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

In [2]:
```
import os
```

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

Out[3]:
```
'F:\\Naresh IT\\NOV\\25th Adv Seaborn'
```

In [4]:
```
movies = pd.read_csv(r'material/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]:
```
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]:
```
len(movies)
```

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

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

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

In [9]:
```
# removing '%' and space from column names(to reduce noice)
movies.columns = ['Film', 'Genre', 'CriticRating', 'AudienceRating','BudgetMilli
```

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

Out[10]:

	Film	Genre	CriticRating	AudienceRating	BudgetMillions	Year
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 [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   CriticRating    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 [12]:
```
movies.describe()
```

Out[12]:

	CriticRating	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

The data type of year is int therefore the mean is 2009.15 which is meaningless. So we will change its data type to categroy

We also have to convert object type to category

In [13]:
```
movies['Film']
```

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.head()
```

Out[16]:
```
0    (500) Days of Summer
1              10,000 B.C.
2               12 Rounds
3                127 Hours
4                17 Again
Name: Film, 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	CriticRating	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

Now we convert Genre and Year

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

In [19]:
```
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   CriticRating    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 [20]:
```
movies.describe()
```

Out[20]:

	CriticRating	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

Now the data is clean and can be used for visualization

Data Visualization

In [21]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

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

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

Out[22]:
```
Index(['Film', 'Genre', 'CriticRating', 'AudienceRating', 'BudgetMillions',
       'Year'],
      dtype='object')
```

In [23]:
```
#movies.reset_index()
```

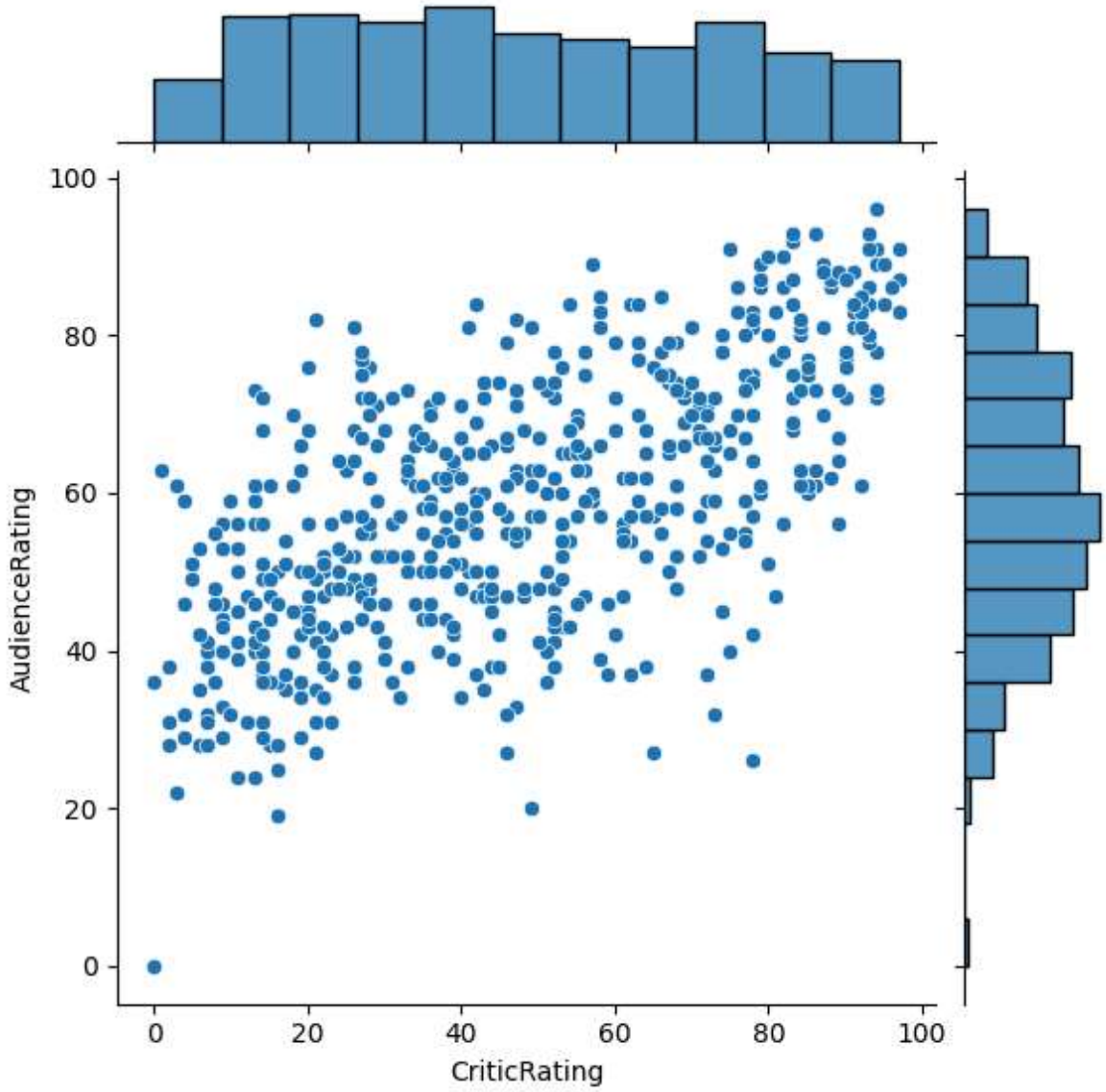
In [24]:
```
pd.__version__
```

Out[24]:
```
'2.3.3'
```

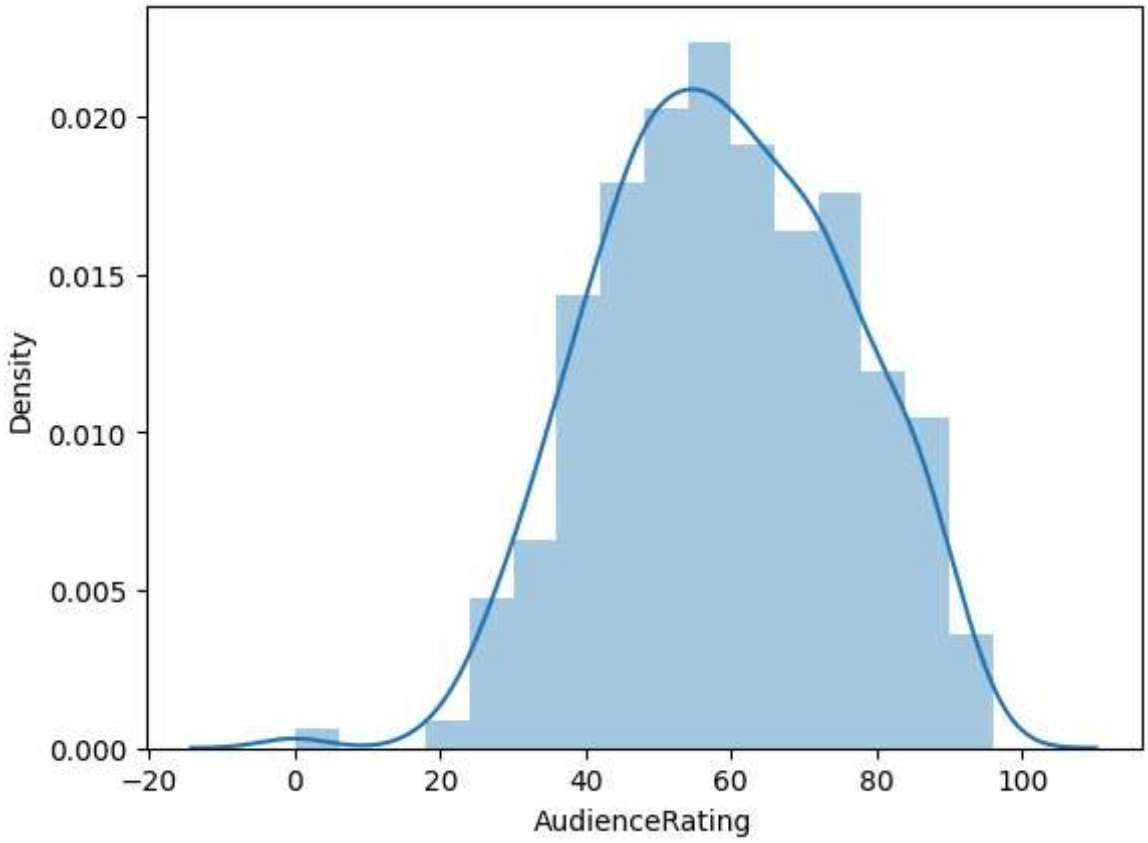
In [25]:
```
movies.columns
```

Out[25]:
```
Index(['Film', 'Genre', 'CriticRating', 'AudienceRating', 'BudgetMillions',
       'Year'],
      dtype='object')
```

In [26]:
```
j = sns.jointplot( data = movies, x = 'CriticRating', y = 'AudienceRating')
```

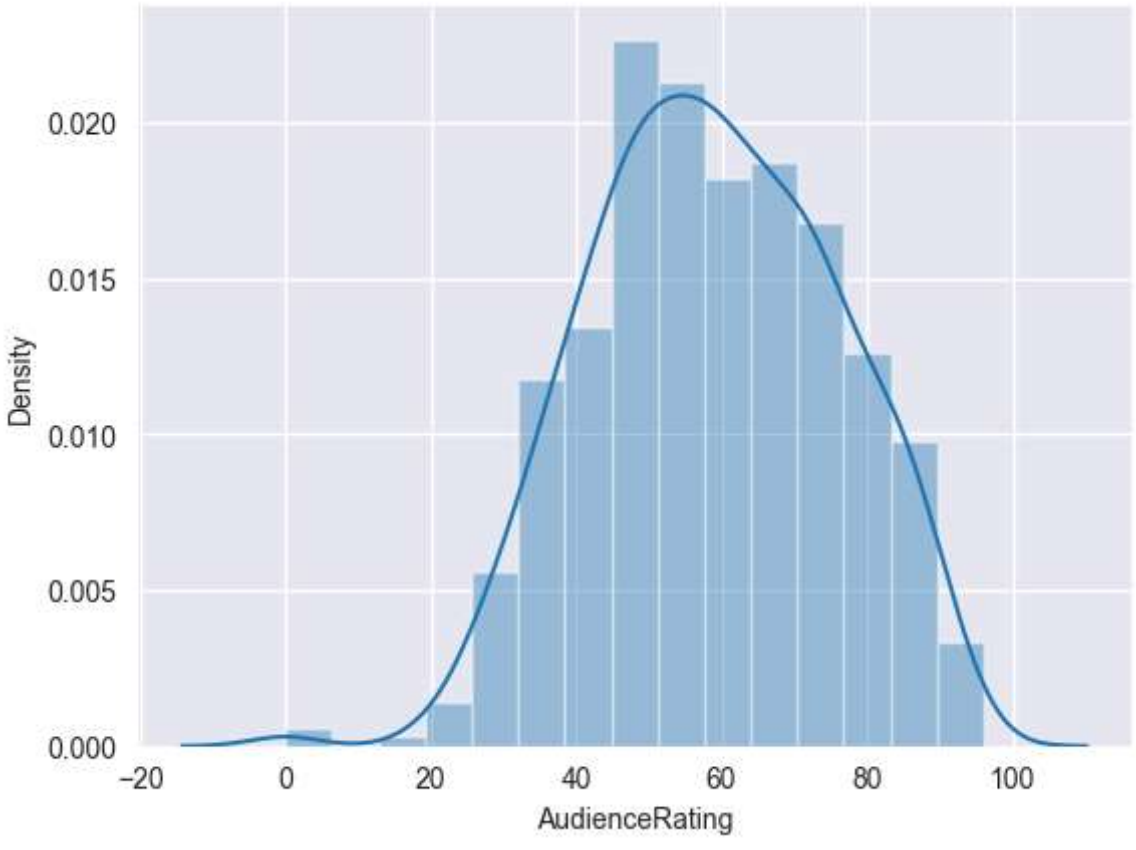


In [27]:
```
m1 = sns.distplot(movies.AudienceRating)
```

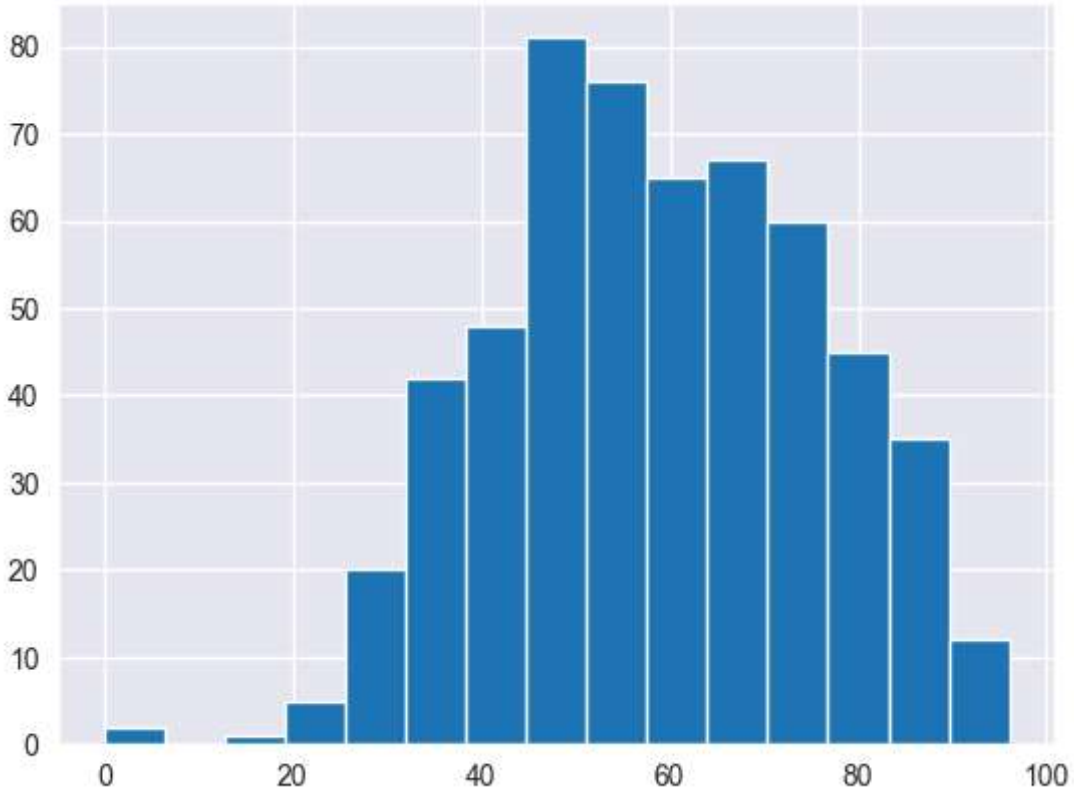


In [28]:
```
sns.set_style('darkgrid')
```

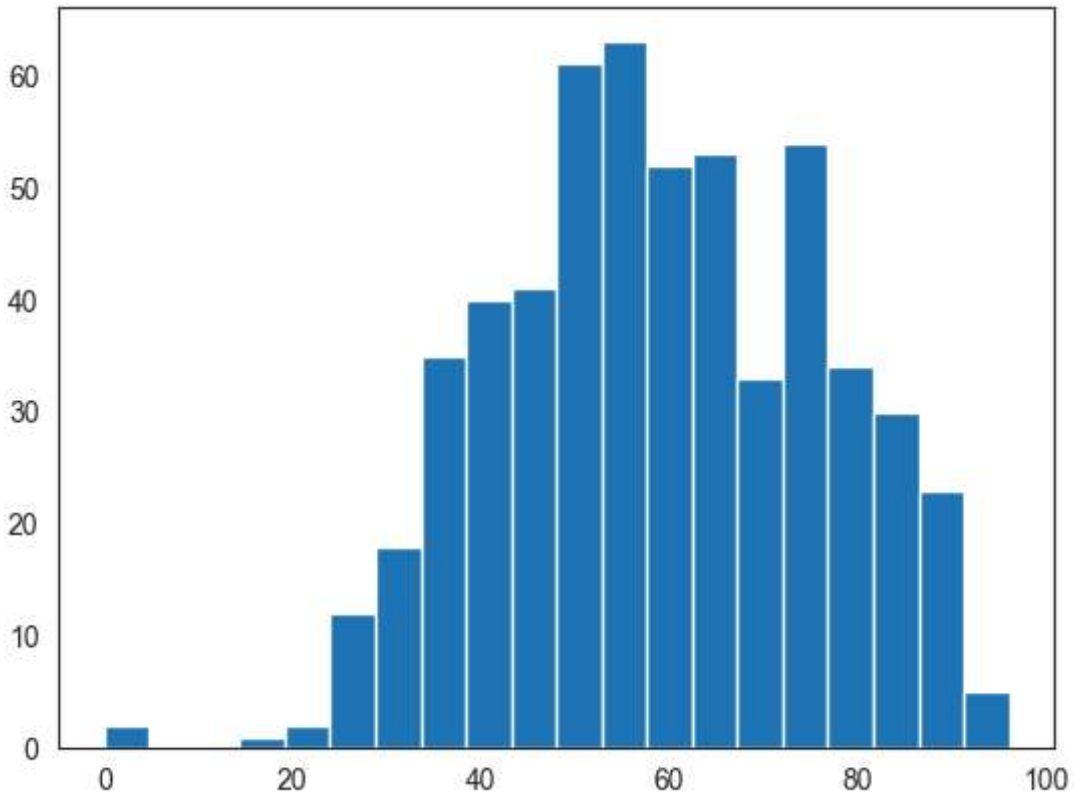
In [29]:
```
m2 = sns.distplot(movies.AudienceRating, bins = 15)
```



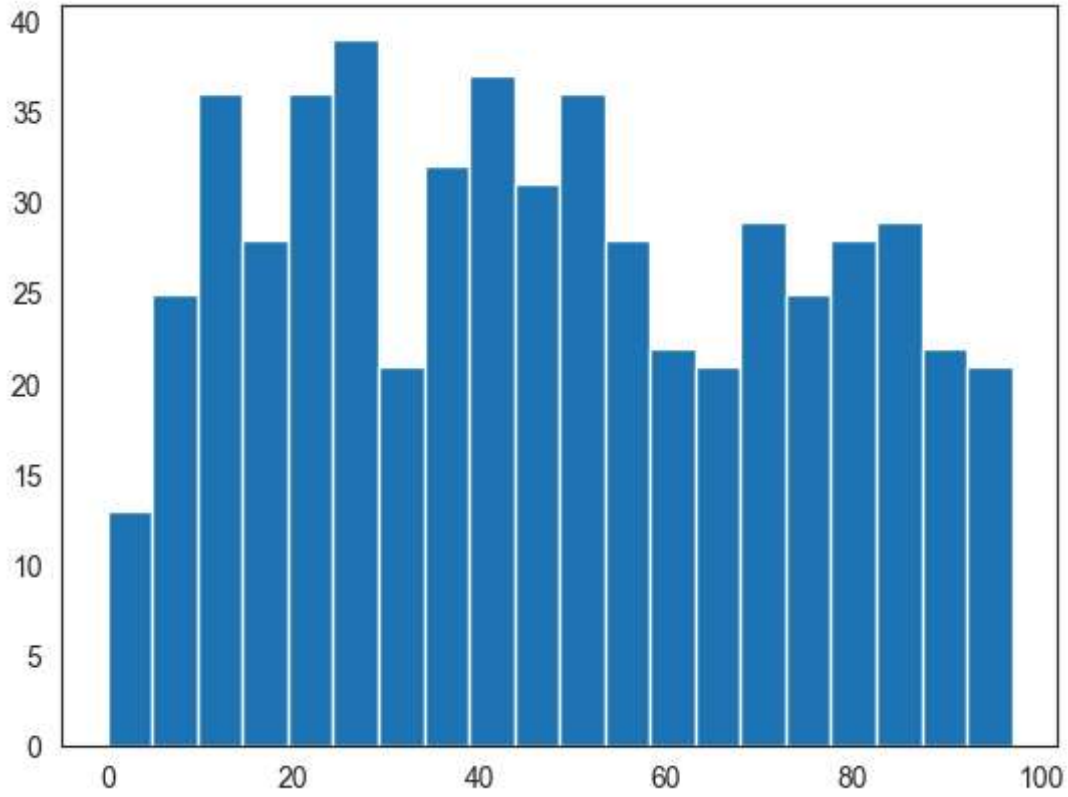
In [30]:
```
sns.set_style('darkgrid')
n1 = plt.hist(movies.AudienceRating, bins=15)
```



In [31]:
```
sns.set_style('white') #normal distribution & called as bell curve
n1 = plt.hist(movies.AudienceRating, bins=20)
```



In [32]:
```
n1 = plt.hist(movies.CriticRating, bins=20) #uniform distribution
```

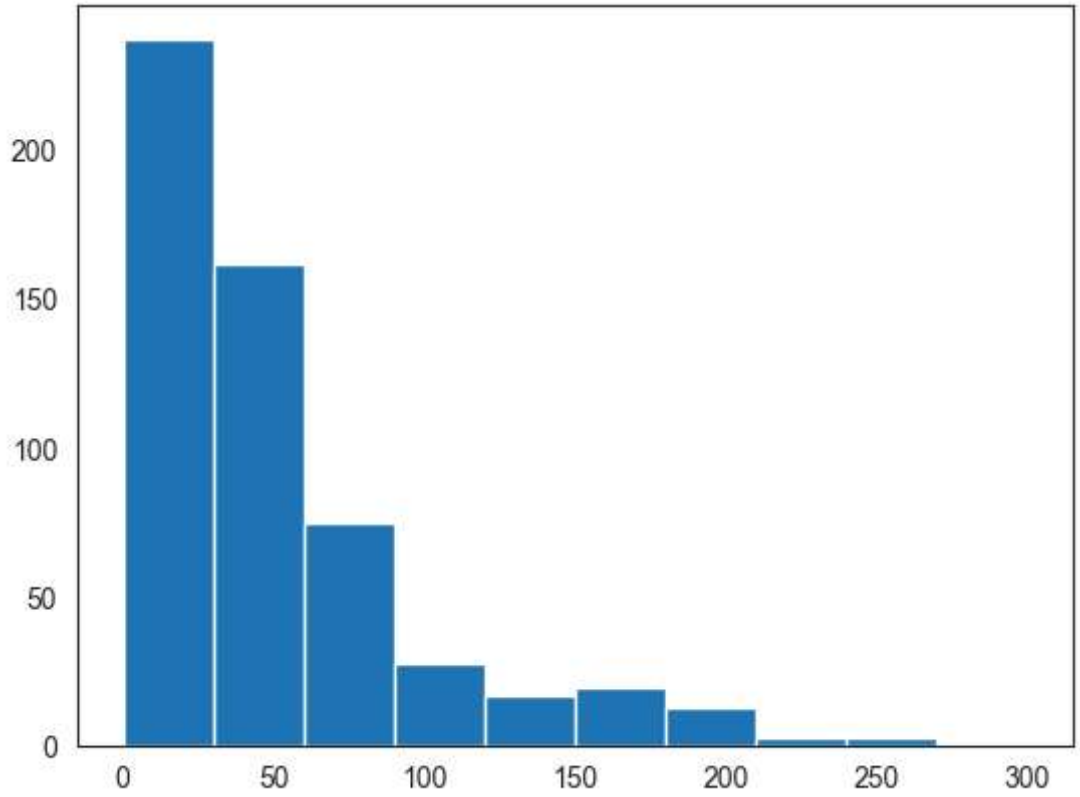


Stacked histograms

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

Out[33]:
```
Index(['Film', 'Genre', 'CriticRating', 'AudienceRating', 'BudgetMillions',
       'Year'],
      dtype='object')
```

In [35]:
```
plt.hist(movies.BudgetMillions)
plt.show()
```



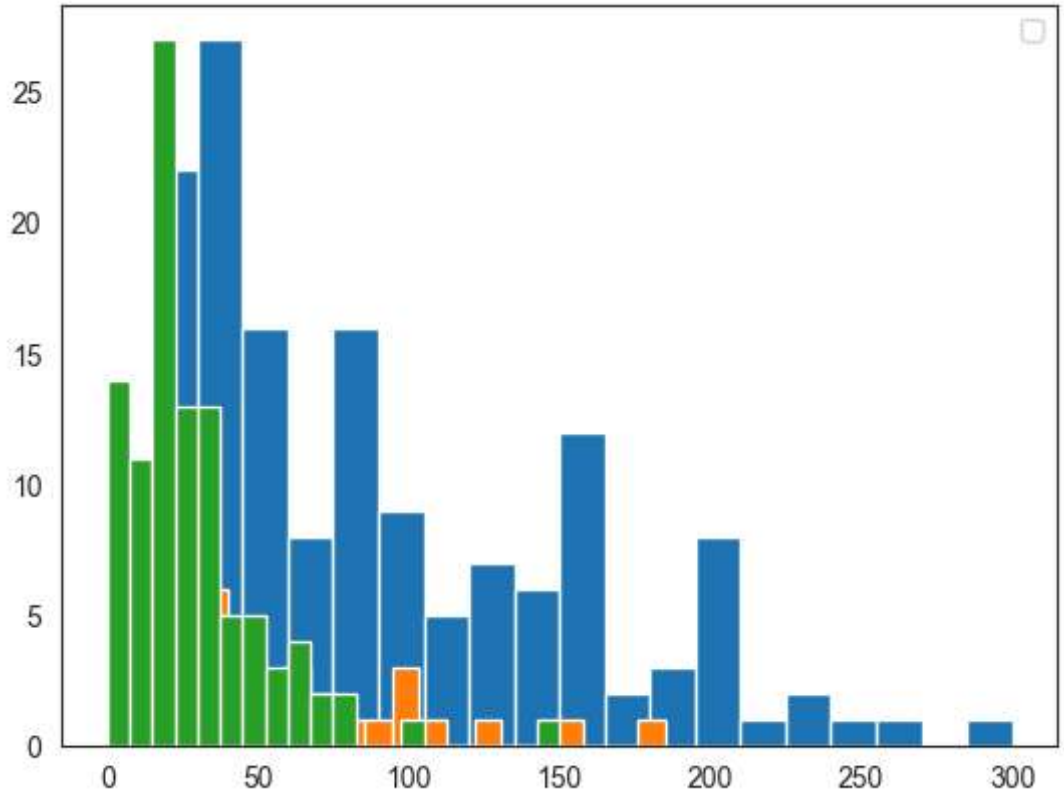
In [36]:
```
movies.Genre
```

Out[36]:
```
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', 'R
omance', 'Thriller']
```

In [37]:
```
movies.columns
```

Out[37]:
```
Index(['Film', 'Genre', 'CriticRating', 'AudienceRating', 'BudgetMillions',
       'Year'],
      dtype='object')
```

In [39]:
```
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()
```

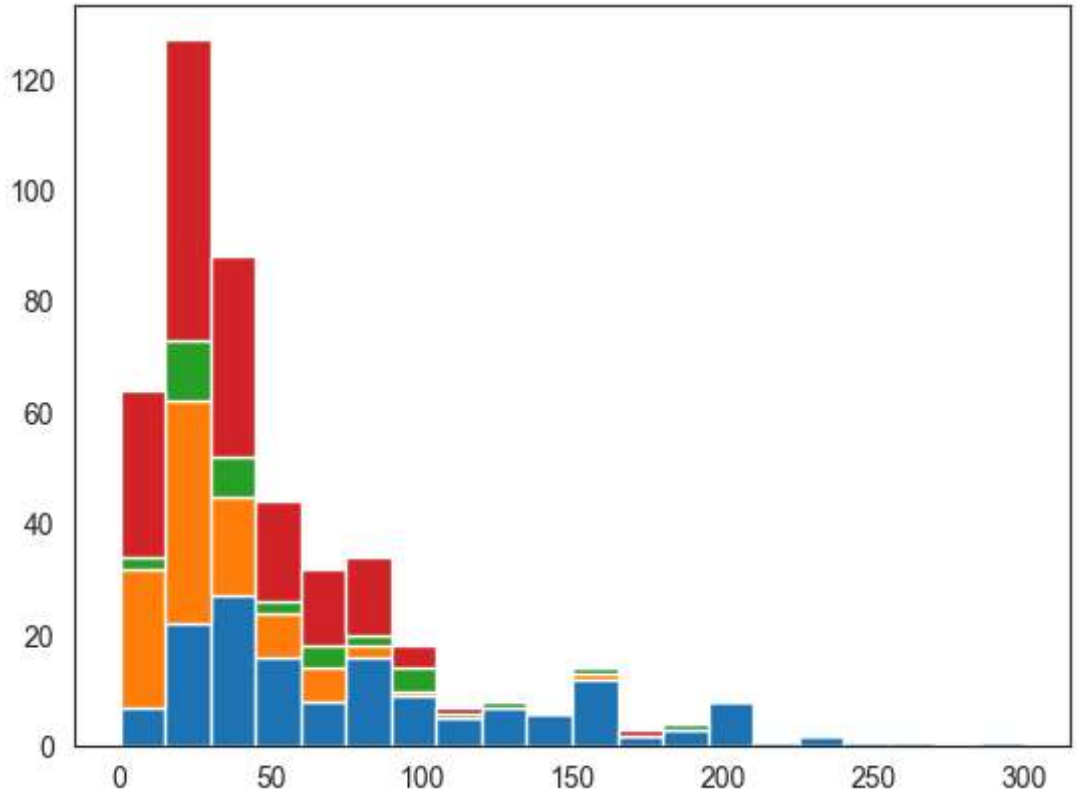


In [41]:
```
movies.Genre.unique()
```

Out[41]:
```
['Comedy', 'Adventure', 'Action', 'Horror', 'Drama', 'Romance', 'Thriller']
Categories (7, object): ['Action', 'Adventure', 'Comedy', 'Drama', 'Horror', 'R
omance', 'Thriller']
```

In [44]:
```
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
```

Out[44]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



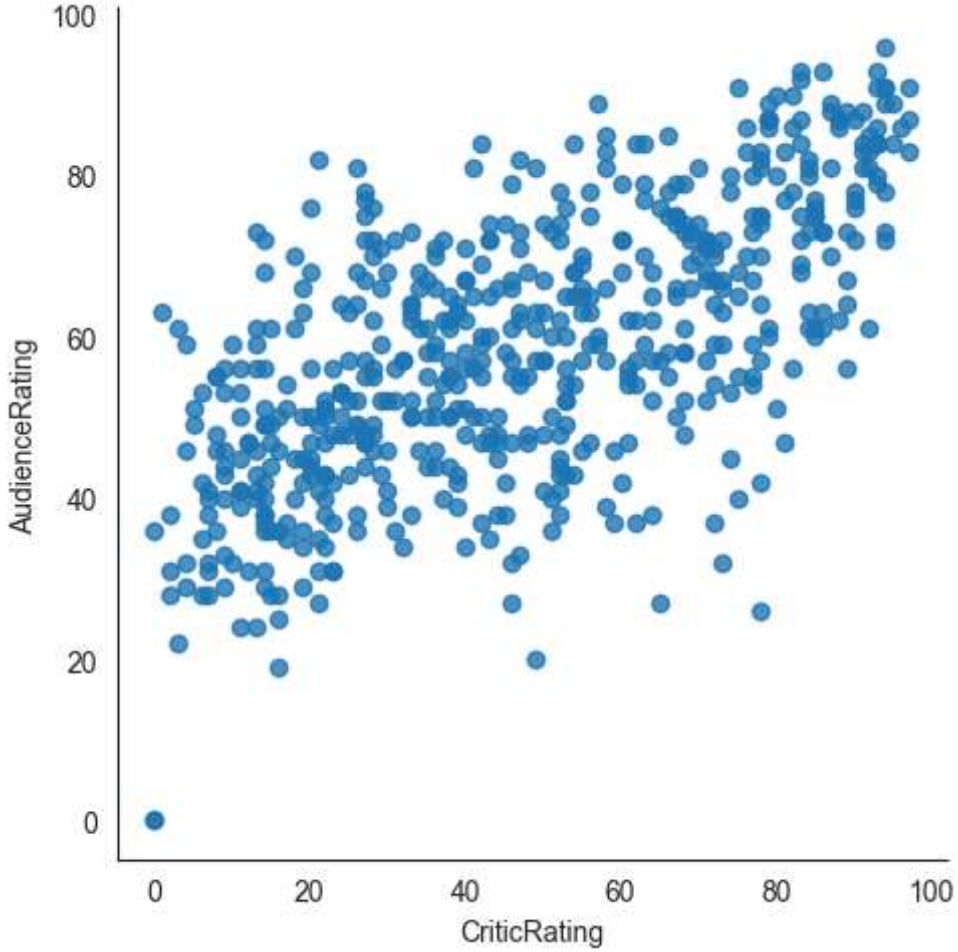
In [45]:
```
for gen in movies.Genre.cat.categories:
    print(gen)
```

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

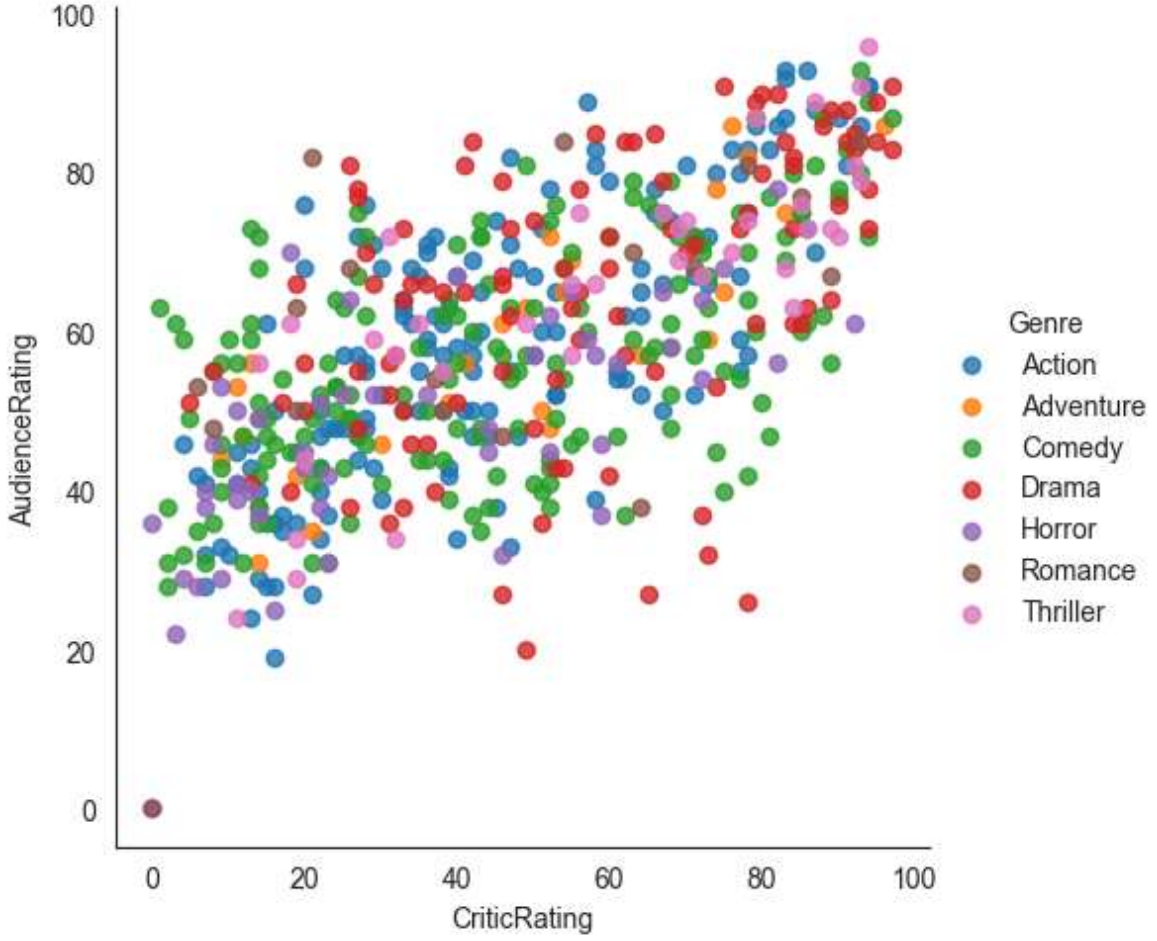
In [46]:
```
movies.info()
```

```
<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   CriticRating    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
```

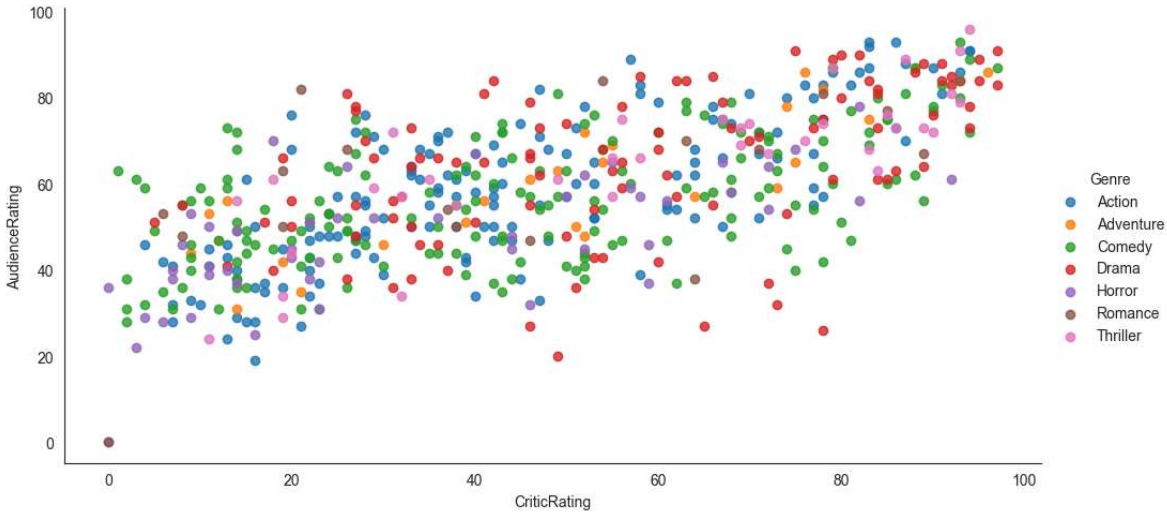
In [47]:
```
vis1 = sns.lmplot(data = movies, x= "CriticRating", y = 'AudienceRating', fit_re
```



In [48]:
```
vis1 = sns.lmplot(data=movies, x='CriticRating', y='AudienceRating',\
                  fit_reg=False, hue = 'Genre')
```

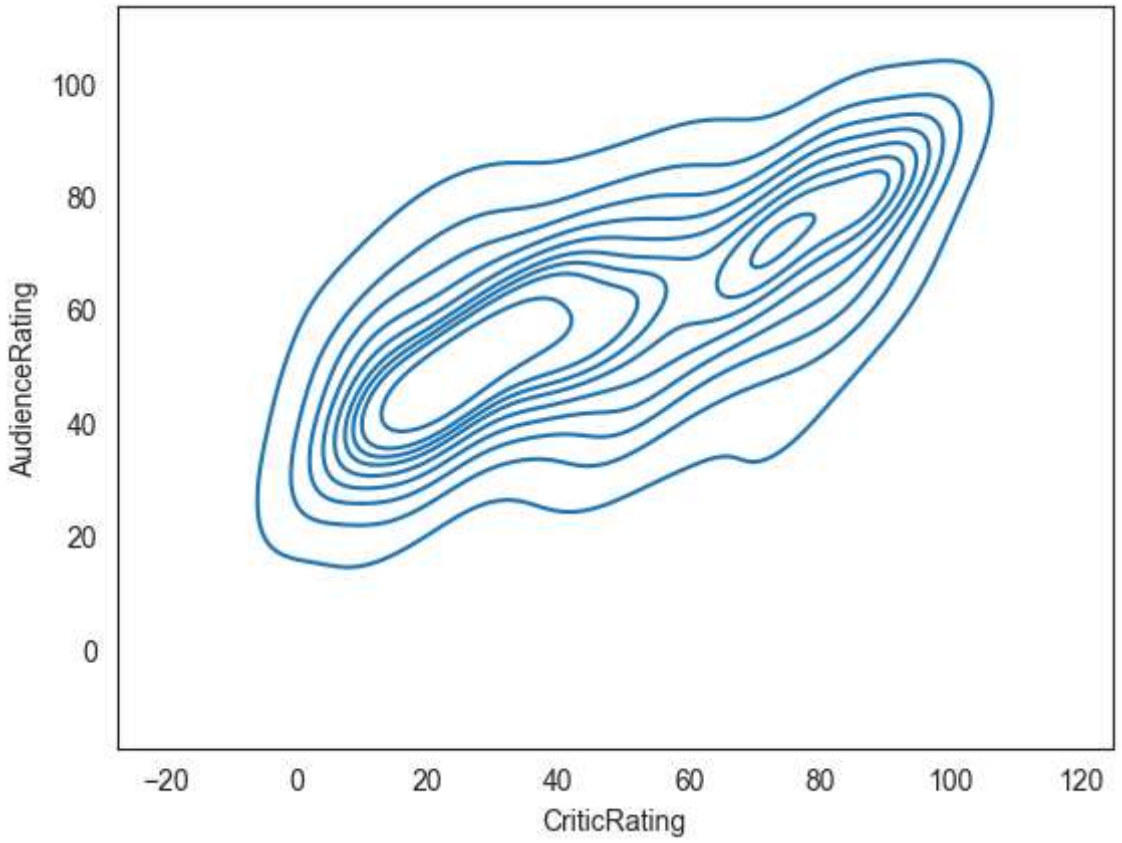


In [55]:
```
vis1 = sns.lmplot(data=movies, x='CriticRating', y='AudienceRating',fit_reg=Fals
```

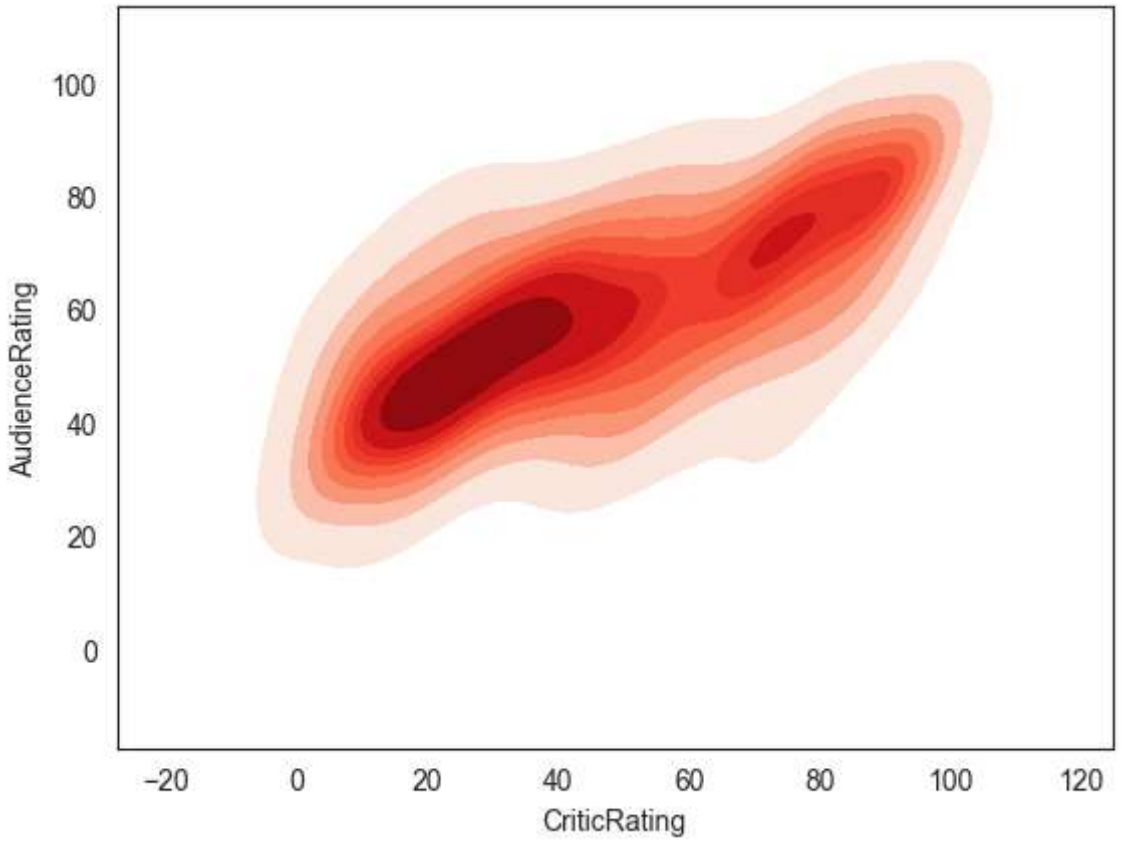


Kernal Density Estimate(KDE) Plot

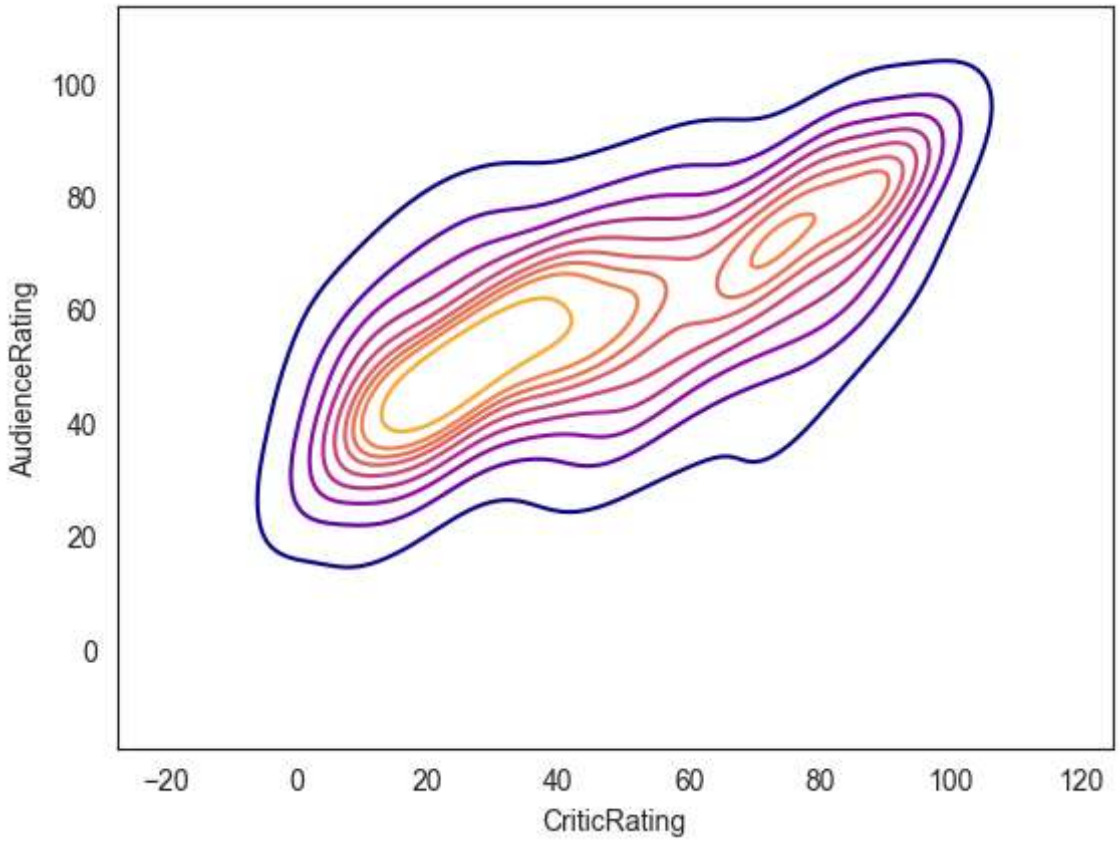
In [58]:
```
k1 = sns.kdeplot(x=movies.CriticRating, y = movies.AudienceRating)
```



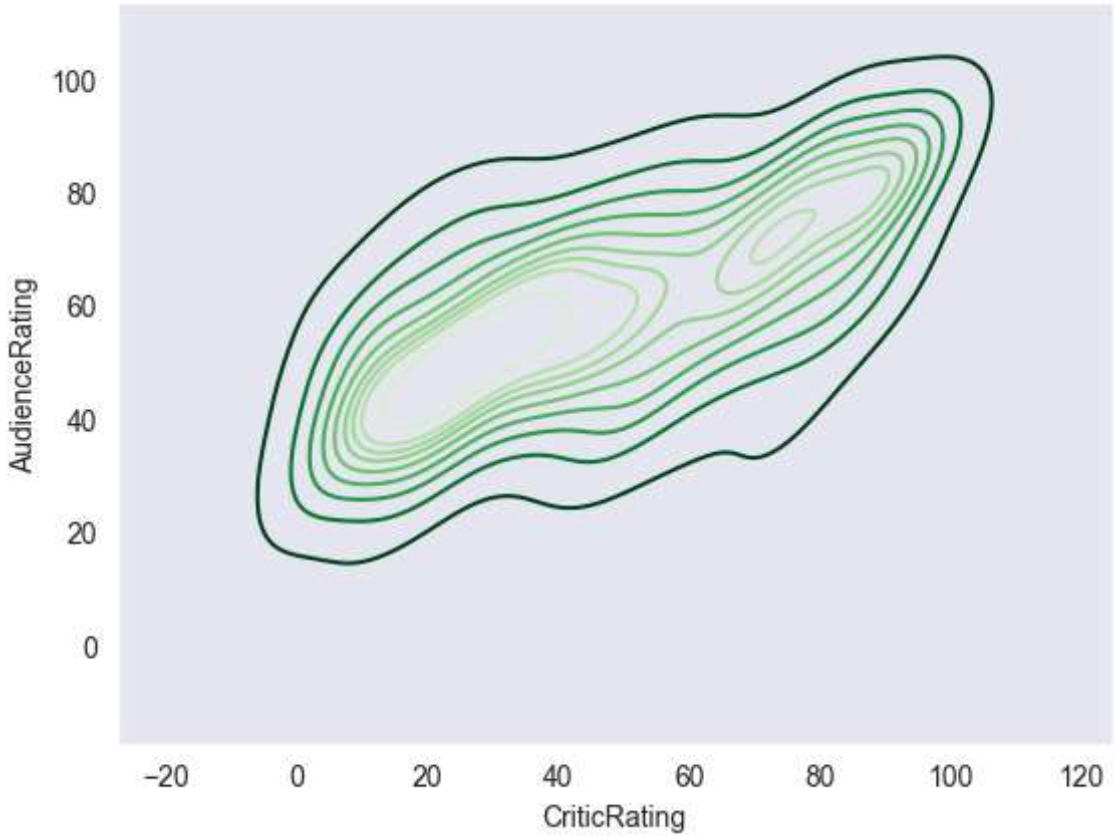
In [60]:
```
k1 = sns.kdeplot(x= movies.CriticRating, y = movies.AudienceRating,shade = True,
```



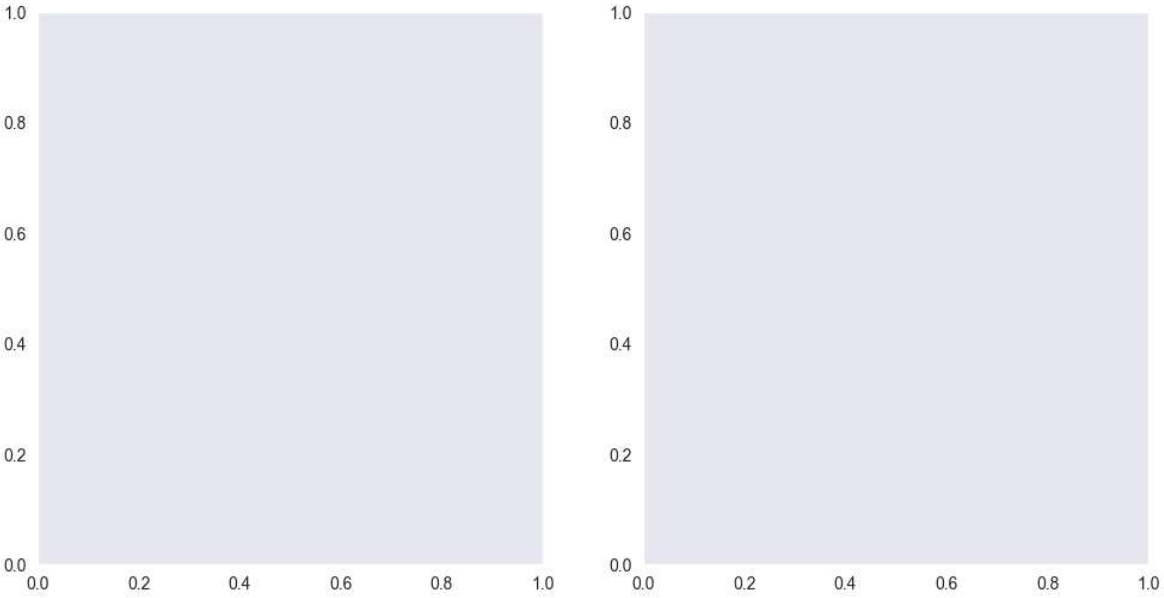
In [64]:
```
k1 = sns.kdeplot(x= movies.CriticRating, y = movies.AudienceRating,shade = False
```



In [66]:
```
sns.set_style('dark')
k1 = sns.kdeplot(x= movies.CriticRating, y = movies.AudienceRating,shade = False
```



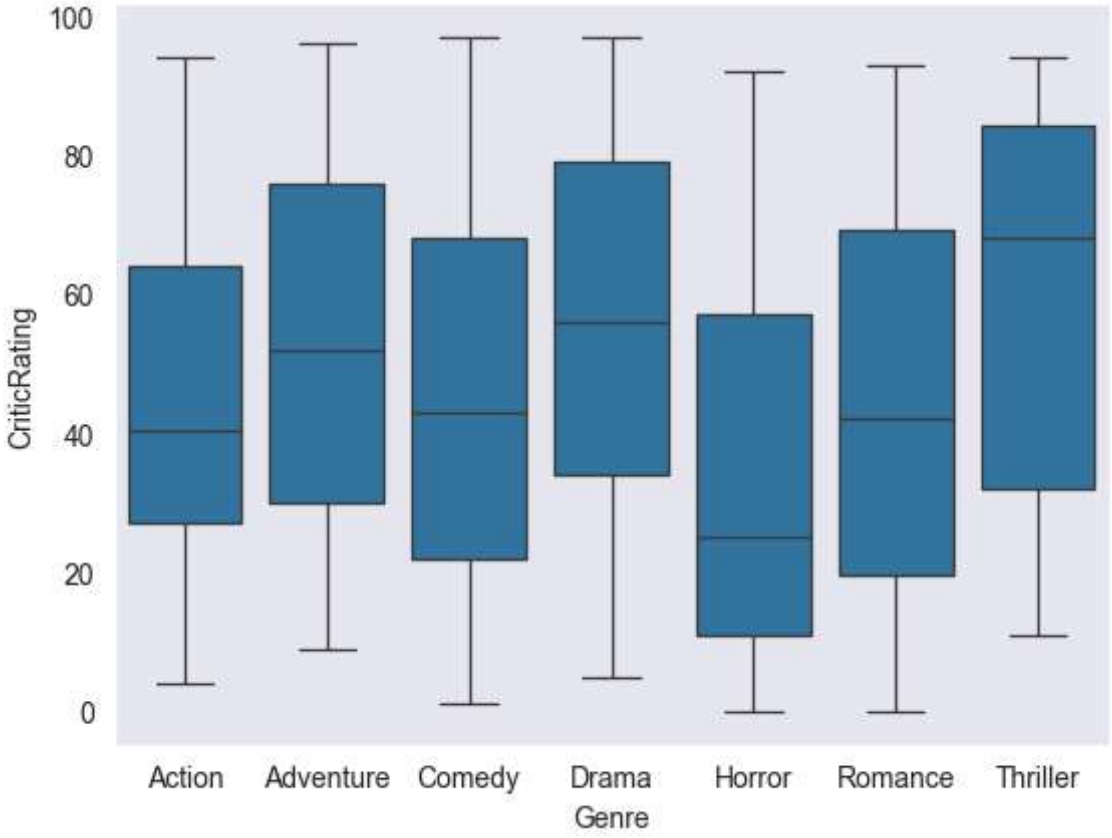
In [72]:
```
f, axes = plt.subplots(1,2, figsize= (12,6))
```



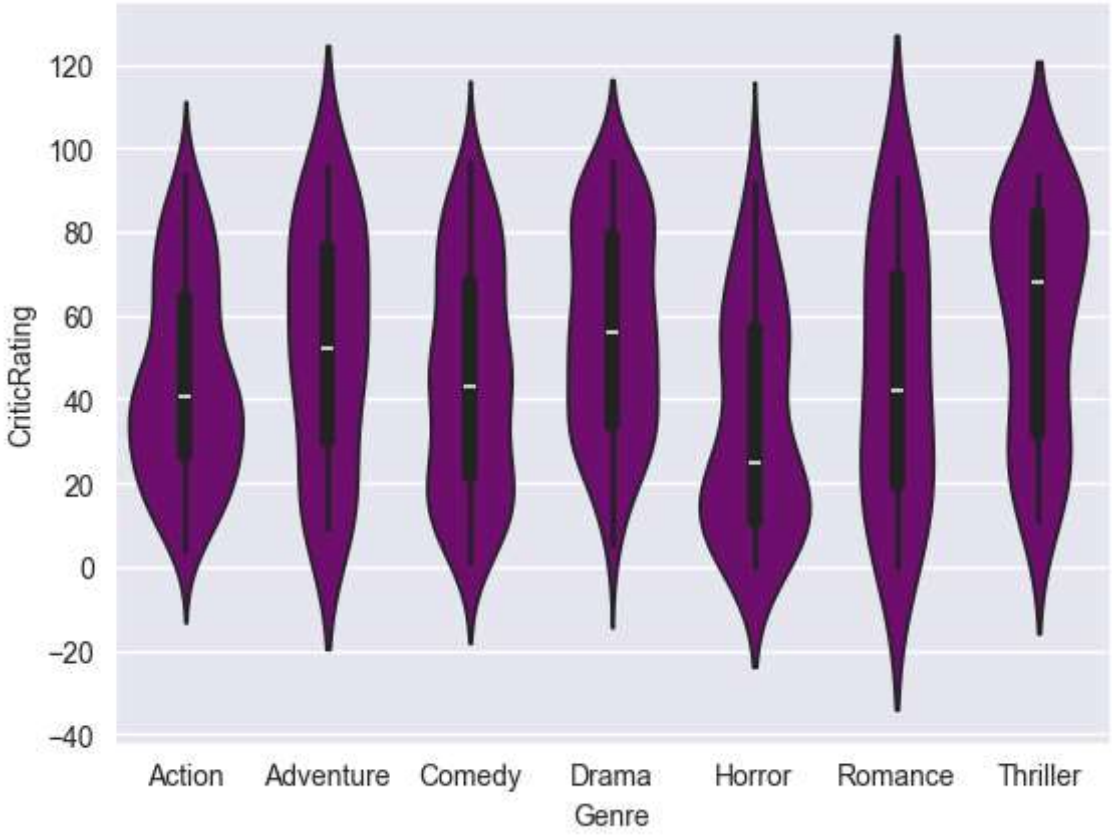
In [76]:
```
k1 = sns.kdeplot(x=movies.BudgetMillions,y=movies.AudienceRating,ax=axes[0])
k2 = sns.kdeplot(x=movies.BudgetMillions,y=movies.CriticRating,ax = axes[1])
k1
```

Out[76]: <Axes: xlabel='BudgetMillions', ylabel='AudienceRating'>

In [77]:
```
# Box plot
w = sns.boxplot(data=movies, x='Genre', y = 'CriticRating')
```

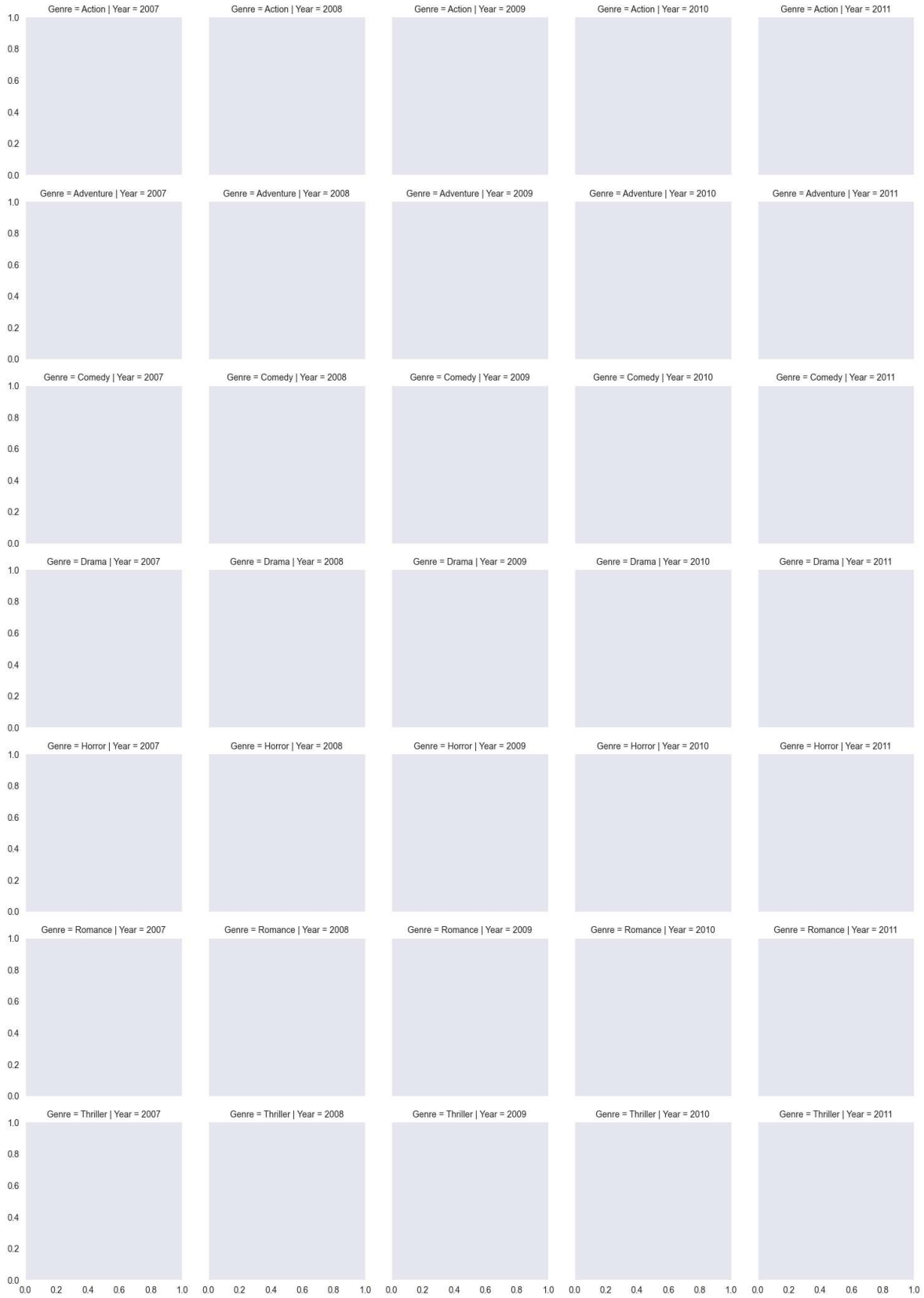


In [88]:
```
#violin plot

z = sns.violinplot(data=movies, x='Genre', y = 'CriticRating', color = 'purple')
```



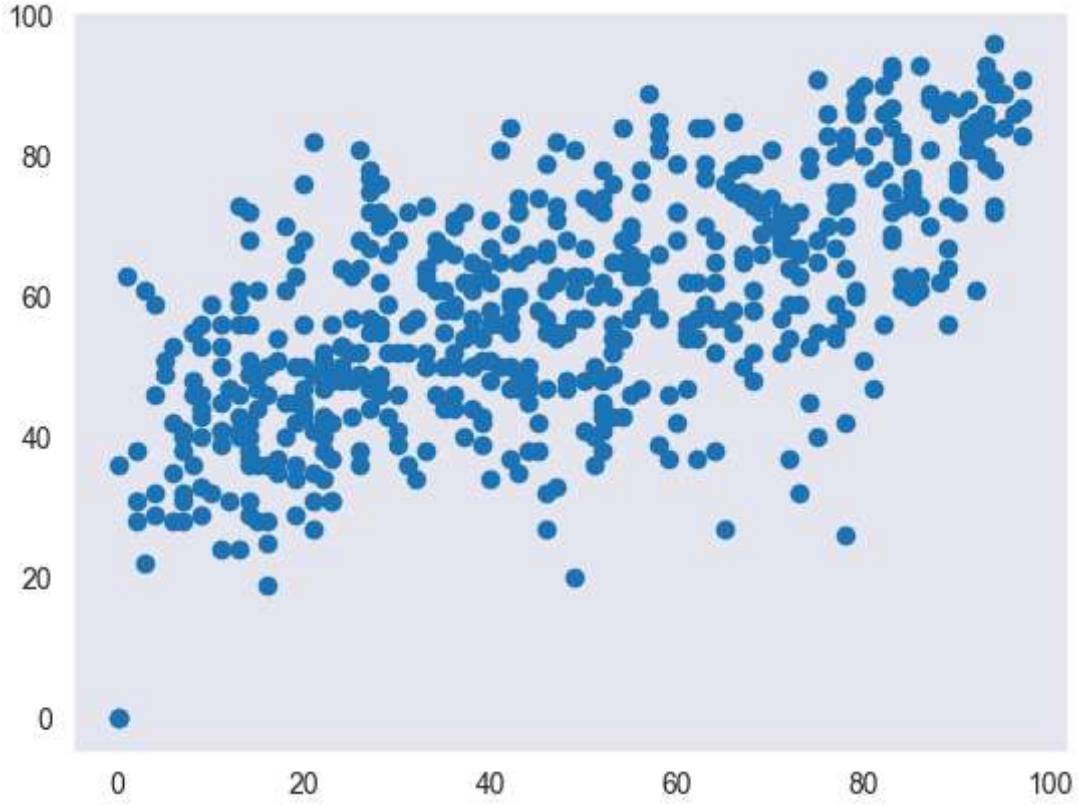
Creating a Facet Grid

In [79]:
```
g = sns.FacetGrid(movies, row = 'Genre', col = 'Year', hue = 'Genre')
```

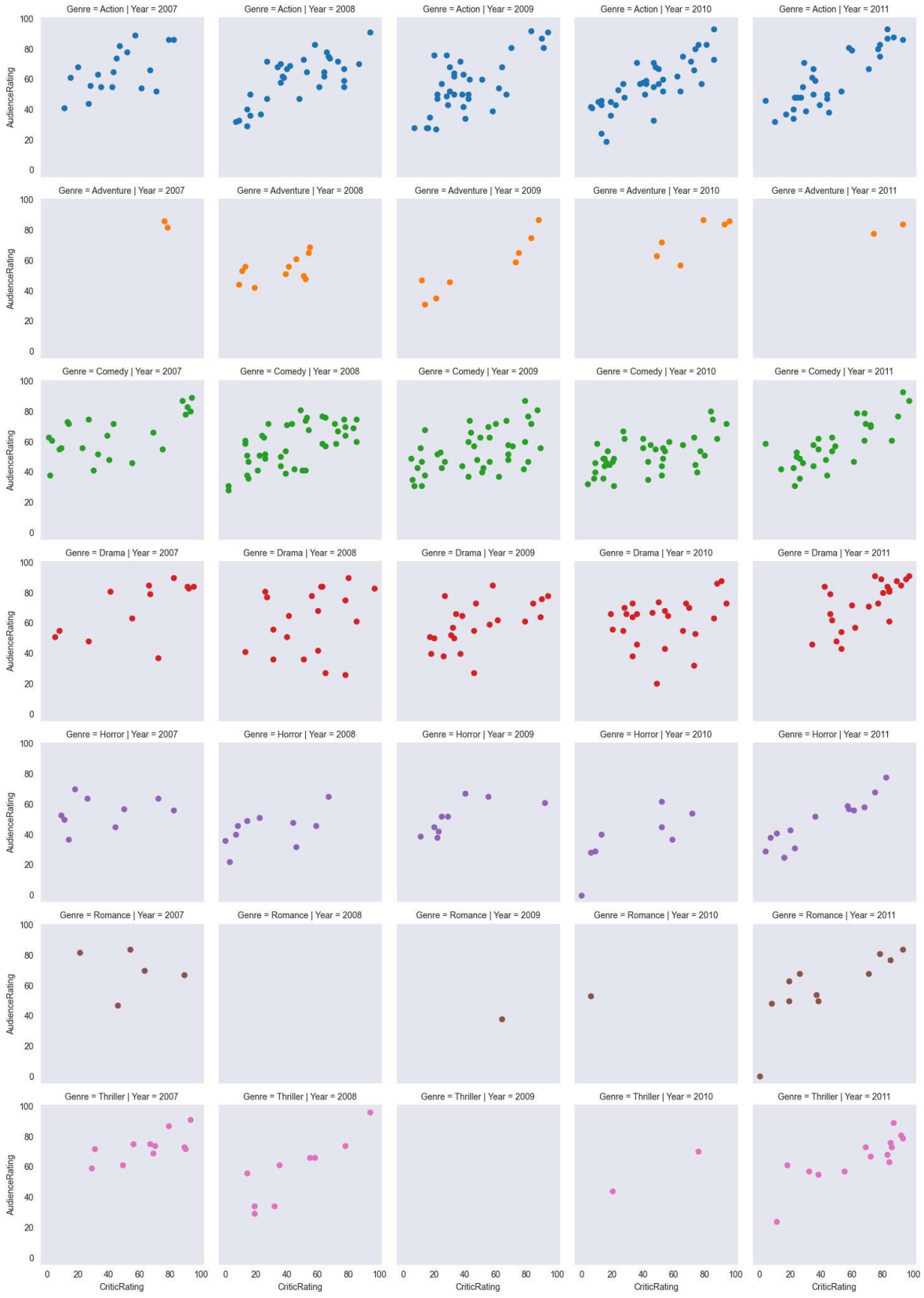


In [80]:
```
plt.scatter(movies.CriticRating,movies.AudienceRating)
```

Out[80]:
```
<matplotlib.collections.PathCollection at 0x2212aee9090>
```

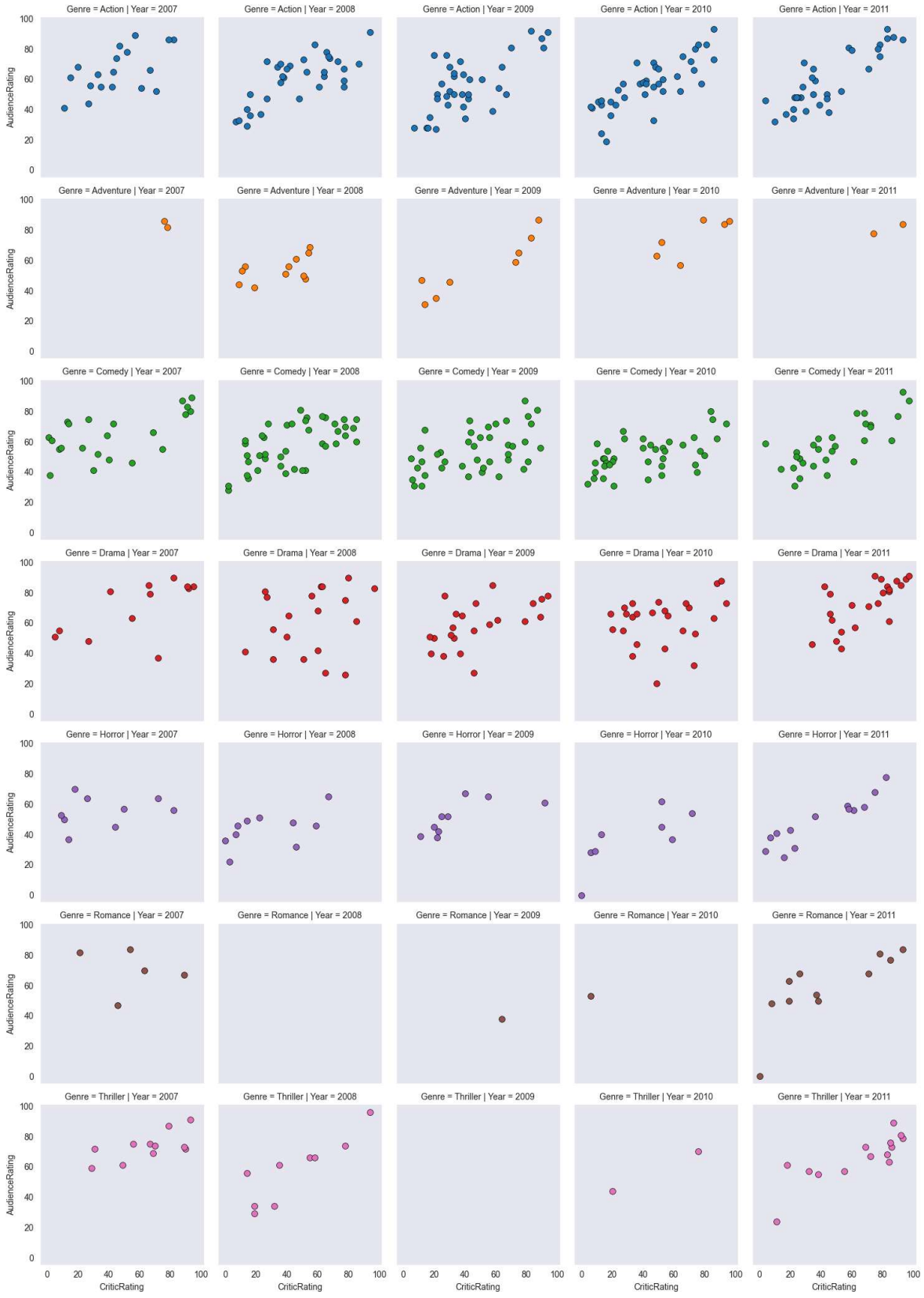


In [81]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.scatter, 'CriticRating', 'AudienceRating' ) #scatterplots are mapp
```



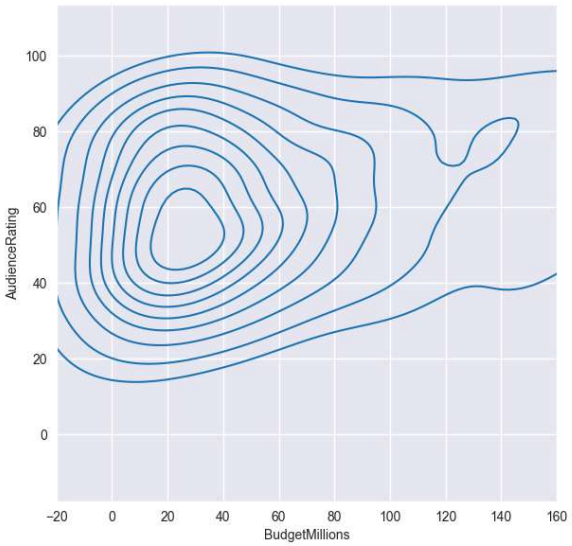
In [83]:
```
# we can put any type of graph in facet grid
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
g = g.map(plt.hist, 'BudgetMillions' )
```

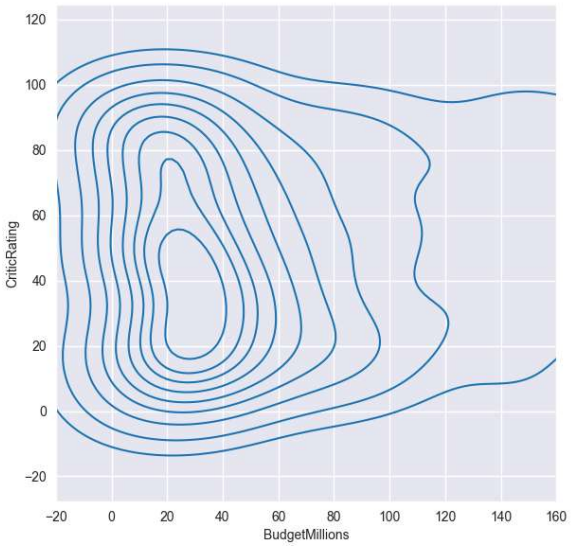
In [84]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre')
kws = dict(s=50, linewidth=0.5,edgecolor='black')
g = g.map(plt.scatter, 'CriticRating', 'AudienceRating',**kws )
```

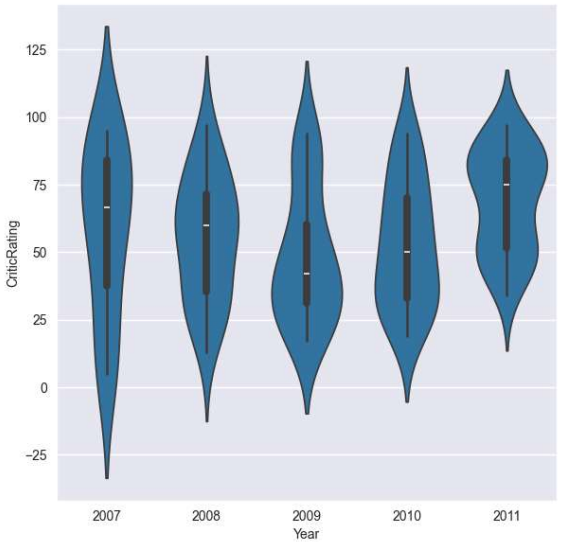


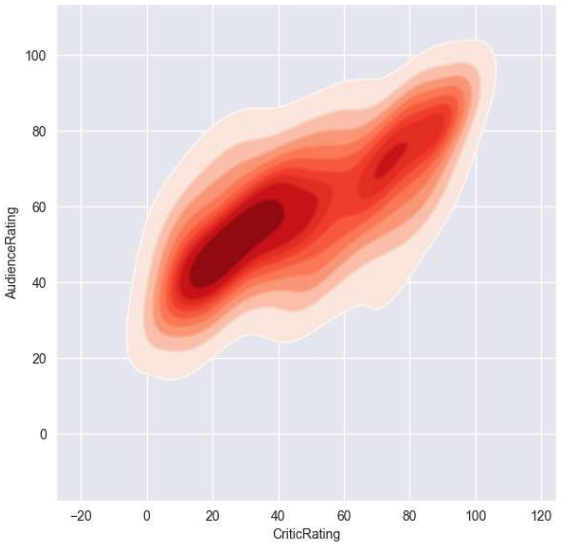
In [86]:
```
#Building Dashboards (Combination of charts or plots)

sns.set_style('darkgrid')
f, axes = plt.subplots (2,2, figsize = (15,15))

k1 = sns.kdeplot( x = movies.BudgetMillions,y = movies.AudienceRating,ax=axes[0,
k2 = sns.kdeplot(x=movies.BudgetMillions,y = movies.CriticRating,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(x = movies.CriticRating,y = movies.AudienceRating,shade = True,

k4b = sns.kdeplot(x=movies.CriticRating,y= movies.AudienceRating,cmap='Reds',ax

plt.show()
```









In [ ]: