MOVIE RATING ANALYTICS (ADVANCED VISULIZATION)

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

In [2]:
```
os.getcwd() ## if you want to change the working directory
```

Out[2]:
```
'C:\\Users\\Achal Raghorte'
```

In [3]:
```
movies=pd.read_csv(r"D:\Data Science with AI\29th-jan-2024\MOVIE RATINGS _ ADVANCE VISUALIZATION _ EDA 1\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.head()
```

Out[6]:

	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 [7]:
```
movies.tail()
```

Out[7]:

	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 [8]:
```
movies.columns
```

Out[8]:
```
Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
       'Budget (million $)', 'Year of release'],
      dtype='object')
```

In [9]:
```
movies.columns=['Film','Genre','Critic Rating','Audience Rating','BudgetMillions','Year']
```

In [10]:
```
movies.head()
```

Out[10]:

	Film	Genre	Critic Rating	Audience Rating	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 [11]:
```
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   Critic Rating    559 non-null    int64 
 3   Audience Rating  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 [12]:
```
movies.describe()
# 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[12]:

	Critic Rating	Audience Rating	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 [13]:
```
movies['Film']
#Movies Audience Rating %
```

Out[13]:
```
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 [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=movies.Film.astype('category')
```

In [16]:
```
movies.Film
```

Out[16]:
```
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 [17]:
```
movies.head()
```

Out[17]:

	Film	Genre	Critic Rating	Audience Rating	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 [18]:
```
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   Critic Rating    559 non-null    int64
 3   Audience Rating  559 non-null    int64
 4   BudgetMillions   559 non-null    int64
 5   Year             559 non-null    int64
dtypes: category(1), int64(4), object(1)
memory usage: 43.6+ KB
```

In [19]:
```
movies.Genre=movies.Genre.astype('category')
movies.Year=movies.Year.astype('category')
```

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

Out[20]:
```
0         Comedy
1      Adventure
2         Action
3      Adventure
4         Comedy
          ...
554       Comedy
555       Comedy
556     Thriller
557       Action
558       Comedy
Name: Genre, Length: 559, dtype: category
Categories (7, object): ['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'Romance', 'Thriller']
```

In [21]:
```
movies.Year
```

Out[21]:
```
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 [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   Critic Rating    559 non-null    int64   
 3   Audience Rating  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]:

	Critic Rating	Audience Rating	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

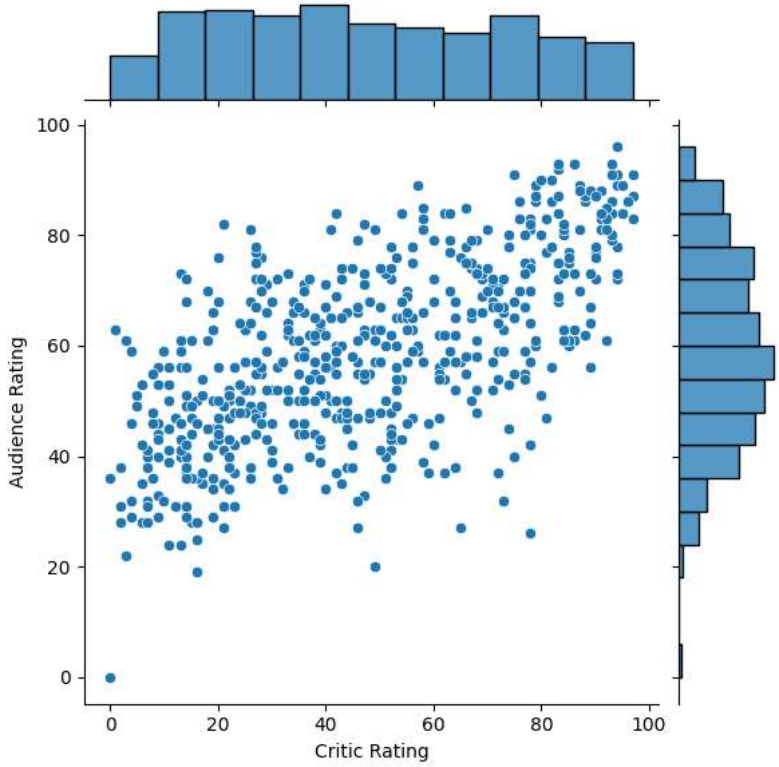
working with joint plots

In [25]:
```
from matplotlib import pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

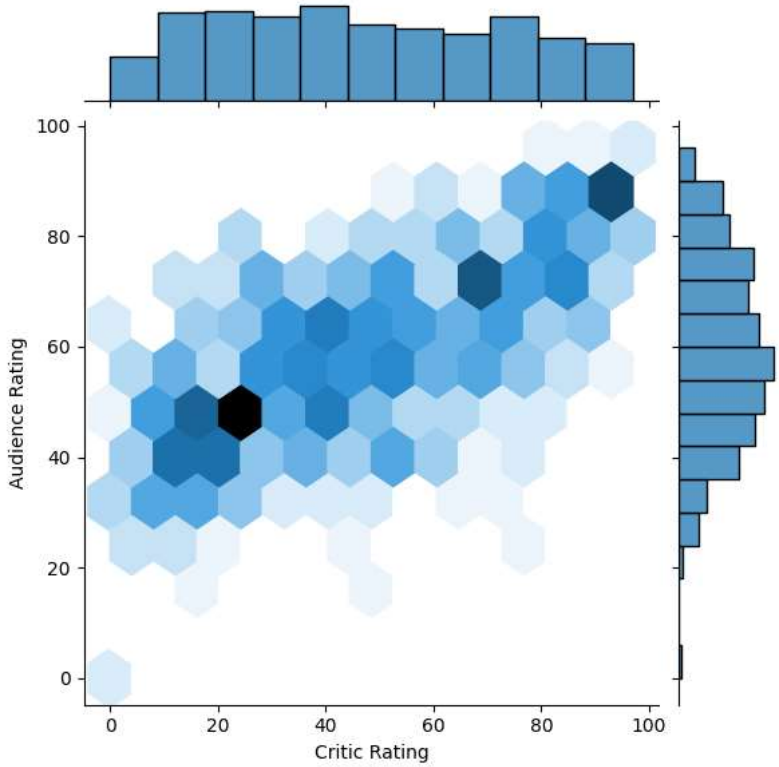
basically joint plot is a scatter plot & it find the relation b/w audiene & critics

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

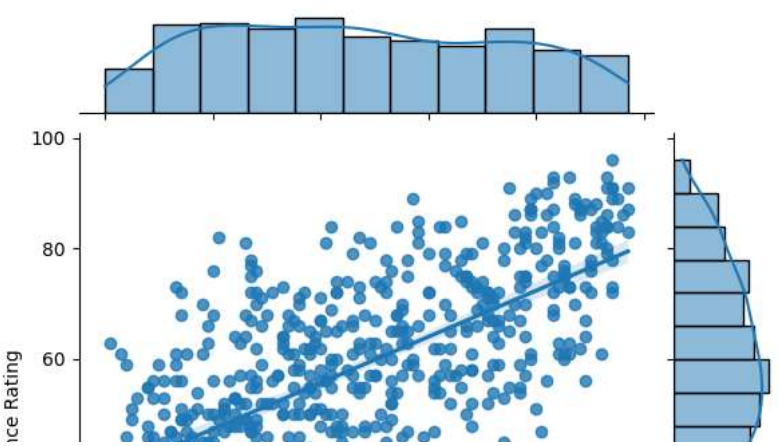
In [26]:
```
intplot( data=movies , x='Critic Rating' , y='Audience Rating')
e rating is more dominant then critics rating
n this we find out as most people are most liklihood to watch audience rating & less likely to wathc critics rating
explain the excel - if you filter audience rating & critic rating. critic rating has very low values compare to audience rating
```



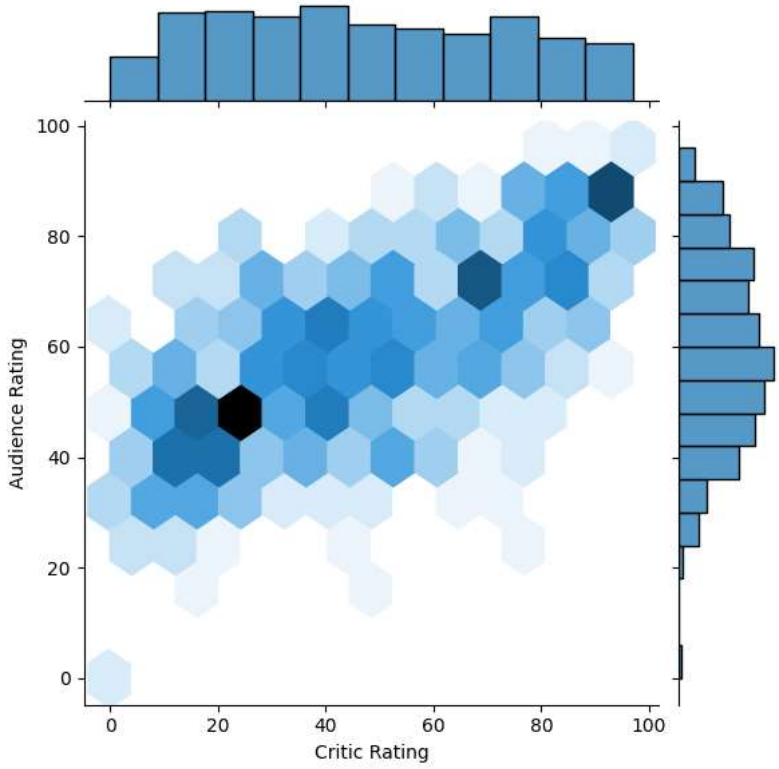
In [27]:
```
j=sns.jointplot( data=movies,x='Critic Rating',y='Audience Rating' , kind='hex')
```



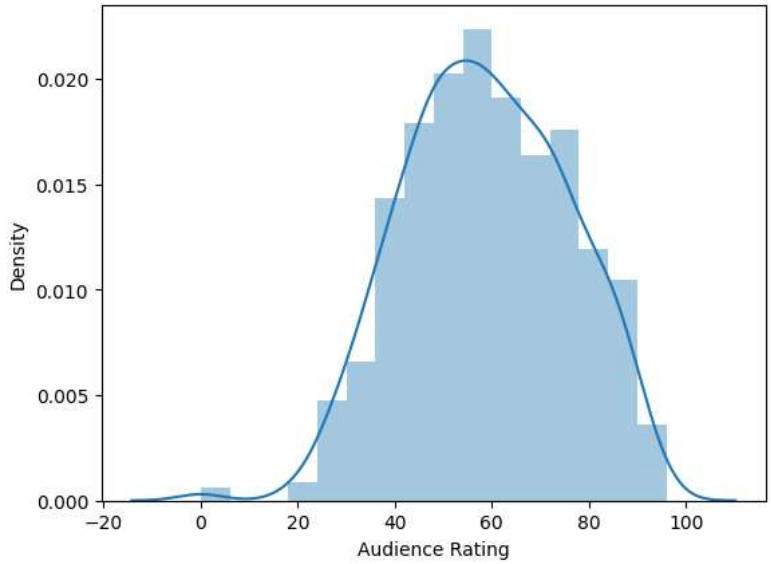
In [28]:
```
j=sns.jointplot( data=movies , x='Critic Rating', y='Audience Rating', kind='reg')
```



In [29]:
```
j=sns.jointplot( data=movies,x='Critic Rating',y='Audience Rating' , kind='hex')
```

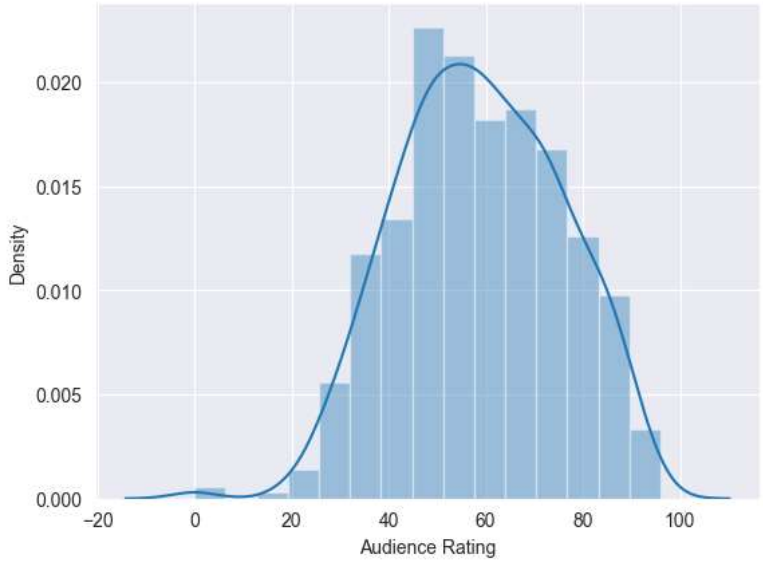


In [30]:
```
#Histograms

# <<< chat1

m1 = sns.distplot(movies["Audience Rating"])

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

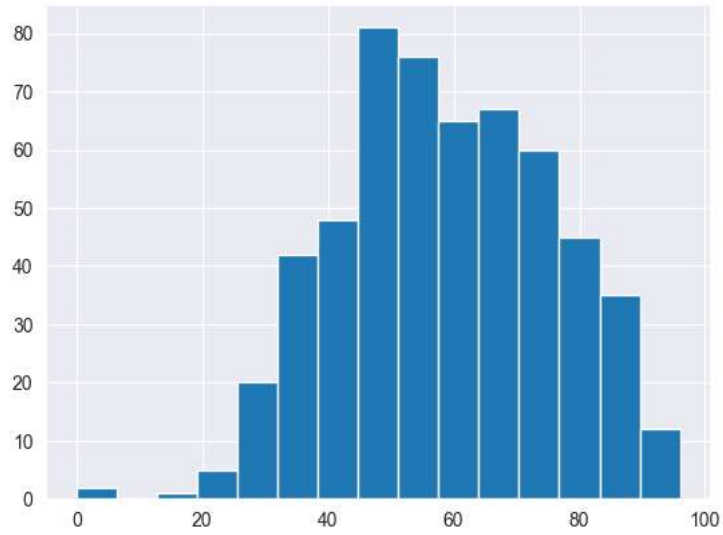


In [31]:
```
sns.set_style('darkgrid')
```

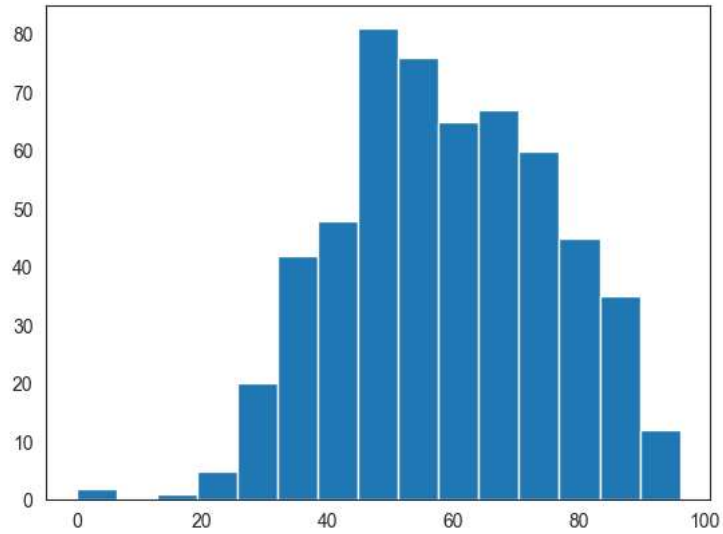
In [32]:
```
m2=sns.distplot(movies["Audience Rating"],bins=15)
```



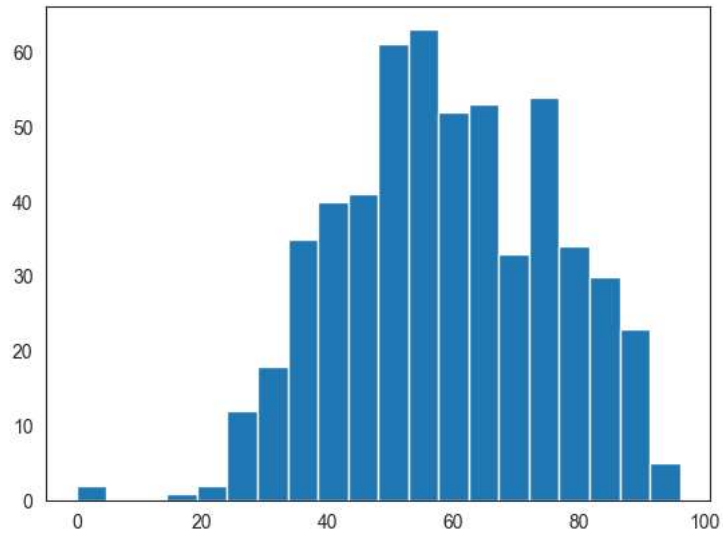
In [33]:
```
#sns.set_style('darkgrid')
n1=plt.hist(movies["Audience Rating"],bins=15)
```



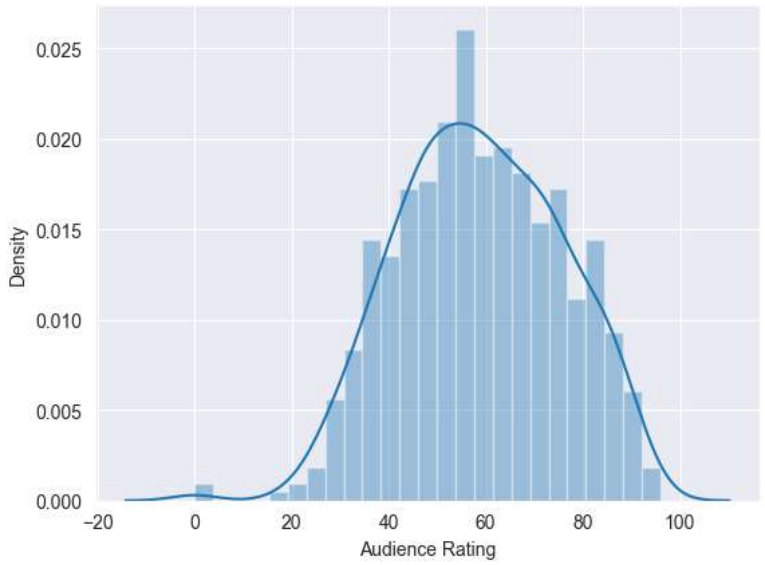
In [34]:
```
sns.set_style('white')
n2=plt.hist(movies["Audience Rating"],bins=15)
```



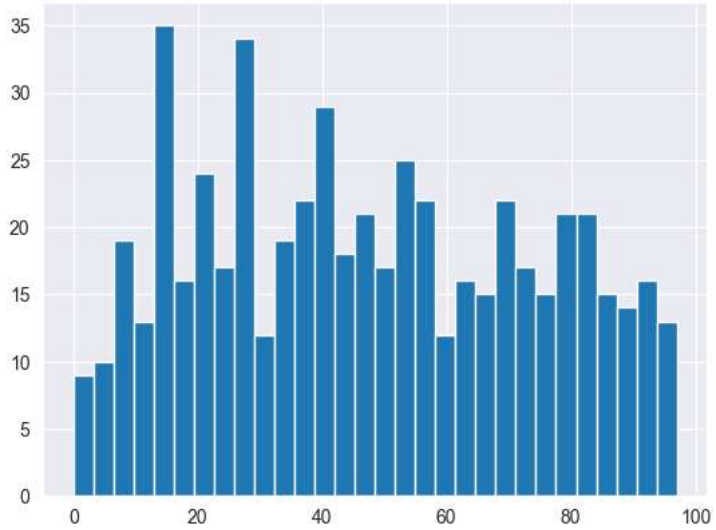
In [35]:
```
sns.set_style('white')
n1=plt.hist(movies["Audience Rating"],bins=20)
```



In [36]:
```
sns.set_style('darkgrid')
b2=sns.distplot(movies["Audience Rating"],bins=25)
```



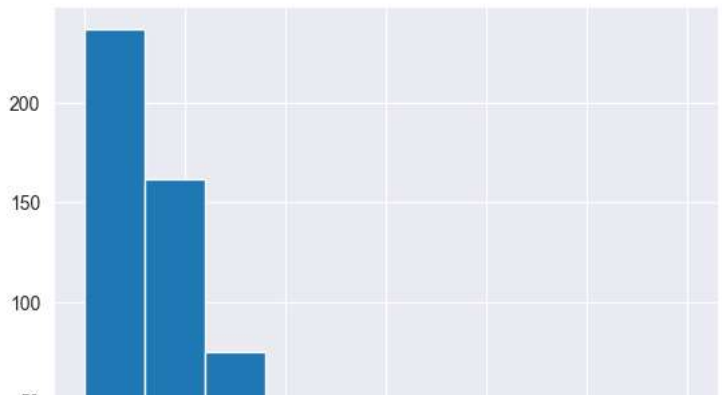
In [37]:
```
b2 = plt.hist(movies["Critic Rating"],bins=30) #uniform distribution
```



creating stacked histograms

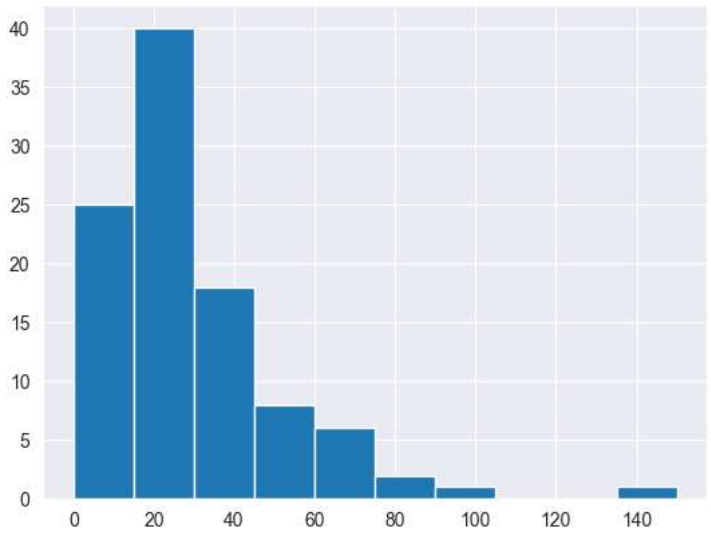
In [38]:
```
#h1=plt.hist(movies.BudgetMillions)
plt.hist(movies.BudgetMillions)
plt.show
```

Out[38]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



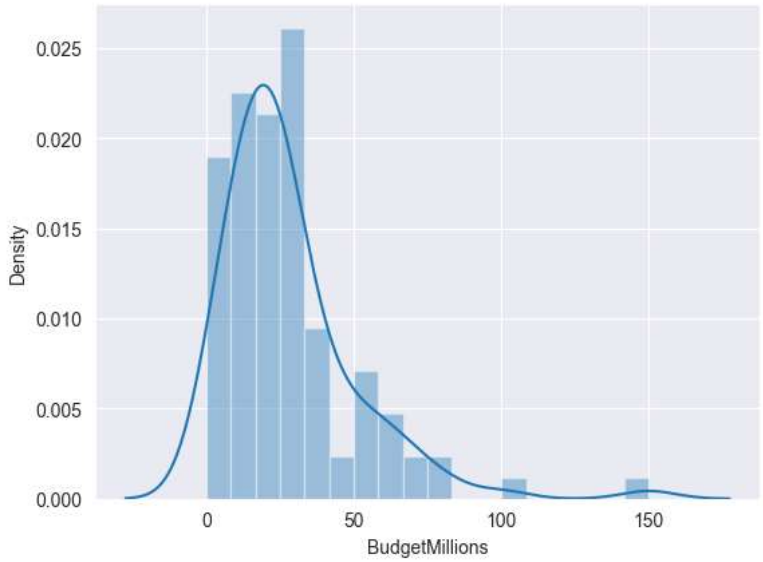
In [39]:
```
plt.hist(movies[movies.Genre=='Drama'].BudgetMillions)
plt.show
```

Out[39]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [40]:
```
sns.distplot(movies[movies.Genre=='Drama'].BudgetMillions)
plt.show
```

Out[40]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [41]:
```
movies.head()
```

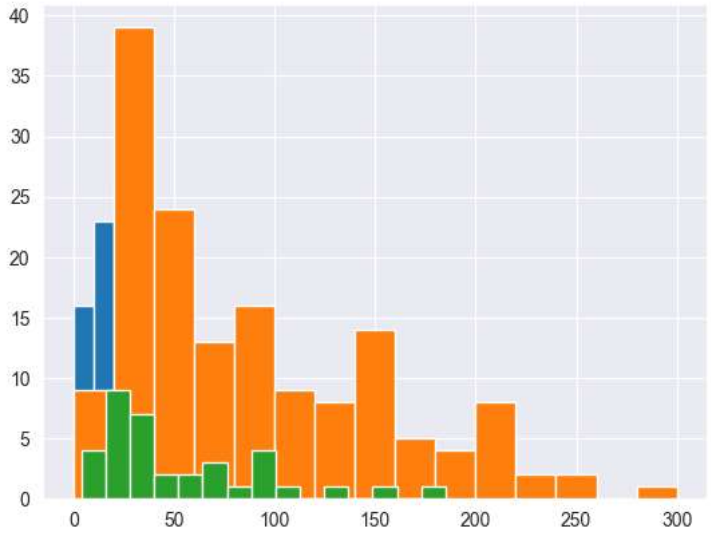
Out[41]:

	Film	Genre	Critic Rating	Audience Rating	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

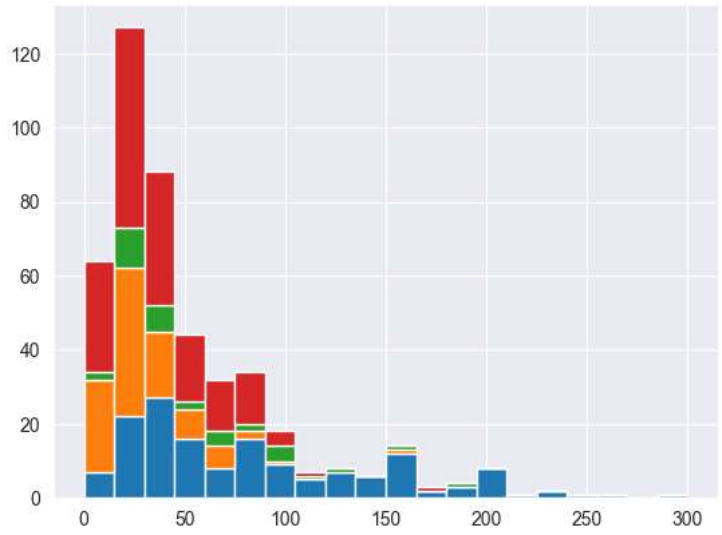
movies.Genre.unique()

In [42]:
```
# Below plots are stacked histogram becuase overlaped
plt.hist(movies[movies.Genre=='Drama'].BudgetMillions,bins=15)
plt.hist(movies[movies.Genre=='Action'].BudgetMillions,bins=15)
plt.hist(movies[movies.Genre=='Thriller'].BudgetMillions,bins=15)
plt.legend
plt.show
```

Out[42]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [43]:
```
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()
```

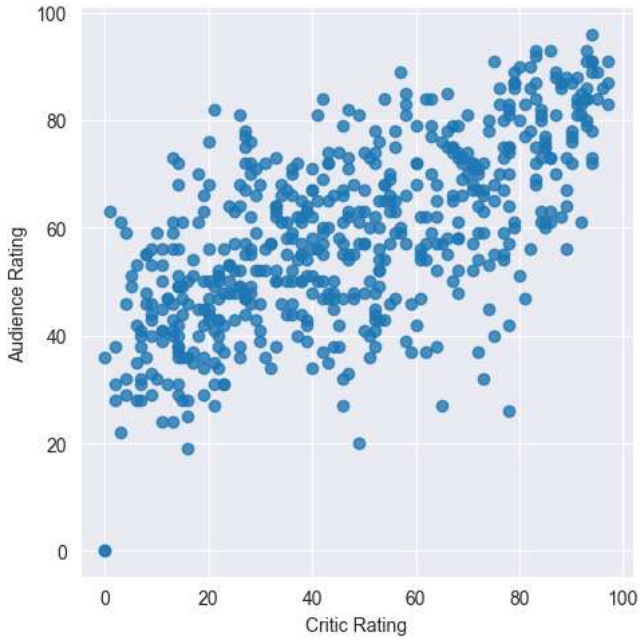


if you have 100 categories you cannot copy & paste all the things

In [44]:
```
for gen  in movies.Genre.cat.categories:
    print(gen)
```

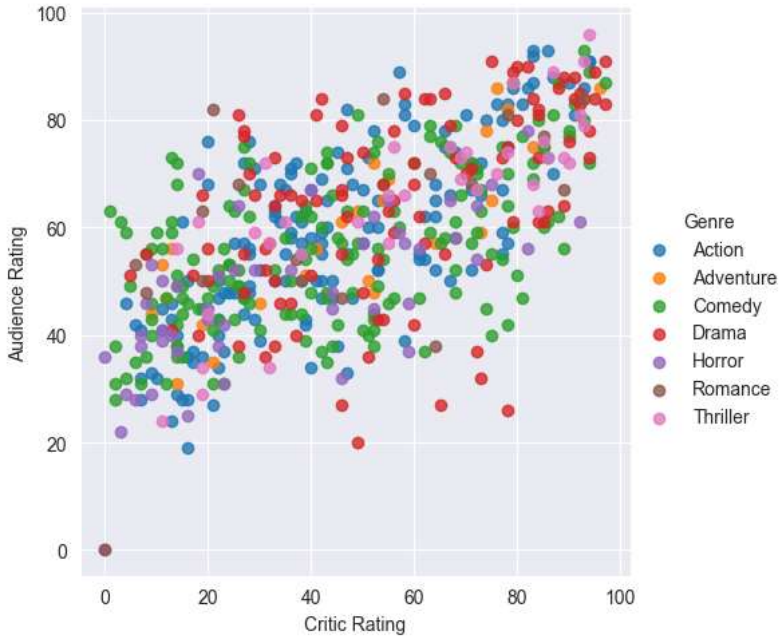
```
Action
Adventure
Comedy
Drama
Horror
Romance
Thriller
```

In [45]:
```
vis1 = sns.lmplot(data=movies, x='Critic Rating', y='Audience Rating',\
     fit_reg=False )
```

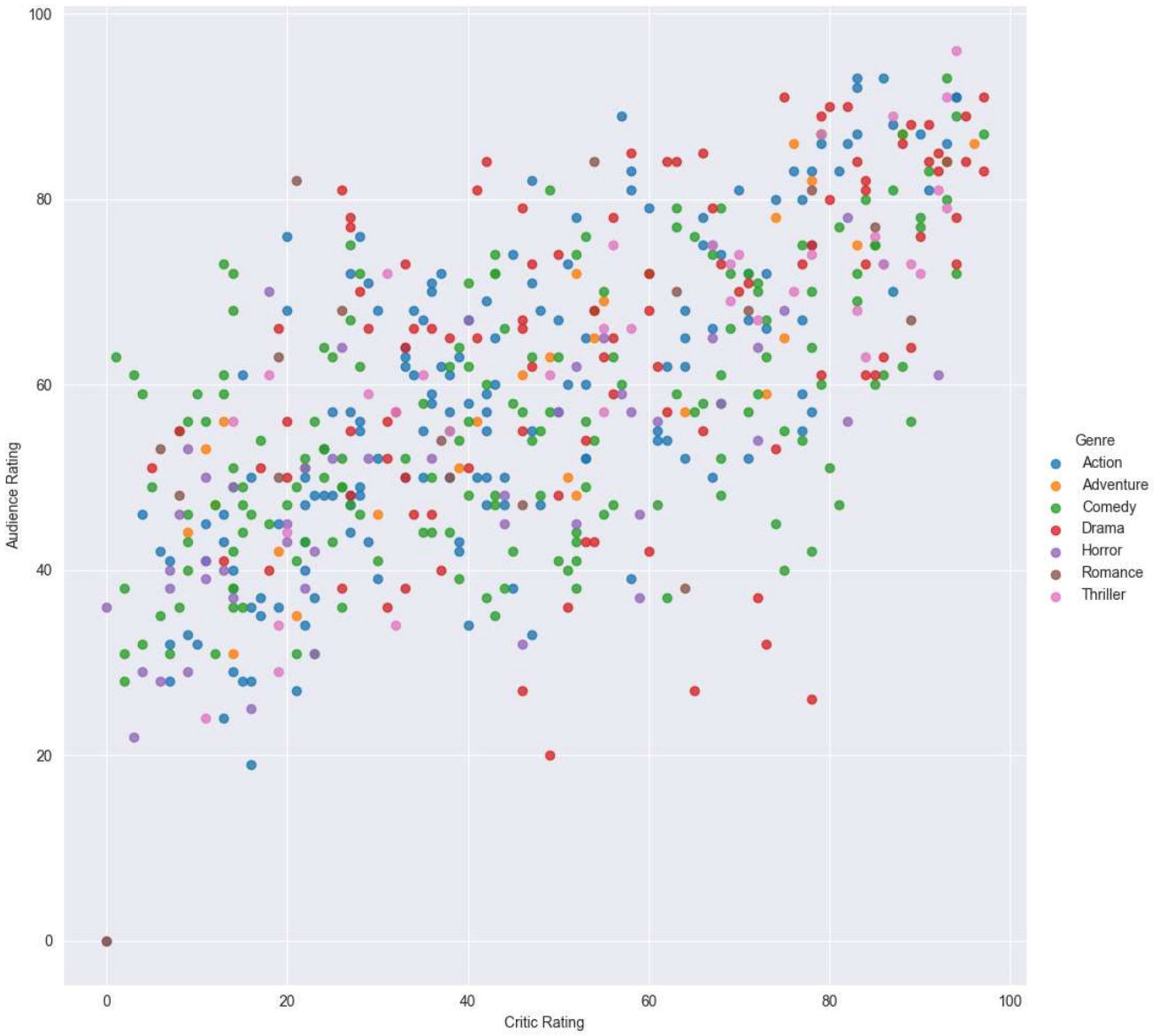


In [46]:
```
# Print the columns of the 'movies' DataFrame
print(movies.columns)

# Check if the columns are present
if 'Audience Rating' in movies.columns and 'Critic Rating' in movies.columns:
    # Your seaborn code here
    vis1 = sns.lmplot(data=movies, x='Critic Rating', y='Audience Rating', fit_reg=False, hue='Genre')
else:
    print("Columns not found in the DataFrame.")
```

```
Index(['Film', 'Genre', 'Critic Rating', 'Audience Rating', 'BudgetMillions',
       'Year'],
      dtype='object')
```



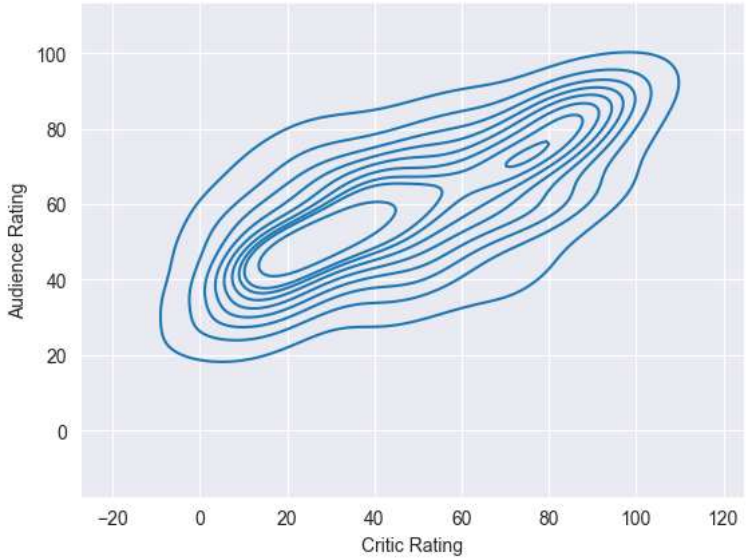
In [47]:
```
vis1= sns.lmplot(data=movies , x='Critic Rating' , y='Audience Rating' ,\
                 fit_reg=False, hue='Genre' , height=10,aspect=1)
```



In [48]:
```
# Kernal Density Estimate plot ( KDE PLOT)
# how can i visulize audience rating & critics rating . using scatterplot
```

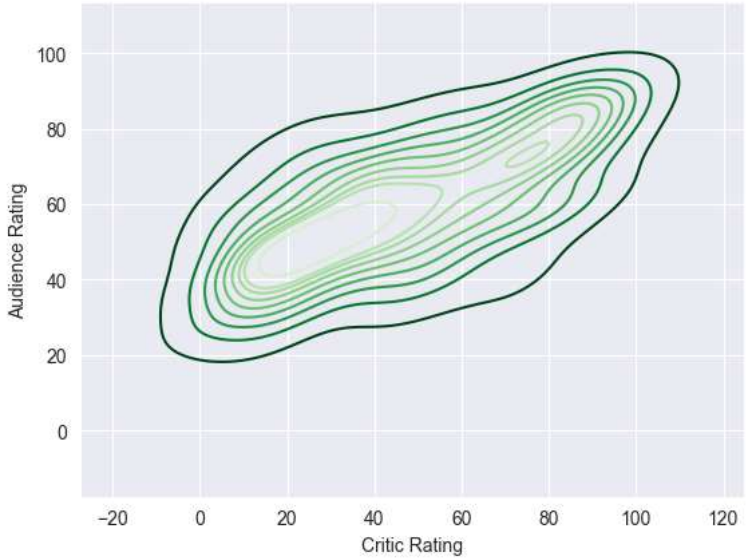
In [49]:
```
#pip install seaborn matplotlib
```

In [50]:
```
#Assuming 'movies' is your DataFrame
k1 = sns.kdeplot(data=movies, x='Critic Rating', y='Audience Rating')
plt.show()
```

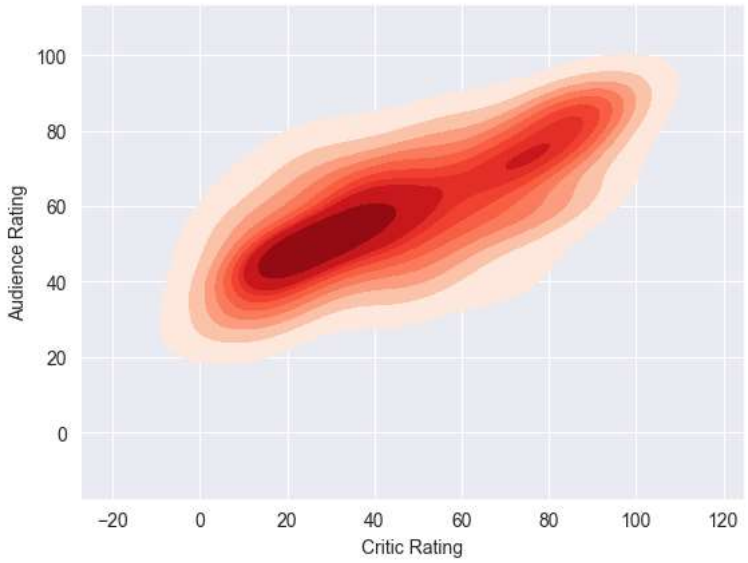


In [51]:
```
#pip install --upgrade seaborn matplotlib
```

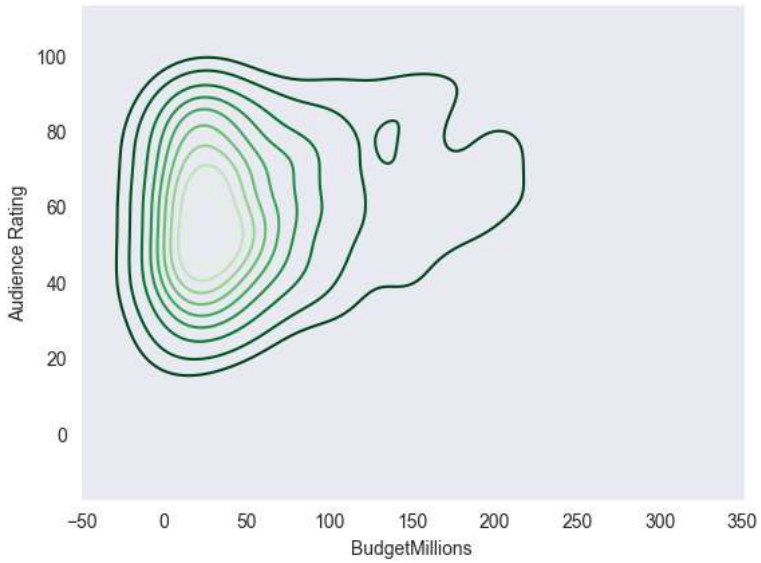
In [52]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'movies' is your DataFrame
k2 = sns.kdeplot(x=movies['Critic Rating'], y=movies['Audience Rating'], shade_lowest=False, cmap='Greens_r')
plt.show()
```



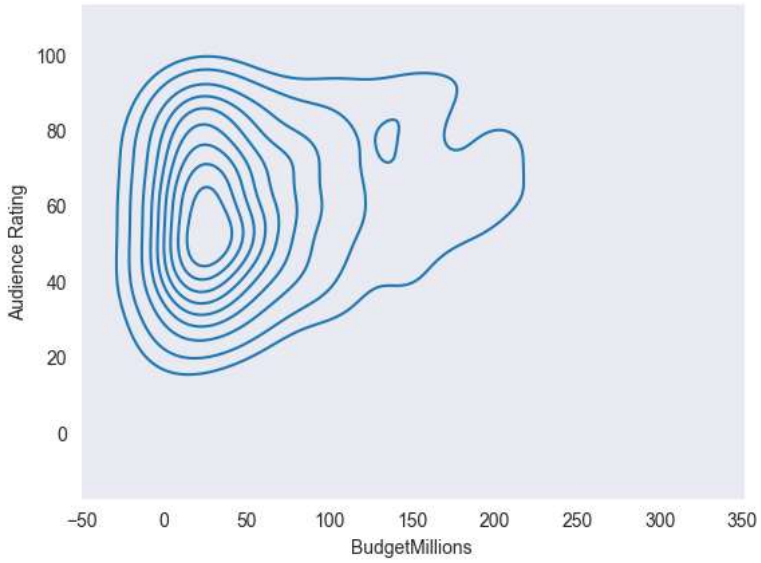
In [53]:
```
k1=sns.kdeplot(x=movies['Critic Rating'] , y=movies['Audience Rating'] ,shade=True,shade_lowest=False,cmap='Reds')
```



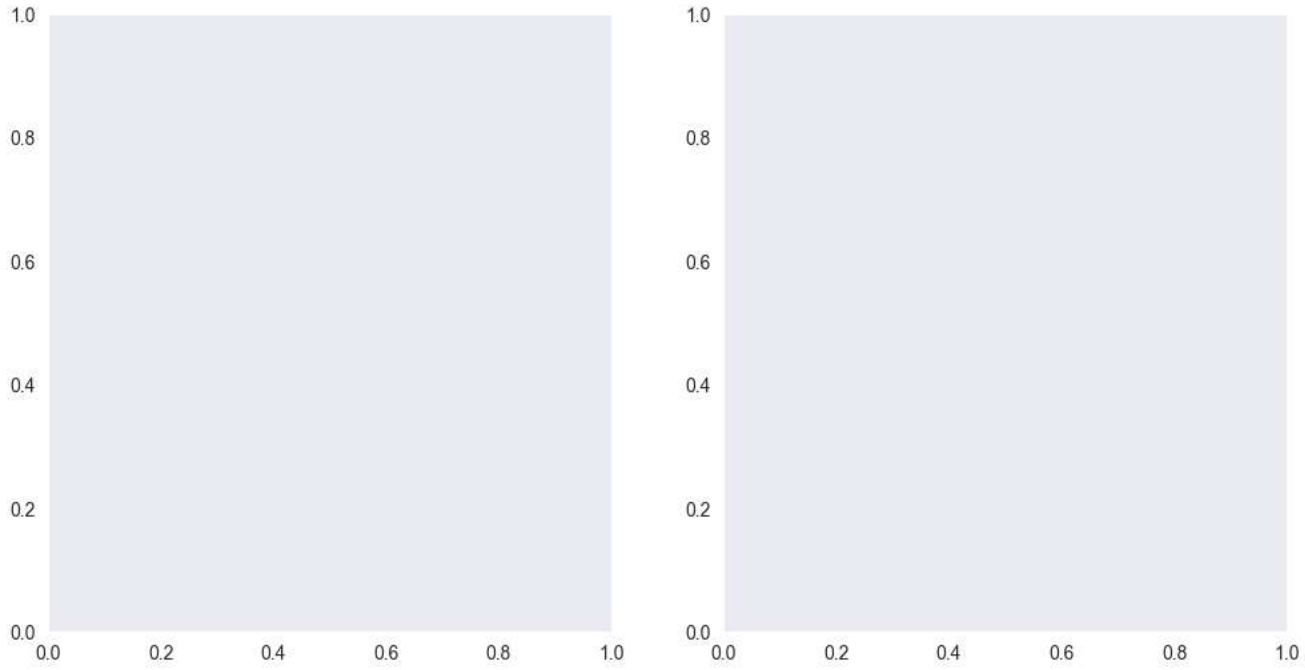
In [54]:
```
sns.set_style('dark')
k1= sns.kdeplot(x=movies['BudgetMillions'], y=movies['Audience Rating'] ,shade_lowest=False ,cmap='Greens_r')
```



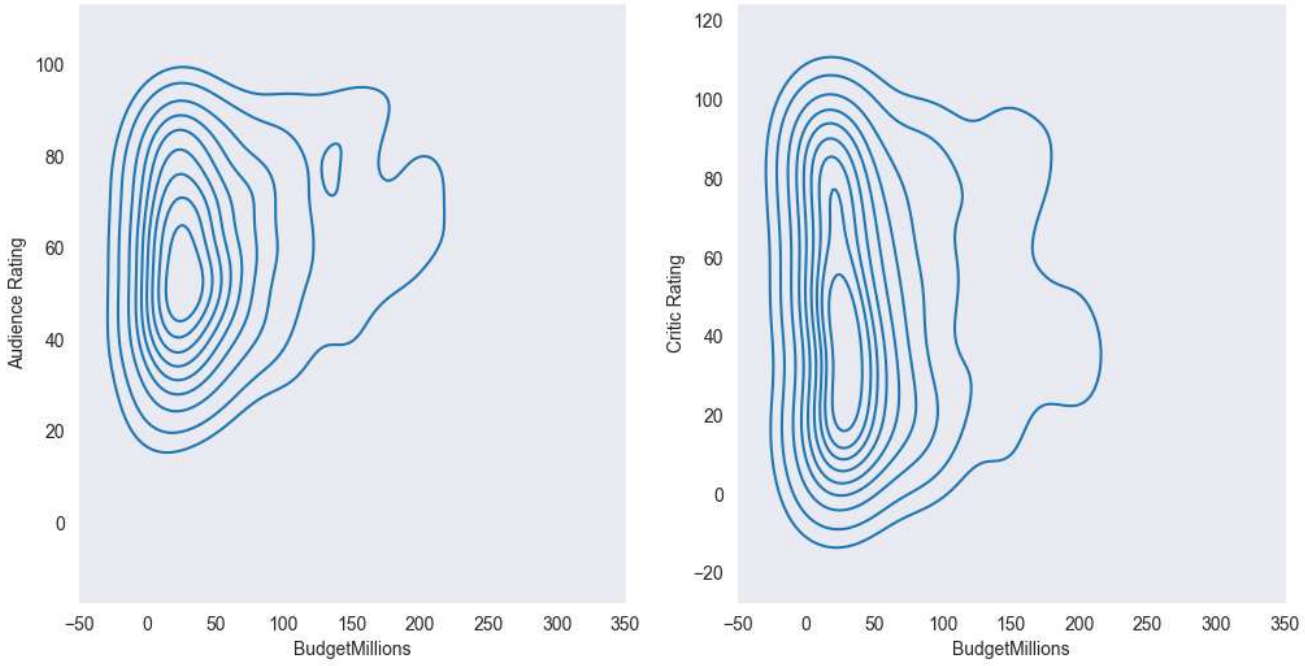
In [55]:
```
sns.set_style('dark')
k1=sns.kdeplot(x=movies['BudgetMillions'], y=movies['Audience Rating'])
```



In [56]:
```
#subplots

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



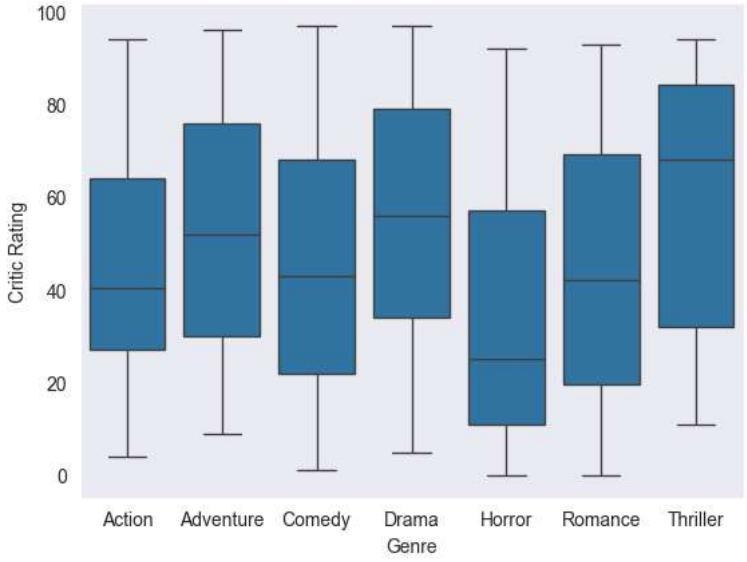
In [57]:
```
f , axes=plt.subplots(1,2 ,figsize=(12,6))

k1=sns.kdeplot(x=movies['BudgetMillions'],y=movies['Audience Rating'],ax=axes[0])
k2=sns.kdeplot(x=movies['BudgetMillions'],y=movies['Critic Rating'], ax=axes[1])
```



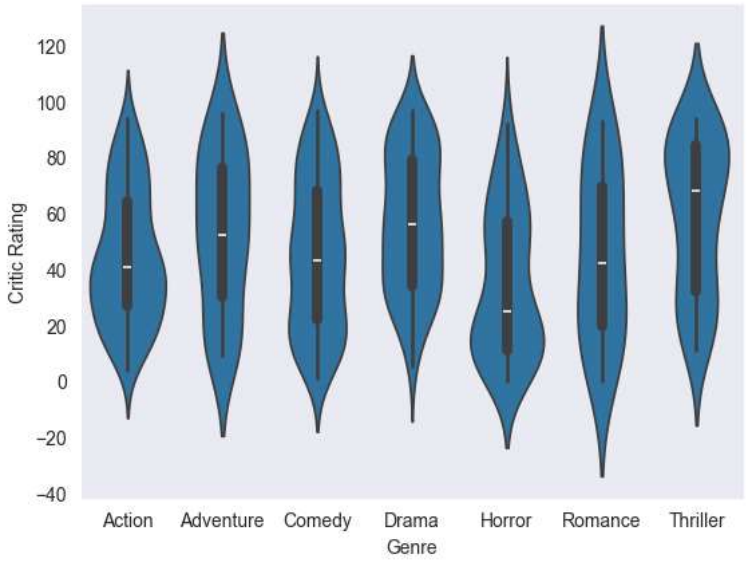
In [58]:
```
axes
```

Out[58]:
```
array([<Axes: xlabel='BudgetMillions', ylabel='Audience Rating'>,
       <Axes: xlabel='BudgetMillions', ylabel='Critic Rating'>],
      dtype=object)
```

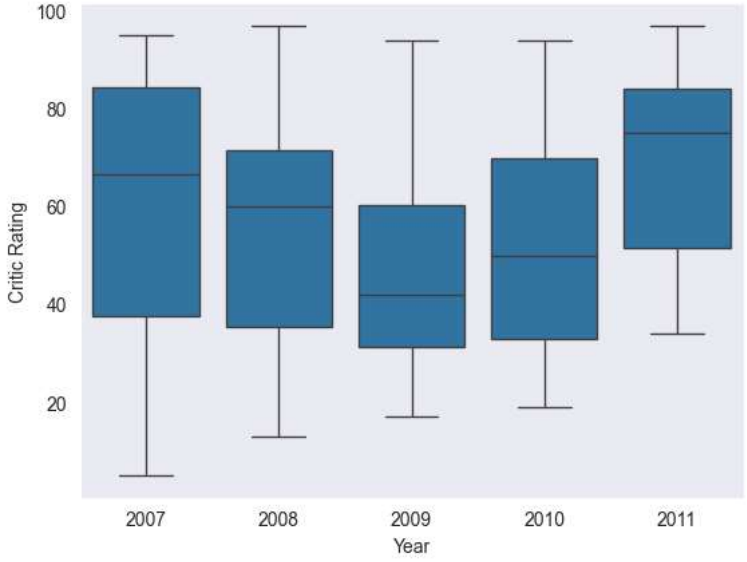
In [59]:
```
#Box plots

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



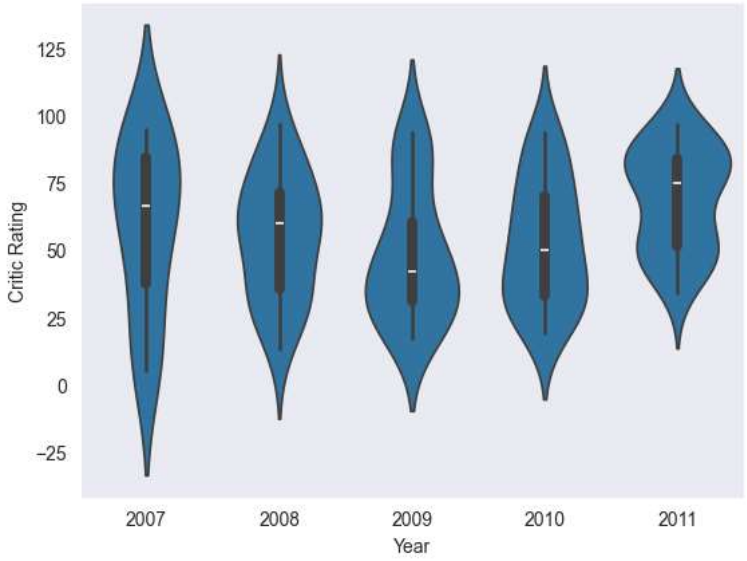
In [60]:
```
#Violin plot

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



In [61]:
```
w1=sns.boxplot(data=movies[movies.Genre=='Drama'] ,x='Year' ,y='Critic Rating')
```

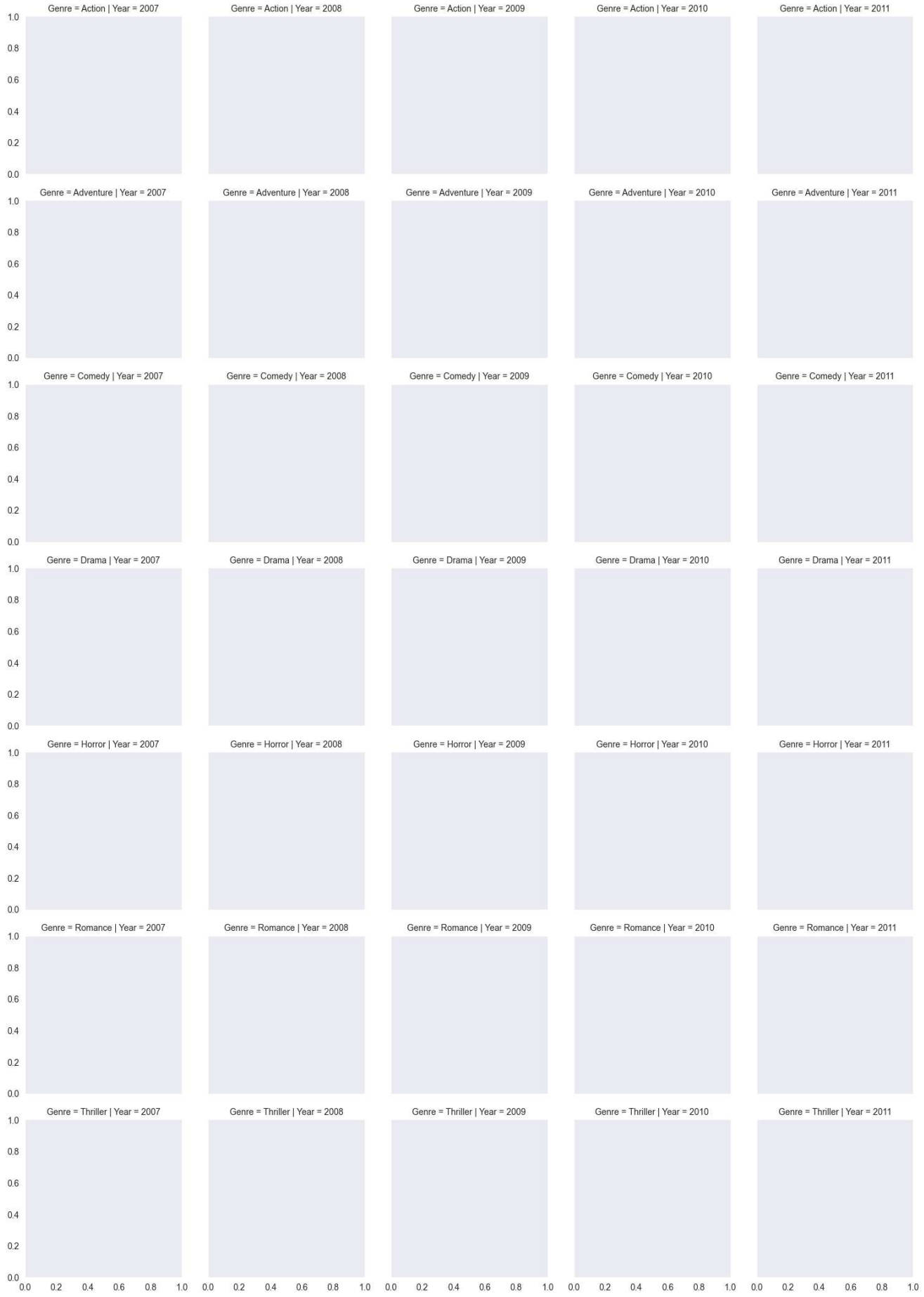


In [62]:
```
z= sns.violinplot(data=movies[movies.Genre=='Drama'],x='Year' ,y='Critic Rating')
```



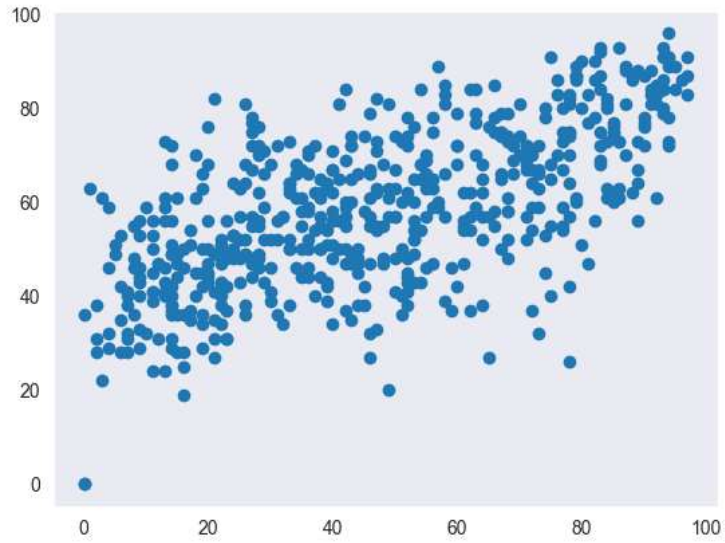
In [63]:
```
# Creating a facet grid
```

In [64]:
```
g= sns.FacetGrid(movies ,row='Genre' ,col='Year',hue='Genre')#kind of subplots
```

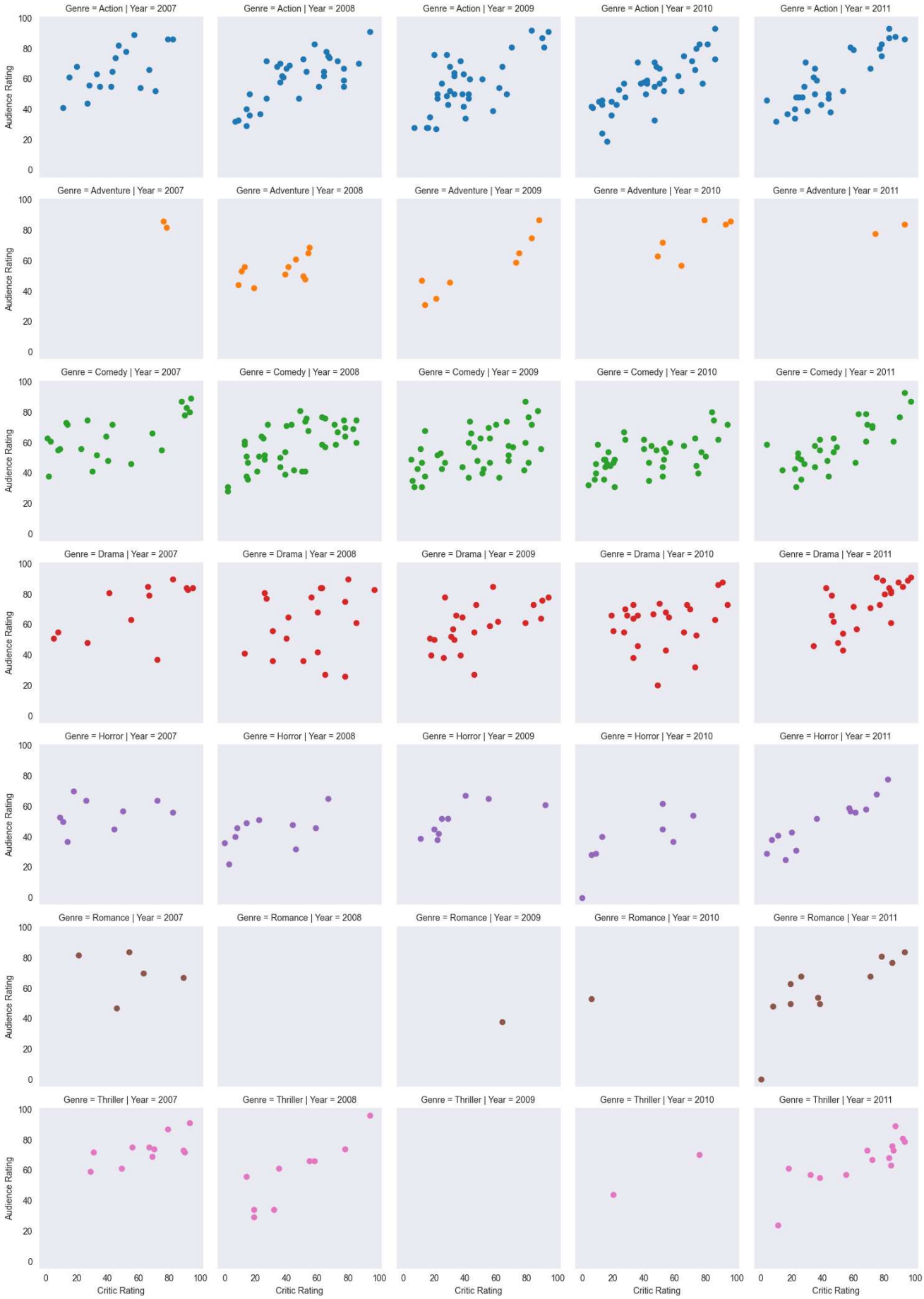


In [66]:
```
plt.scatter(x=movies['Critic Rating'],y=movies['Audience Rating'])
```

Out[66]:
```
<matplotlib.collections.PathCollection at 0x1cb7bd5a450>
```



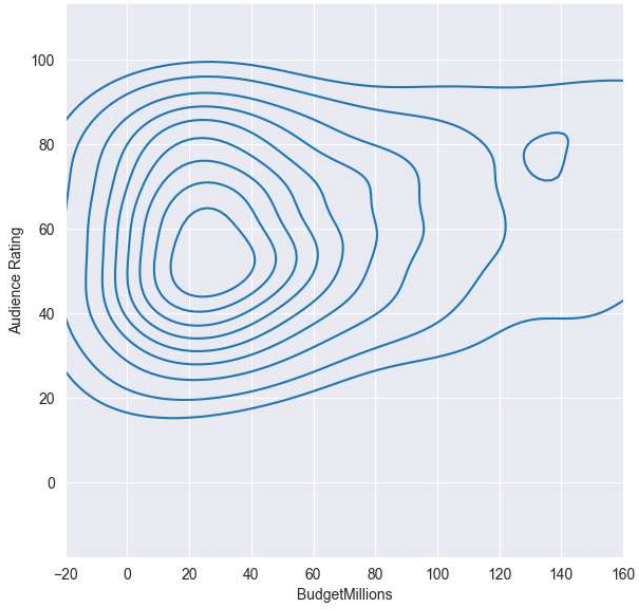
In [70]:
```
g= sns.FacetGrid(movies ,row='Genre' ,col='Year' ,hue='Genre')
g=g.map(plt.scatter ,'Critic Rating' ,'Audience Rating') #scatterplots are mapped in facetgrid
```

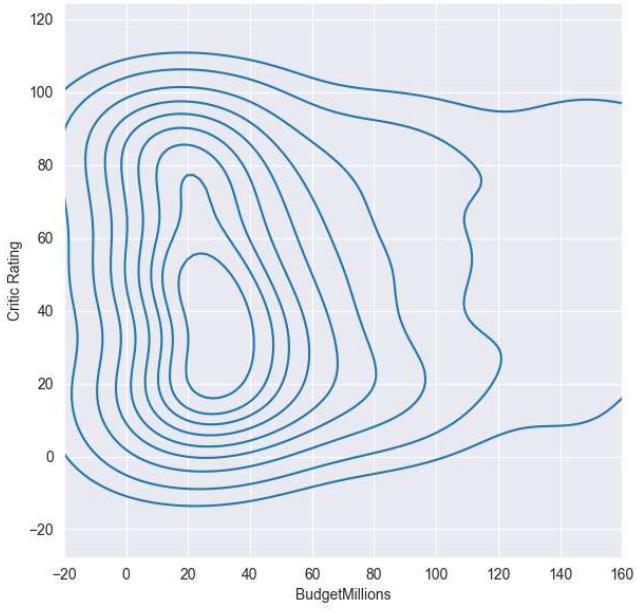

Genre = Action | Year = 2007
Genre = Action | Year = 2008
Genre = Action | Year = 2009
Genre = Action | Year = 2010
Genre = Action | Year = 2011
Genre = Adventure | Year = 2007
Genre = Adventure | Year = 2008
Genre = Adventure | Year = 2009
Genre = Adventure | Year = 2010
Genre = Adventure | Year = 2011
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
Genre = Horror | Year = 2007
Genre = Horror | Year = 2008
Genre = Horror | Year = 2009
Genre = Horror | Year = 2010
Genre = Horror | Year = 2011
Genre = Romance | Year = 2007
Genre = Romance | Year = 2008
Genre = Romance | Year = 2009
Genre = Romance | Year = 2010
Genre = Romance | Year = 2011
Genre = Thriller | Year = 2007
Genre = Thriller | Year = 2008
Genre = Thriller | Year = 2009
Genre = Thriller | Year = 2010
Genre = Thriller | Year = 2011
Audience Rating
Critic Rating
0
20
40
60
80
100

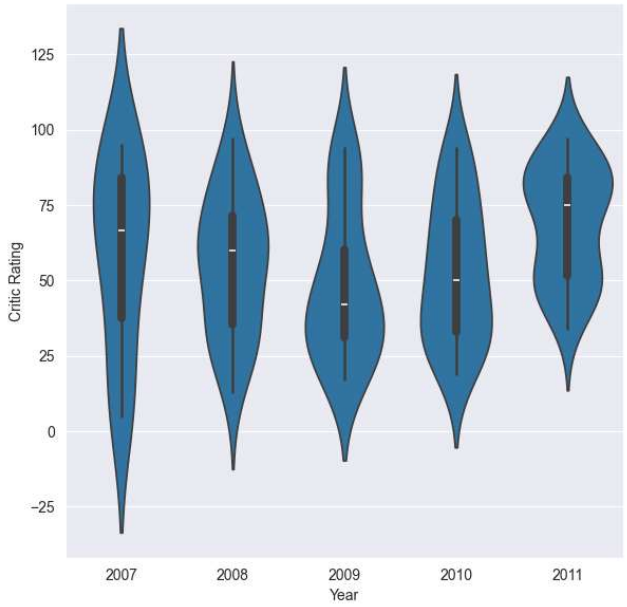
In [71]:
```
# you can populated any type of chat

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

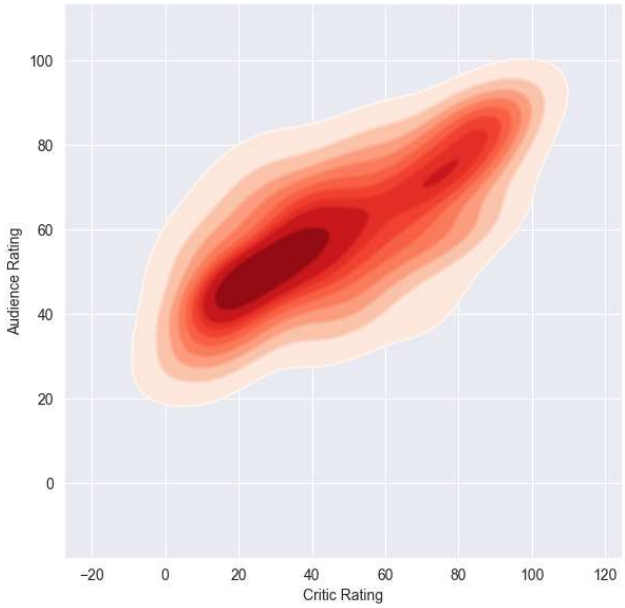

Genre = Action | Year = 2007
Genre = Action | Year = 2008
Genre = Action | Year = 2009
Genre = Action | Year = 2010
Genre = Action | Year = 2011
Genre = Adventure | Year = 2007
Genre = Adventure | Year = 2008
Genre = Adventure | Year = 2009
Genre = Adventure | Year = 2010
Genre = Adventure | Year = 2011
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
Genre = Horror | Year = 2007
Genre = Horror | Year = 2008
Genre = Horror | Year = 2009
Genre = Horror | Year = 2010
Genre = Horror | Year = 2011
Genre = Romance | Year = 2007
Genre = Romance | Year = 2008
Genre = Romance | Year = 2009
Genre = Romance | Year = 2010
Genre = Romance | Year = 2011
Genre = Thriller | Year = 2007
Genre = Thriller | Year = 2008
Genre = Thriller | Year = 2009
Genre = Thriller | Year = 2010
Genre = Thriller | Year = 2011
0 2 4 6 8 10 12 14 16
0 100 200 300
BudgetMillions

In [73]:
```
## 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['BudgetMillions'],y=movies['Audience Rating'],ax=axes[0,0])
k2 = sns.kdeplot(x=movies['BudgetMillions'],y=movies['Critic Rating'],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 = 'Critic Rating', ax=axes[1,0])

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

k4b = sns.kdeplot(x=movies['Critic Rating'], y=movies['Audience Rating'],cmap='Reds',ax = axes[1,1])

plt.show()
```

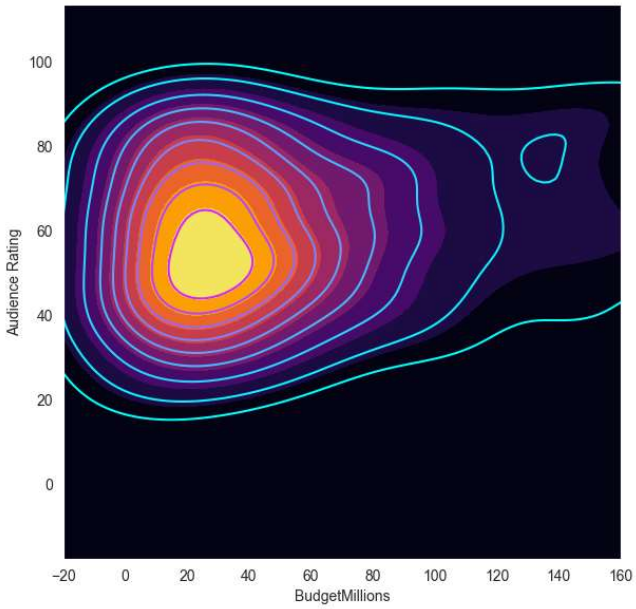


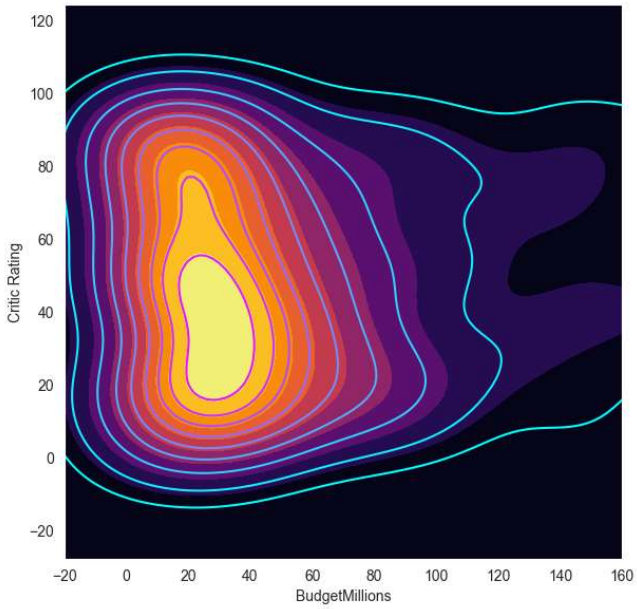


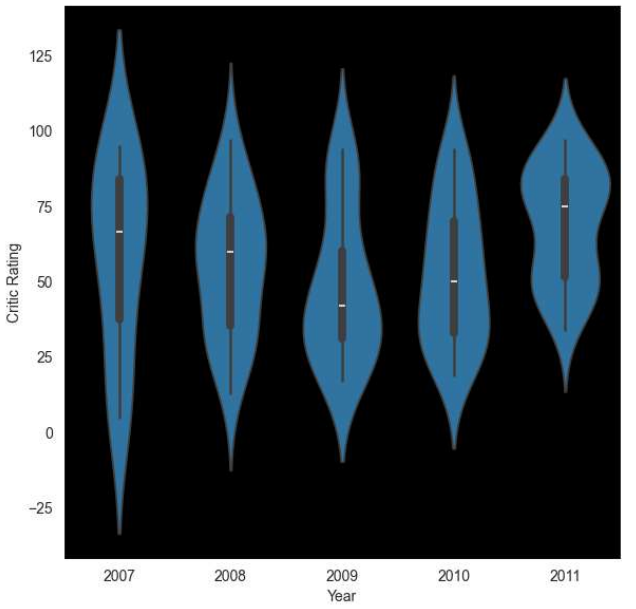


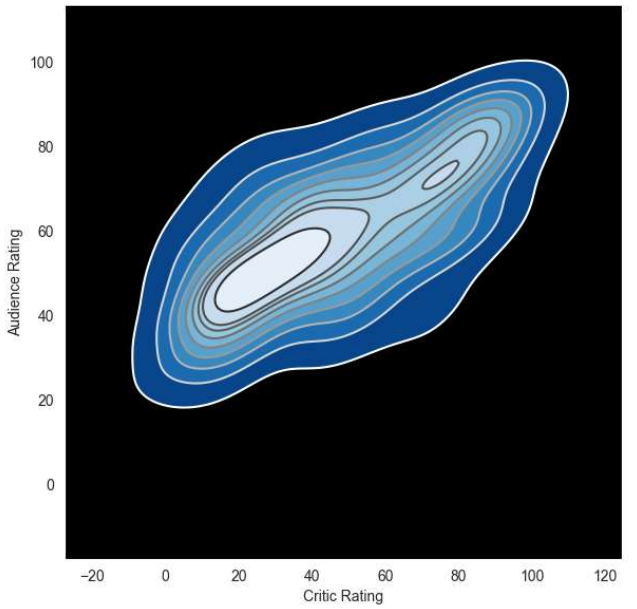


In [75]:
```
# How can you style your dashboard  using different color map

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

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

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

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

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

#plot[1,1]
k4 = sns.kdeplot(x=movies['Critic Rating'],y=movies['Audience Rating'], \
                 shade = True,shade_lowest=False,cmap='Blues_r', \
                 ax=axes[1,1])

k4b = sns.kdeplot(x=movies['Critic Rating'], y=movies['Audience Rating'], \
                  cmap='gist_gray_r',ax = axes[1,1])


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

plt.show()
```









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

In [ ]: