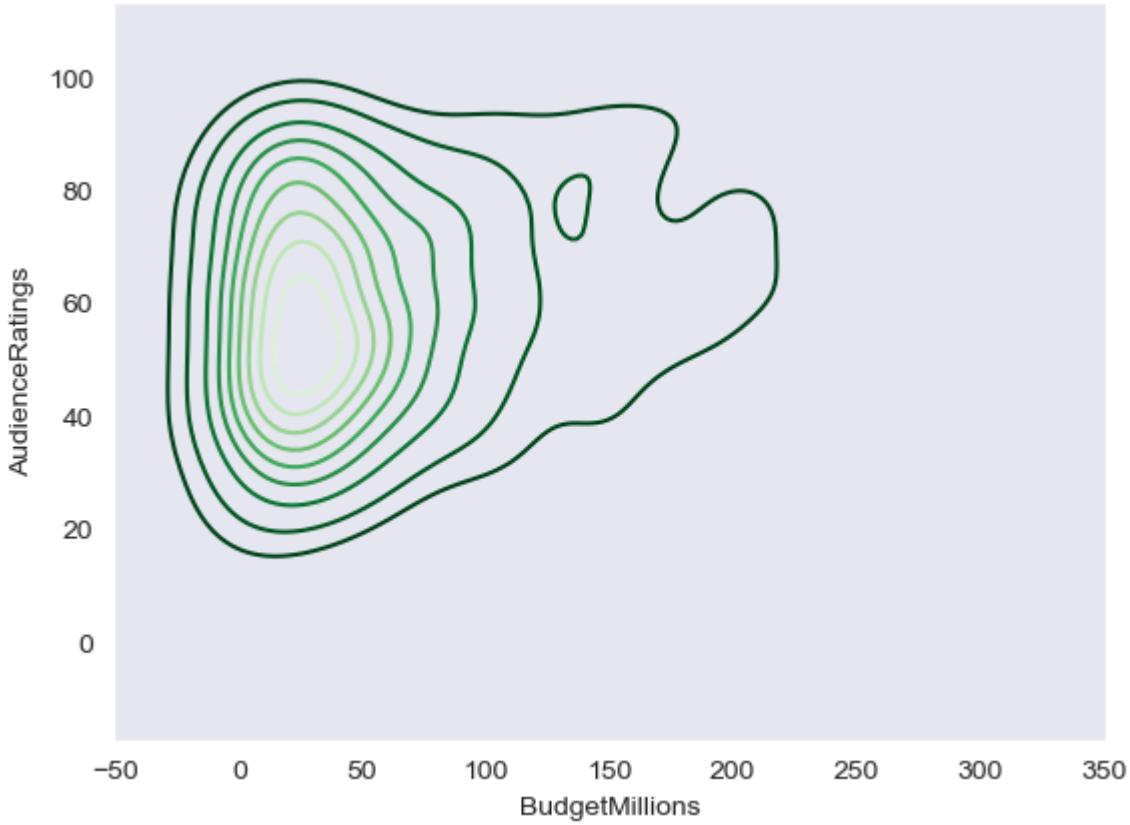
In [73]:
```
k1 = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',shade = True,
```



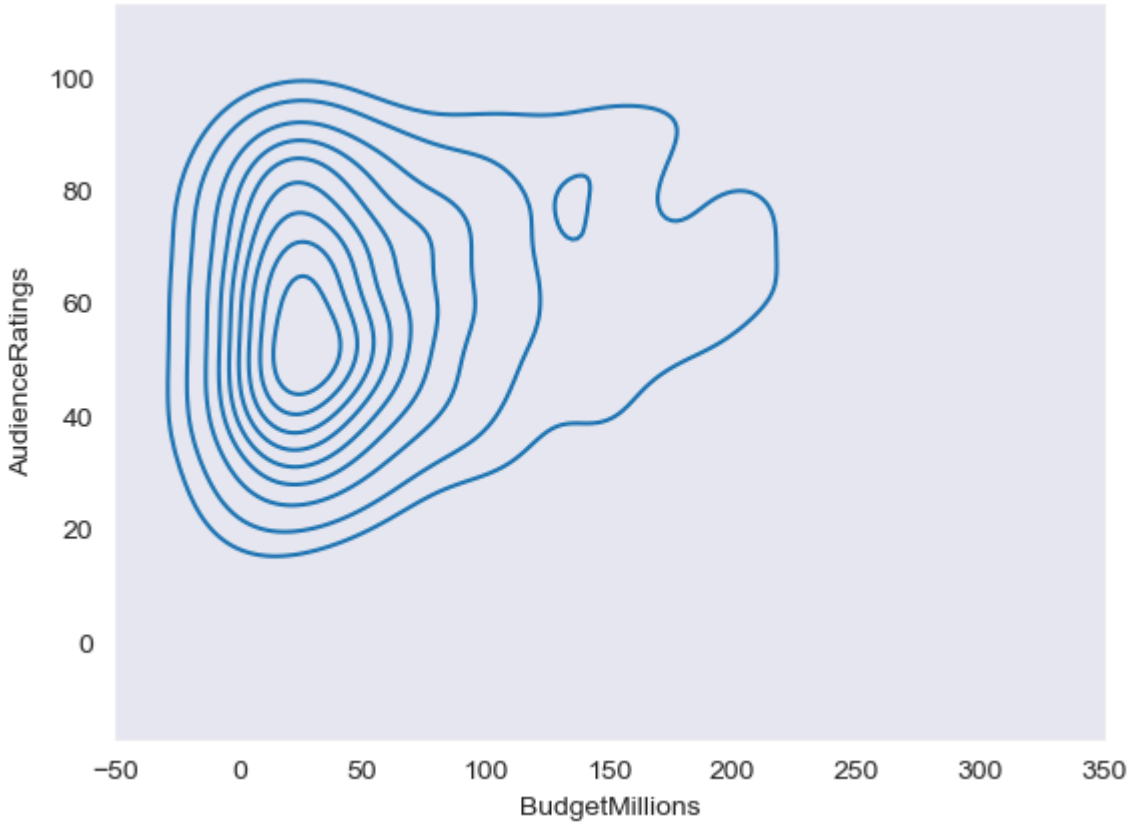
In [74]:
```
k2 = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',shade_lowest=
```



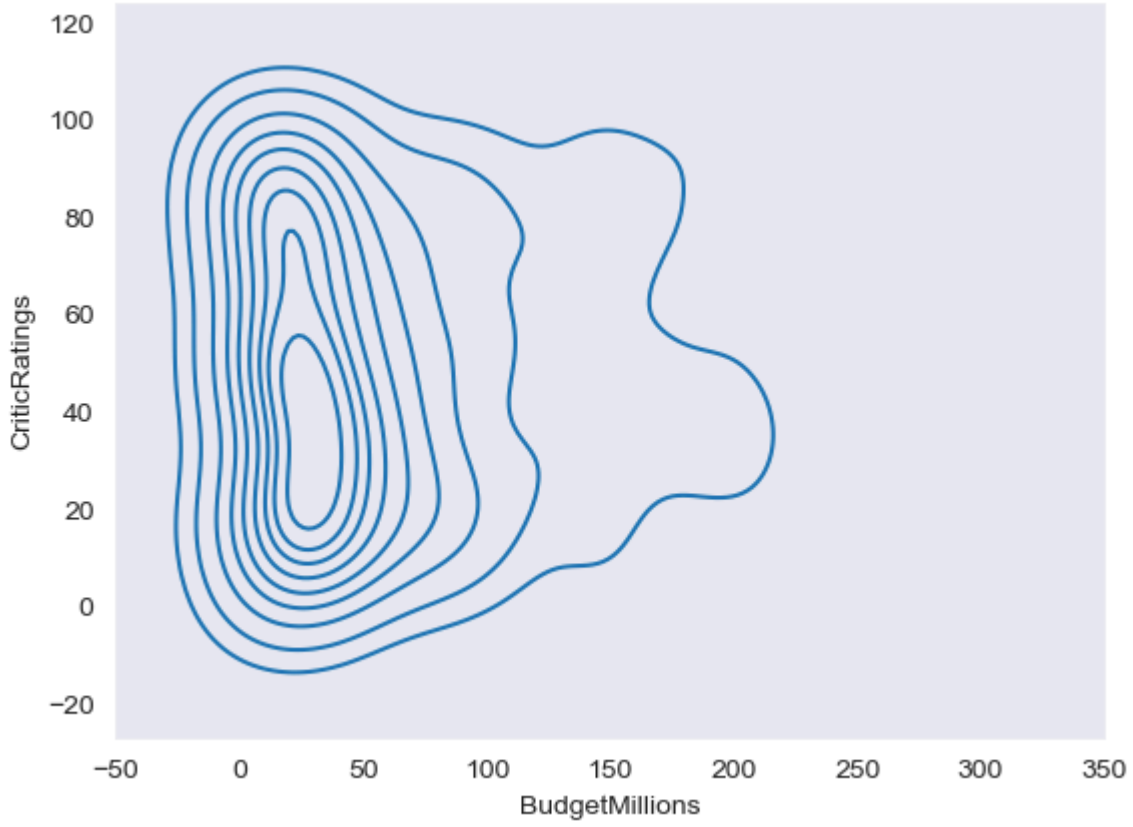
In [75]:
```
sns.set_style('dark')
k1 = sns.kdeplot(data=movies,x='BudgetMillions',y='AudienceRatings',shade_lowest
```



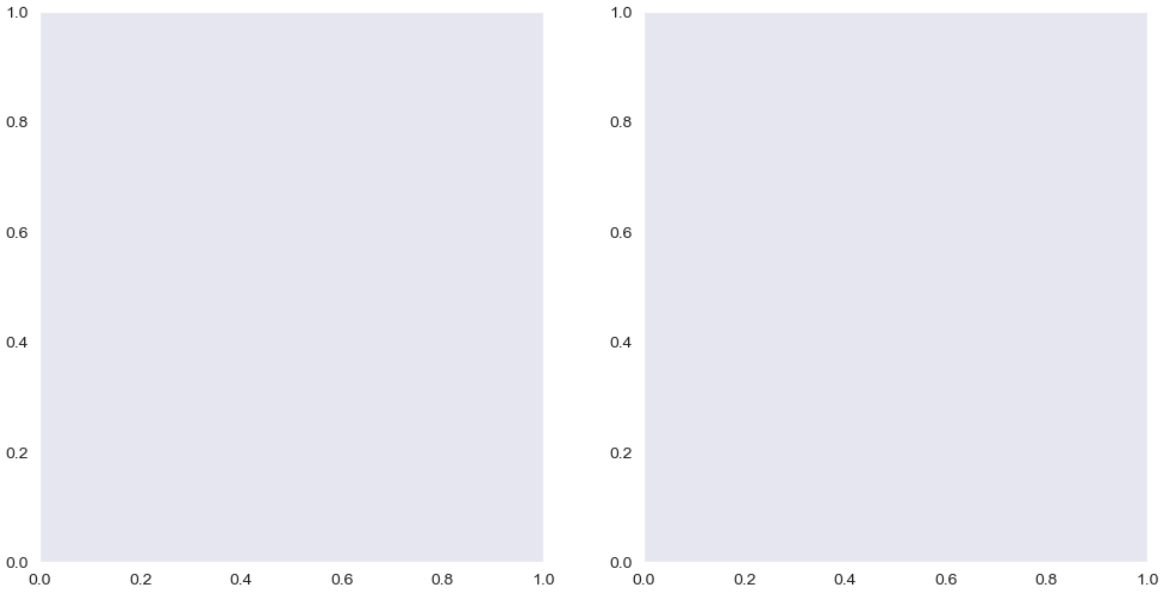
In [77]:
```
sns.set_style('dark')
k1 = sns.kdeplot(data=movies,x='BudgetMillions',y='AudienceRatings')
```



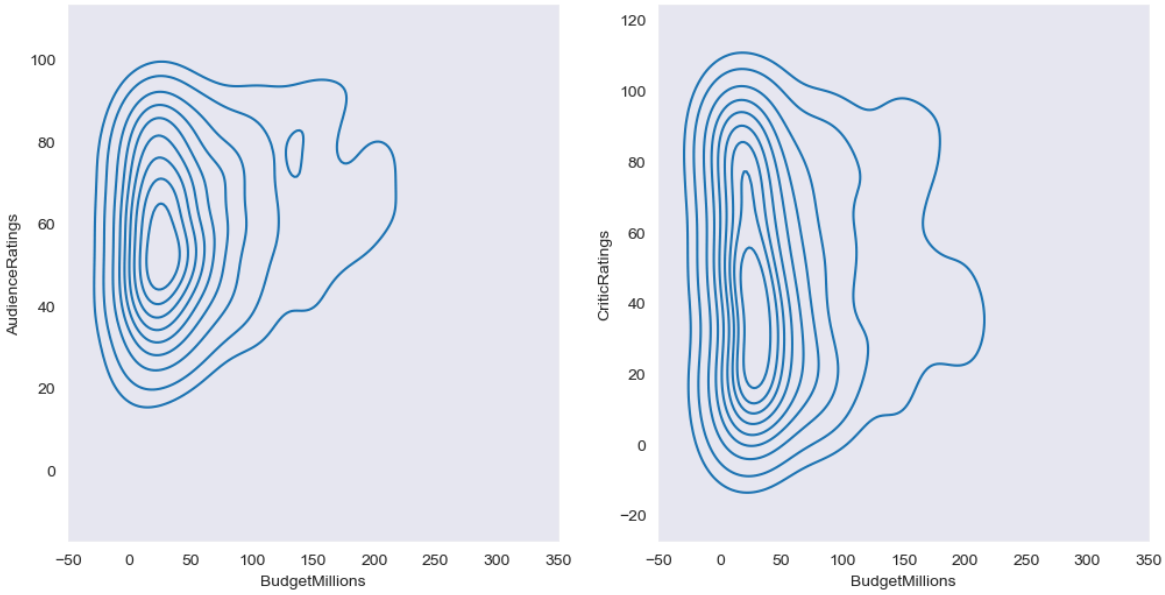
In [78]:
```
k2 = sns.kdeplot(data=movies,x='BudgetMillions',y='CriticRatings')
```



In [79]:
```
#subplots
f, ax = plt.subplots(1,2, figsize =(12,6))
#f, ax = plt.subplots(3,3, figsize =(12,6))
```



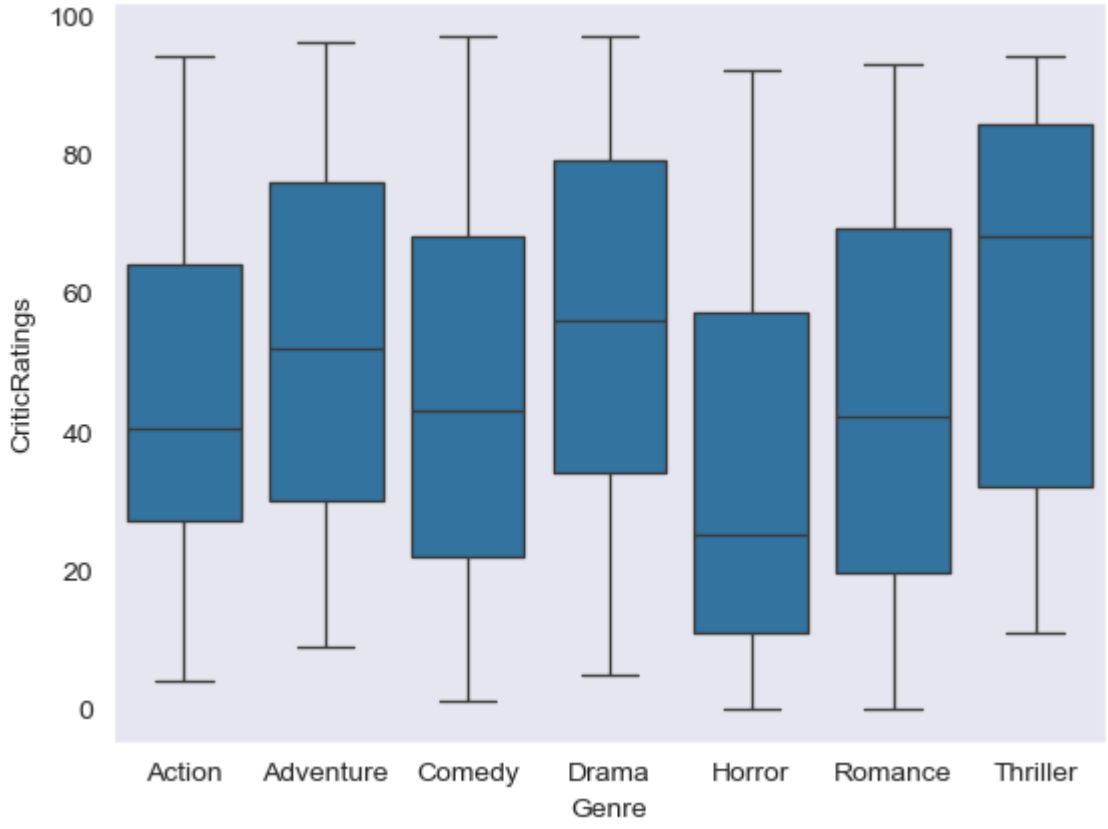
In [81]:
```
f, axes = plt.subplots(1,2, figsize =(12,6))

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



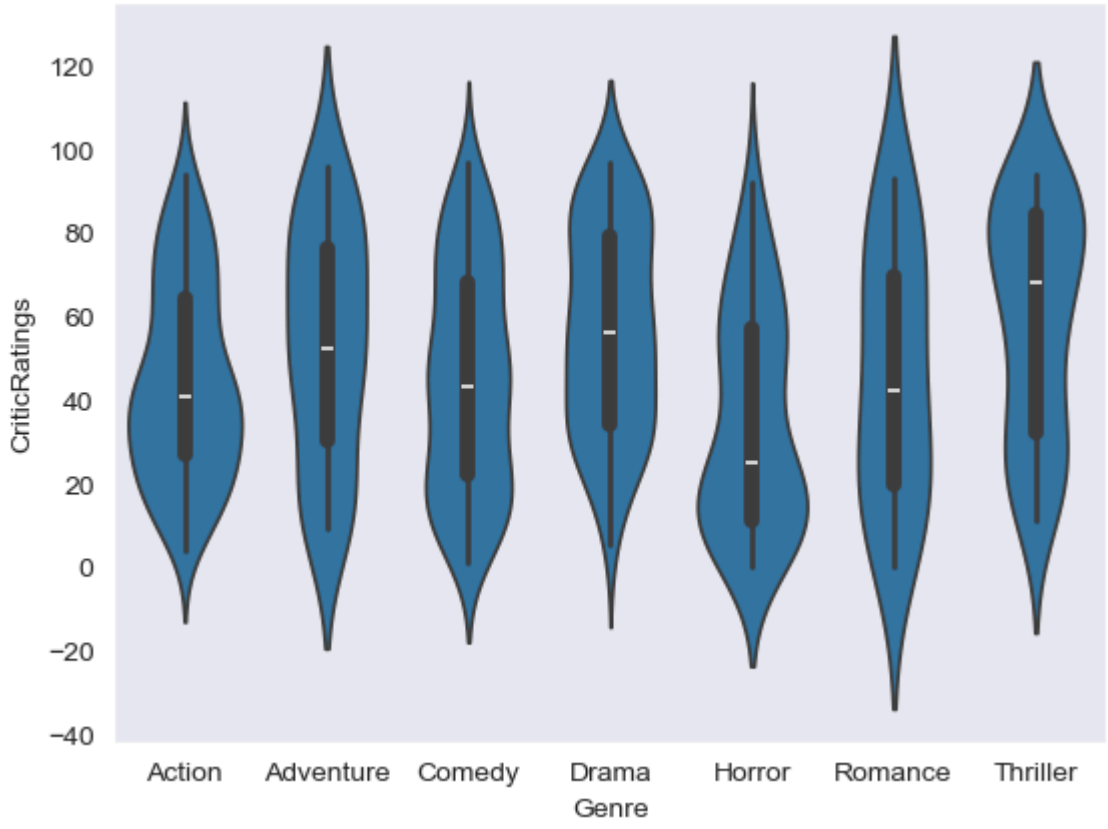
In [82]:
```
axes
```

Out[82]:
```
array([<Axes: xlabel='BudgetMillions', ylabel='AudienceRatings'>,
       <Axes: xlabel='BudgetMillions', ylabel='CriticRatings'>],
      dtype=object)
```

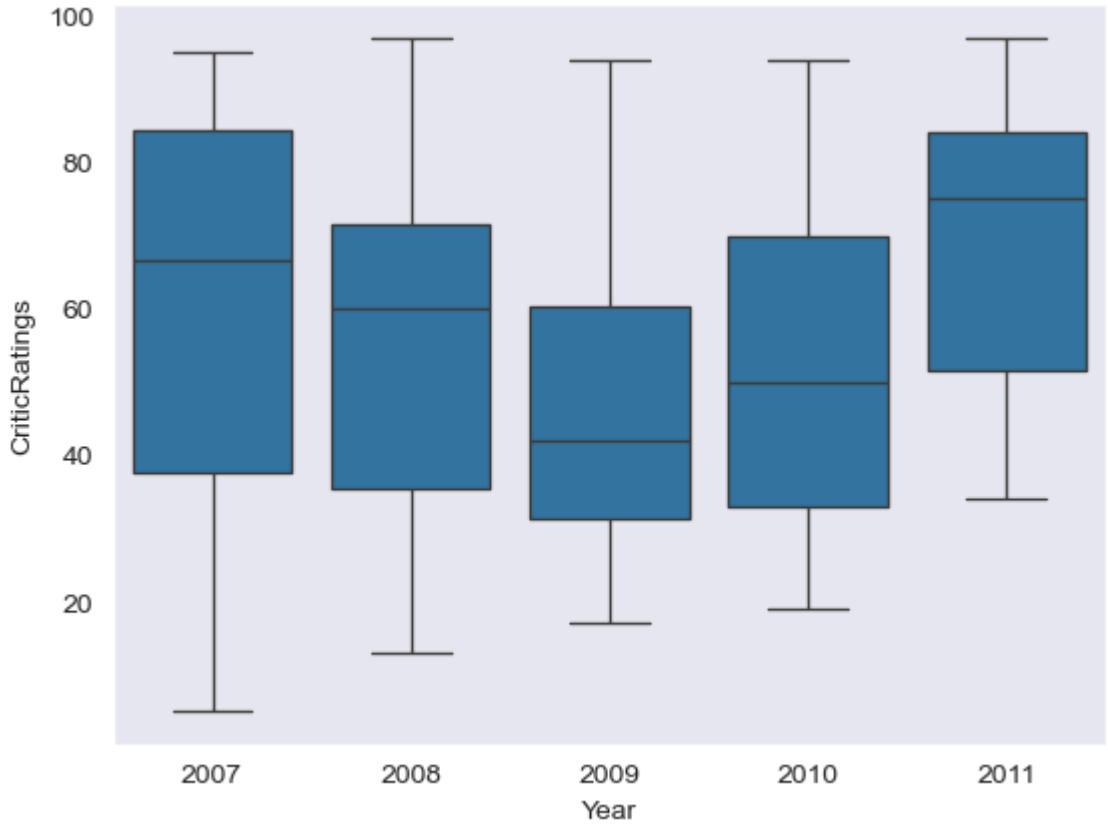
In [84]:
```
#Box plots -
w = sns.boxplot(data=movies, x='Genre', y = 'CriticRatings')
```



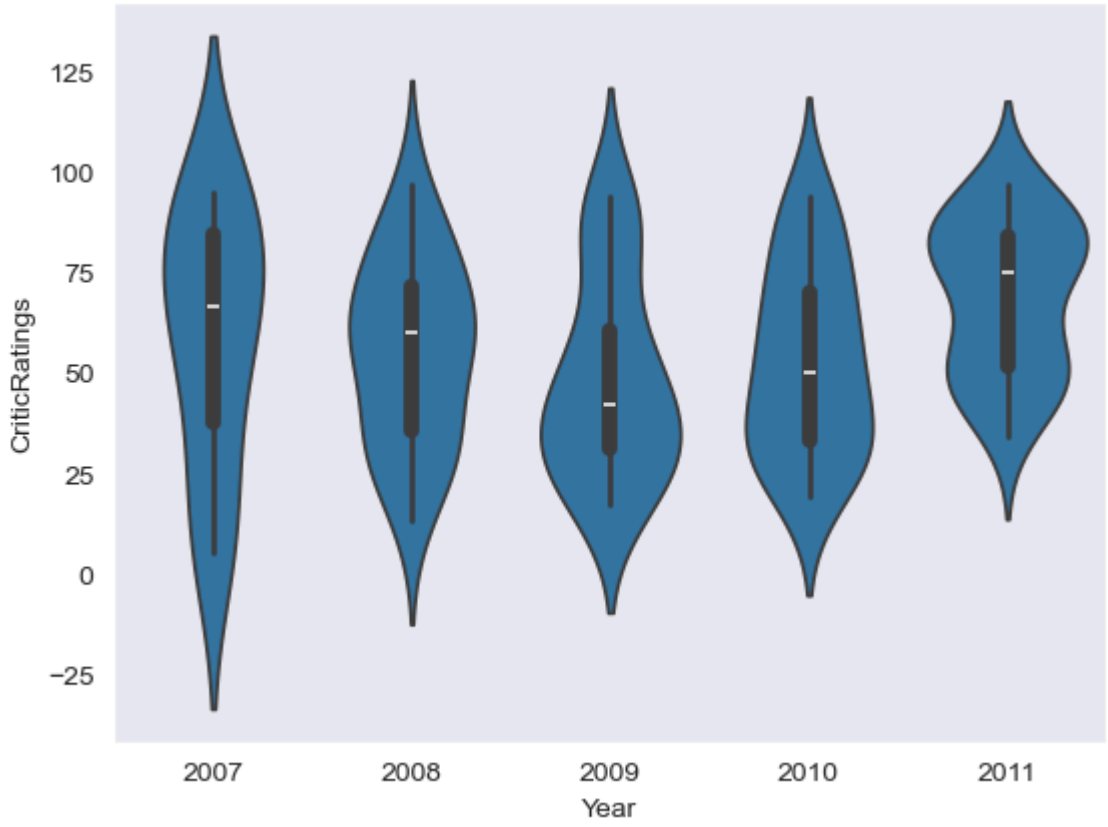
In [85]:
```
#violin plot
z = sns.violinplot(data=movies, x='Genre', y = 'CriticRatings')
```



In [86]:
```
w1 = sns.boxplot(data=movies[movies.Genre == 'Drama'], x='Year', y = 'CriticRati
```



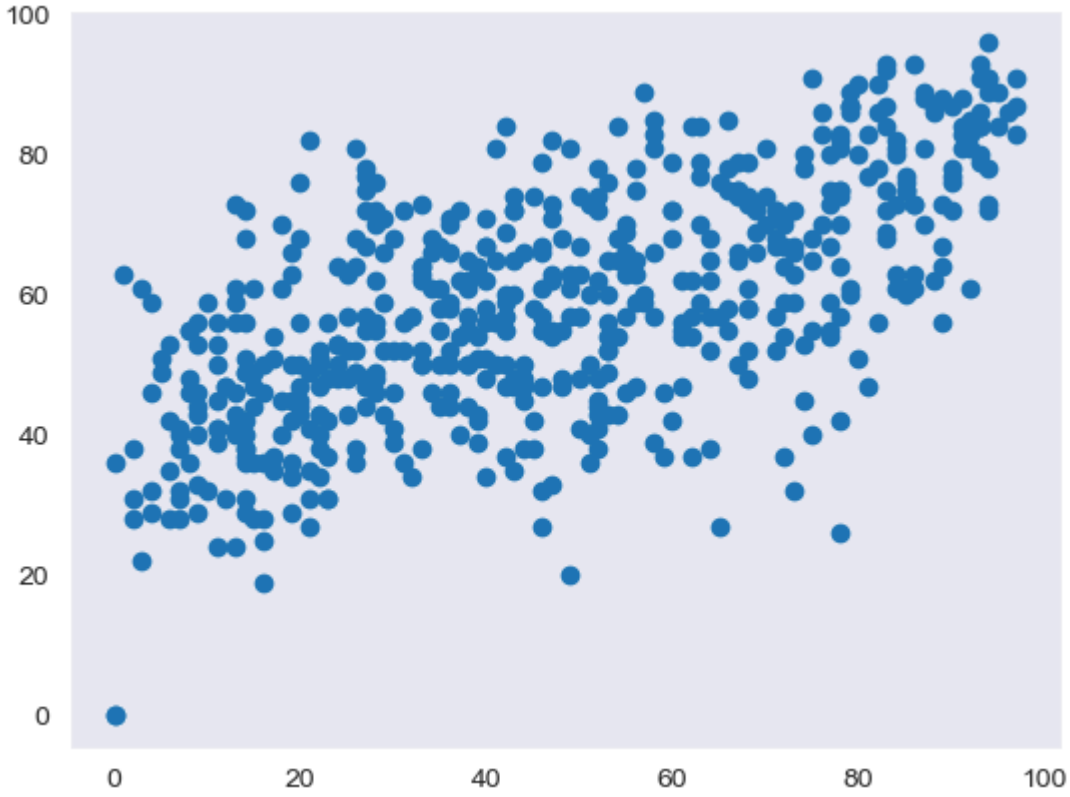
In [88]:
```
z = sns.violinplot(data=movies[movies.Genre == 'Drama'], x='Year', y = 'CriticRa
```



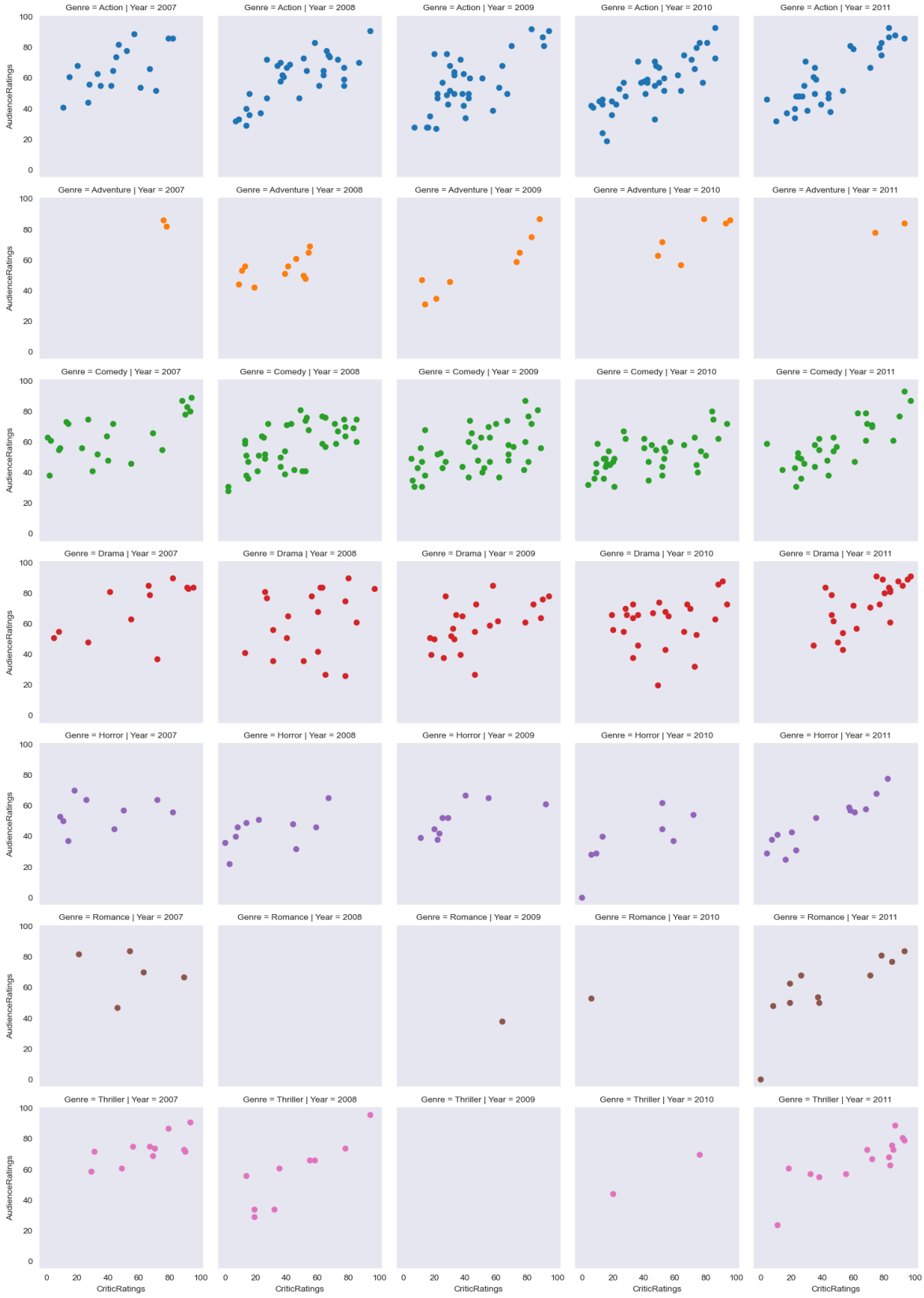
In [91]:
```
# Createing a Facet grid
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre') #kind of s
```

In [93]:
```
plt.scatter(movies.CriticRatings,movies.AudienceRatings)
```

Out[93]: <matplotlib.collections.PathCollection at 0x27a67402690>

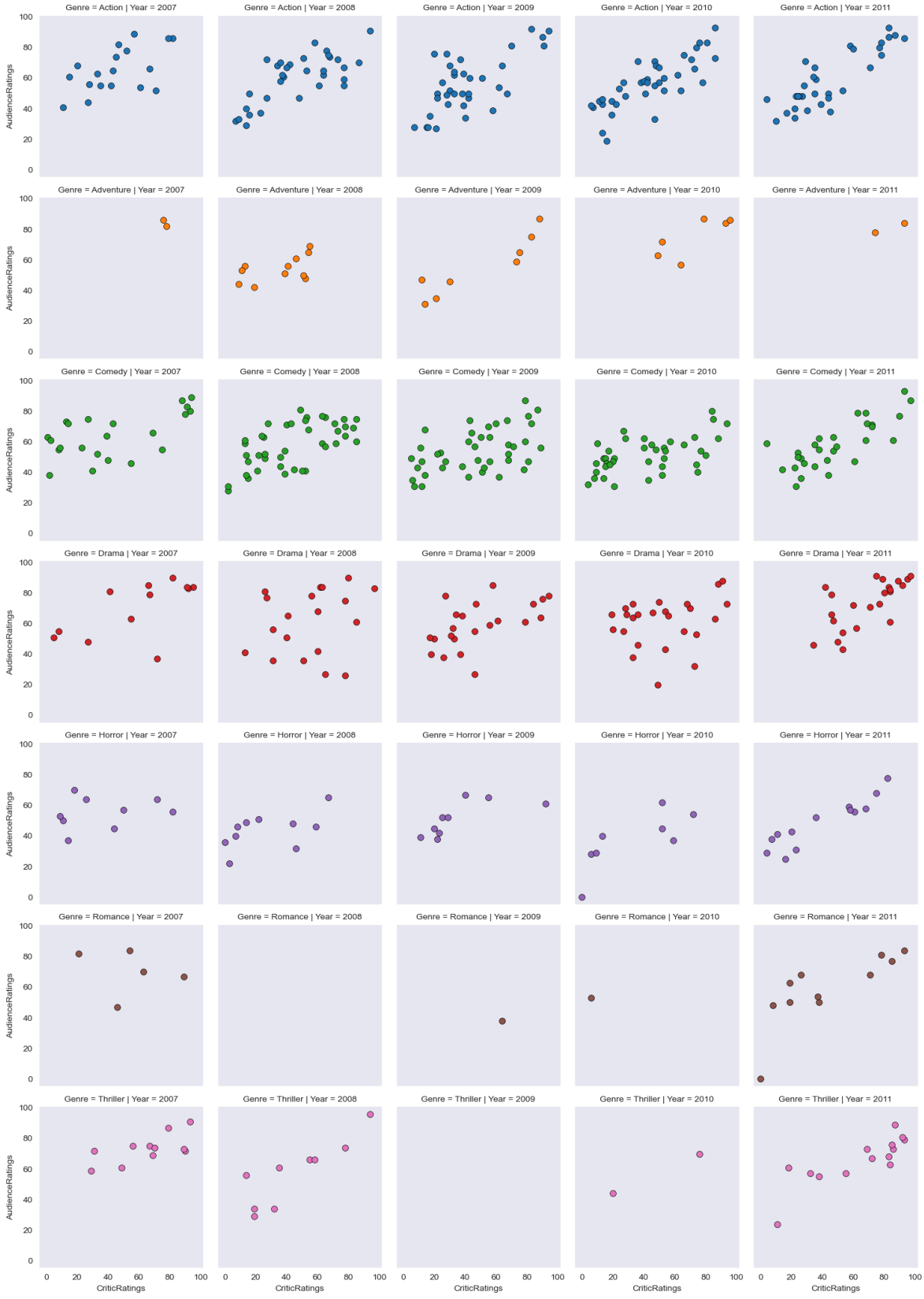


In [94]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.scatter, 'CriticRatings', 'AudienceRatings' ) #scatterplots are ma
```



In [96]:
```
# you can populated any type of chat.
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.hist, 'BudgetMillions') #scatterplots are mapped in facetgrid
```

In [98]:
```
#
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 ) #scatterplots
```



In [100…

```
movies
```

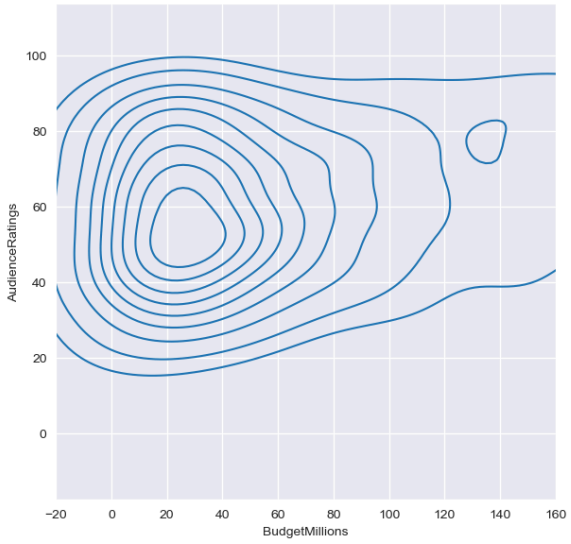
Out[100…

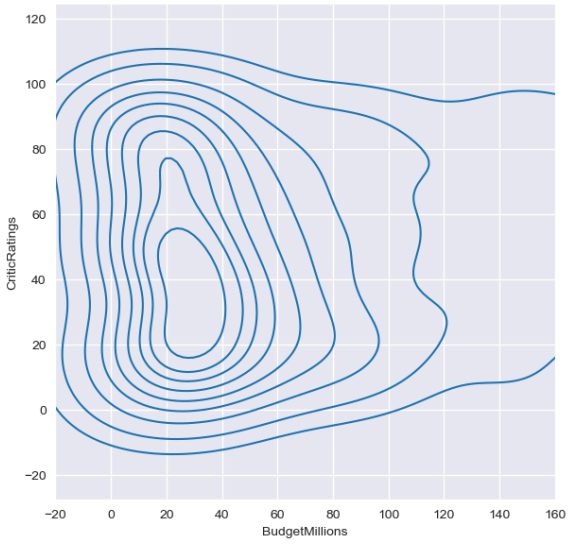
	Film	Genre	CriticRatings	AudienceRatings	BudgetMillions	Year
0	(500) Days of Summer	Comedy	87	81	8	2009
1	10,000 B.C.	Adventure	9	44	105	2008
2	12 Rounds	Action	30	52	20	2009
3	127 Hours	Adventure	93	84	18	2010
4	17 Again	Comedy	55	70	20	2009
...	...	...	...	...	...	...
554	Your Highness	Comedy	26	36	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

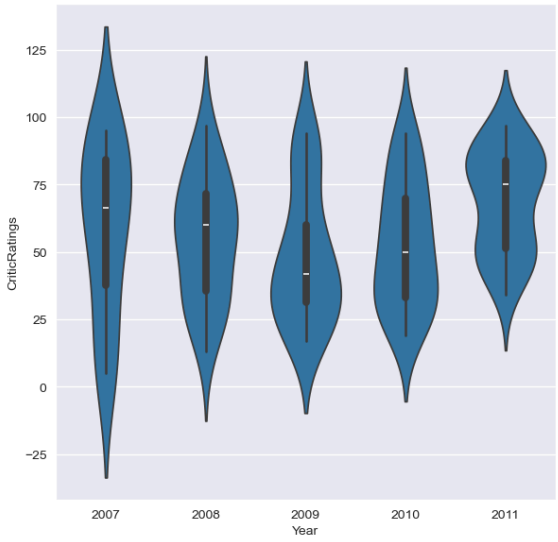
559 rows × 6 columns

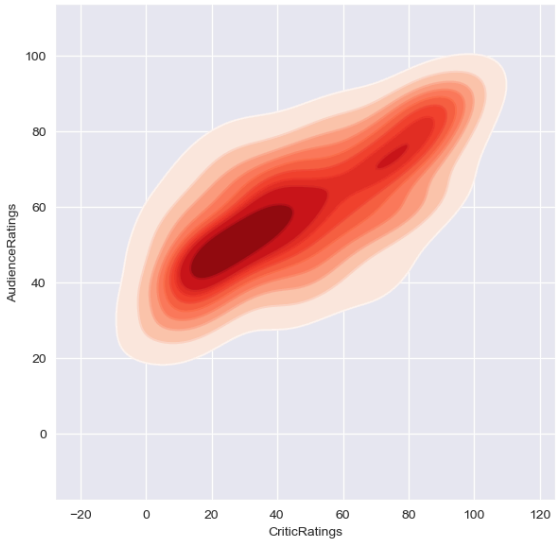
In [102…

```
# 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(data=movies,x='BudgetMillions',y='AudienceRatings',ax=axes[0,0]
k2 = sns.kdeplot(data=movies,x='BudgetMillions',y='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 = 'CriticRati
k4 = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',shade = True,
k4b = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',cmap='Reds',
plt.show()
```





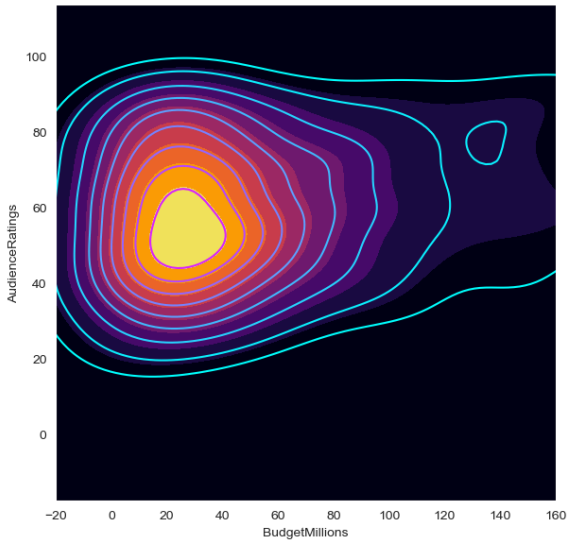


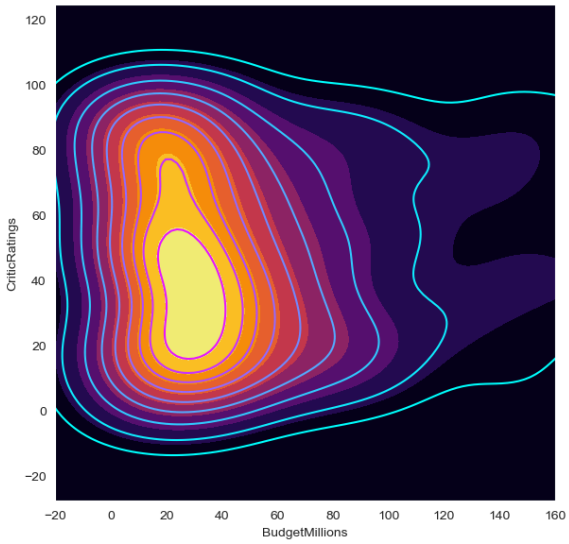


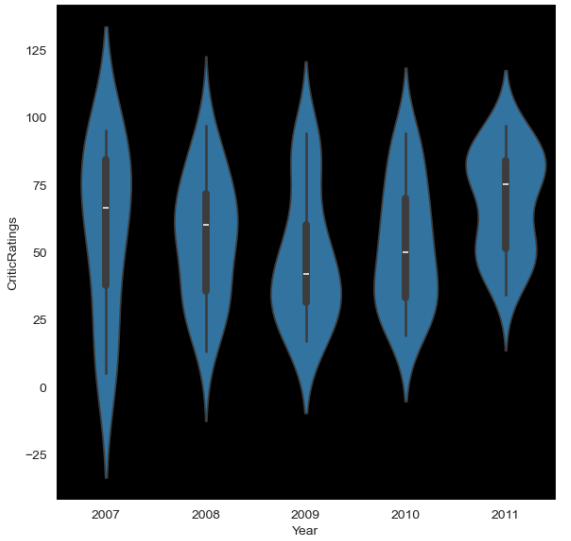
In [105…

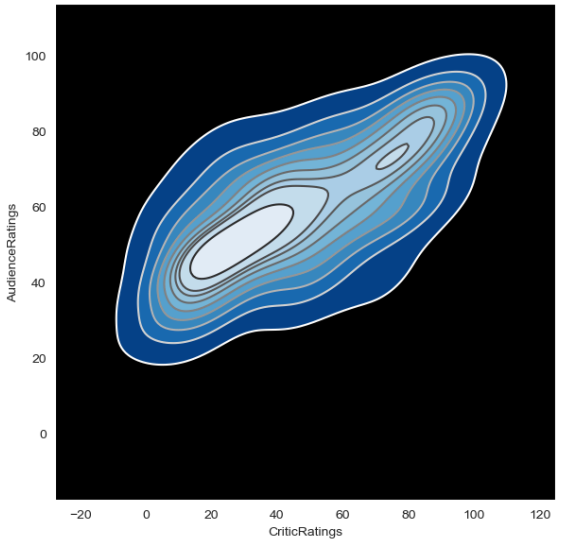
```
# How can you style your dashboard  using different color map
# python is not vectorize programming language
# Building dashboards (dashboard - combination of chats)

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='BudgetMillions',y='AudienceRatings',shade = True
k1b = sns.kdeplot(data=movies,x='BudgetMillions',y='AudienceRatings',cmap = 'coo

#plot [0,1]
k2 = sns.kdeplot(data=movies,x='BudgetMillions',y='CriticRatings',shade=True, sh
k2b = sns.kdeplot(data=movies,x='BudgetMillions',y='CriticRatings',cmap = 'cool'

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

#plot[1,1]
k4 = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',shade = True,
k4b = sns.kdeplot(data=movies,x='CriticRatings',y='AudienceRatings',cmap='gist_g

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

```
plt.show()
```









In [ ]:

In [ ]:

In [ ]: