ion-movie-rating-suggetion-project

September 4, 2024

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

[4]:
```
movies=pd.read_csv(r'/Movie-Rating.csv')
```

[5]:
```
movies
```

[5]:
```
                      Film      Genre  Rotten Tomatoes 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]
```

[6]:
```
len(movies)
```

[6]:
```
559
```

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

[7]:
```
                    Film      Genre  Rotten Tomatoes 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

   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
```

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

[8]:
```
                Film     Genre  Rotten Tomatoes Ratings %  Audience Ratings %  \
554    Your Highness    Comedy                         26                  36
555  Youth in Revolt    Comedy                         68                  52
556           Zodiac  Thriller                         89                  73
557      Zombieland     Action                         90                  87
558        Zookeeper    Comedy                         14                  42

     Budget (million $)  Year of release
554                  50             2011
555                  18             2009
556                  65             2007
557                  24             2009
558                  80             2011
```

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

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

[10]:
```
movies.
↪columns=['Film','Genre','CriticRating','AudienceRating','BudgetMillions','Year']
```

[13]:
```
movies.head()
```

[13]:
```
                    Film      Genre  CriticRating  AudienceRating  \
0  (500) Days of Summer      Comedy            87              81
1            10,000 B.C.  Adventure             9              44
2             12 Rounds      Action            30              52
```

```
3              127 Hours  Adventure            93              84
4              17 Again      Comedy            55              70

   BudgetMillions  Year
0               8  2009
1             105  2008
2              20  2009
3              18  2010
4              20  2009
```

[14]:
```
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
```

[15]:
```
movies.describe()
```

[15]:

	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

[16]:
```
movies['Film']
```

[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: object
```

[18]:
```
movies.Film=movies.Film.astype('category')
```

[19]:
```
movies.Film
```

[19]:
```
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']
```

[20]:
```
movies.head()
```

[20]:

	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

[21]:
```
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   CriticRating     559 non-null    int64
 3   AudienceRating   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
```

[22]:
```
movies.Genre = movies.Genre.astype('category')
movies.Year = movies.Year.astype('category')
```

[23]:
```
movies.Genre
```

[23]:
```
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']
```

[24]:
```
movies.Year
```

[24]:
```
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]
```

[25]:
```
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
```

[26]:
```
movies.describe()
```

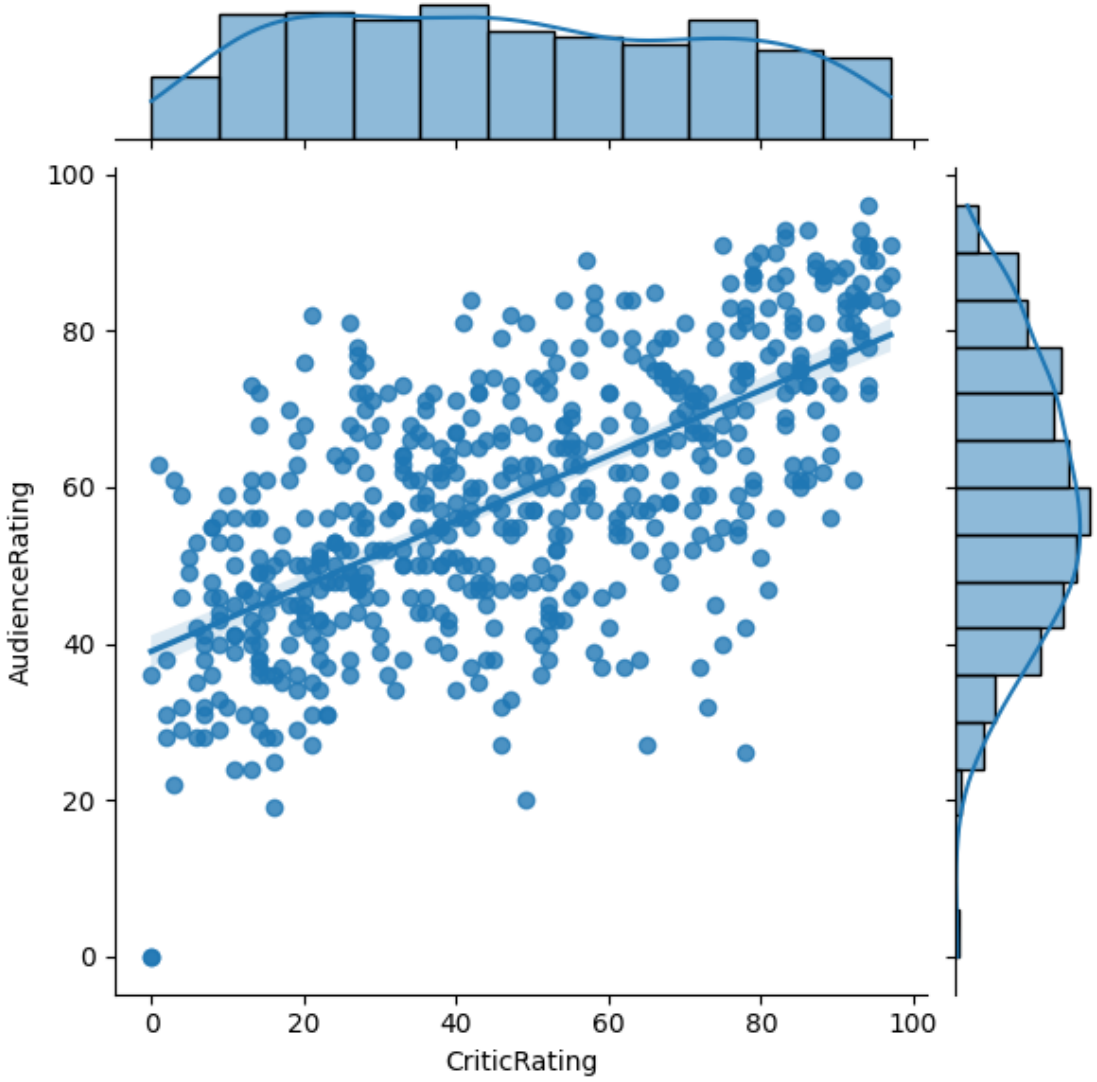
[26]:

	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

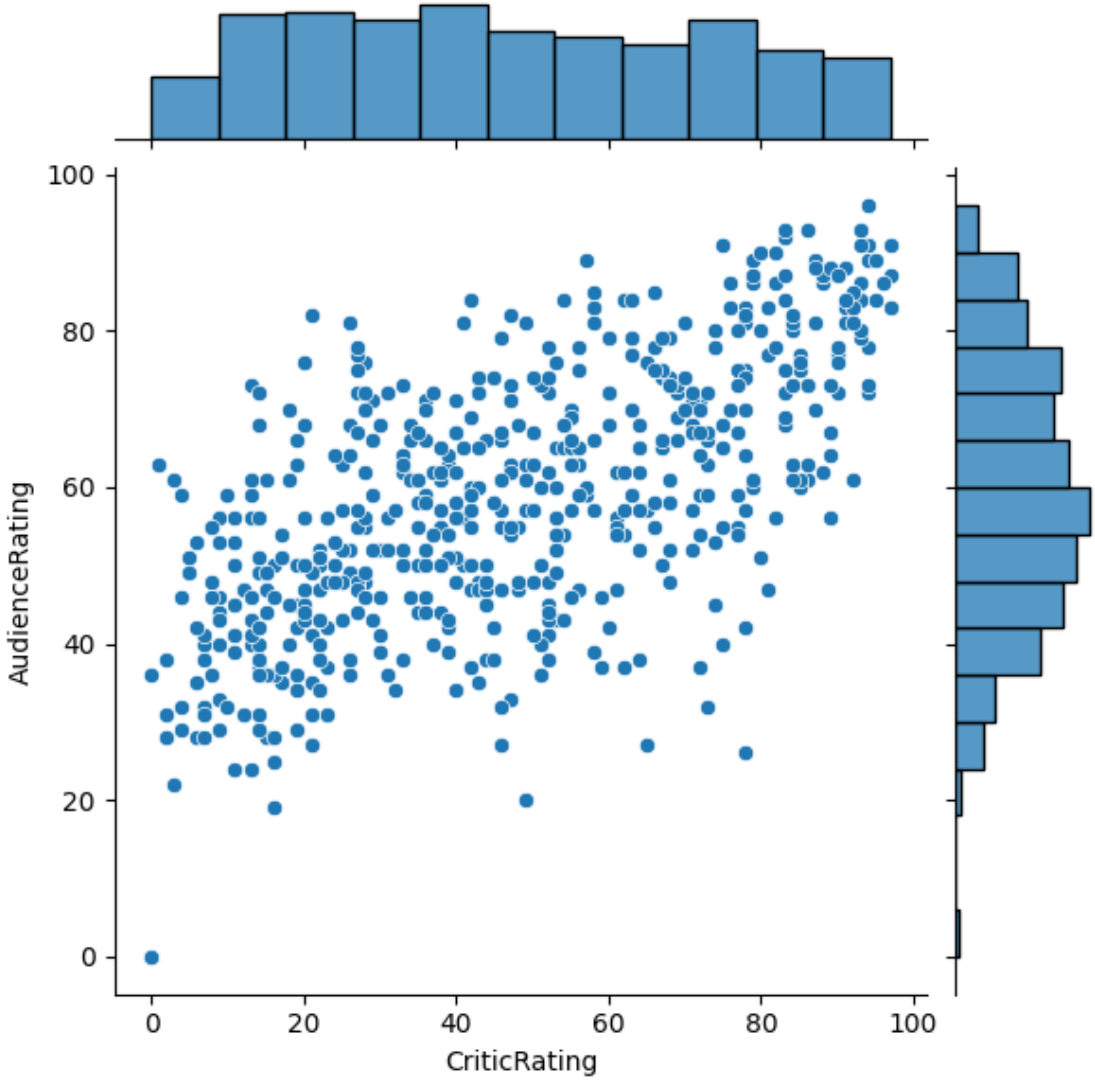
1 working with joint plots:

[30]:
```
from matplotlib import pyplot as pyplot
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

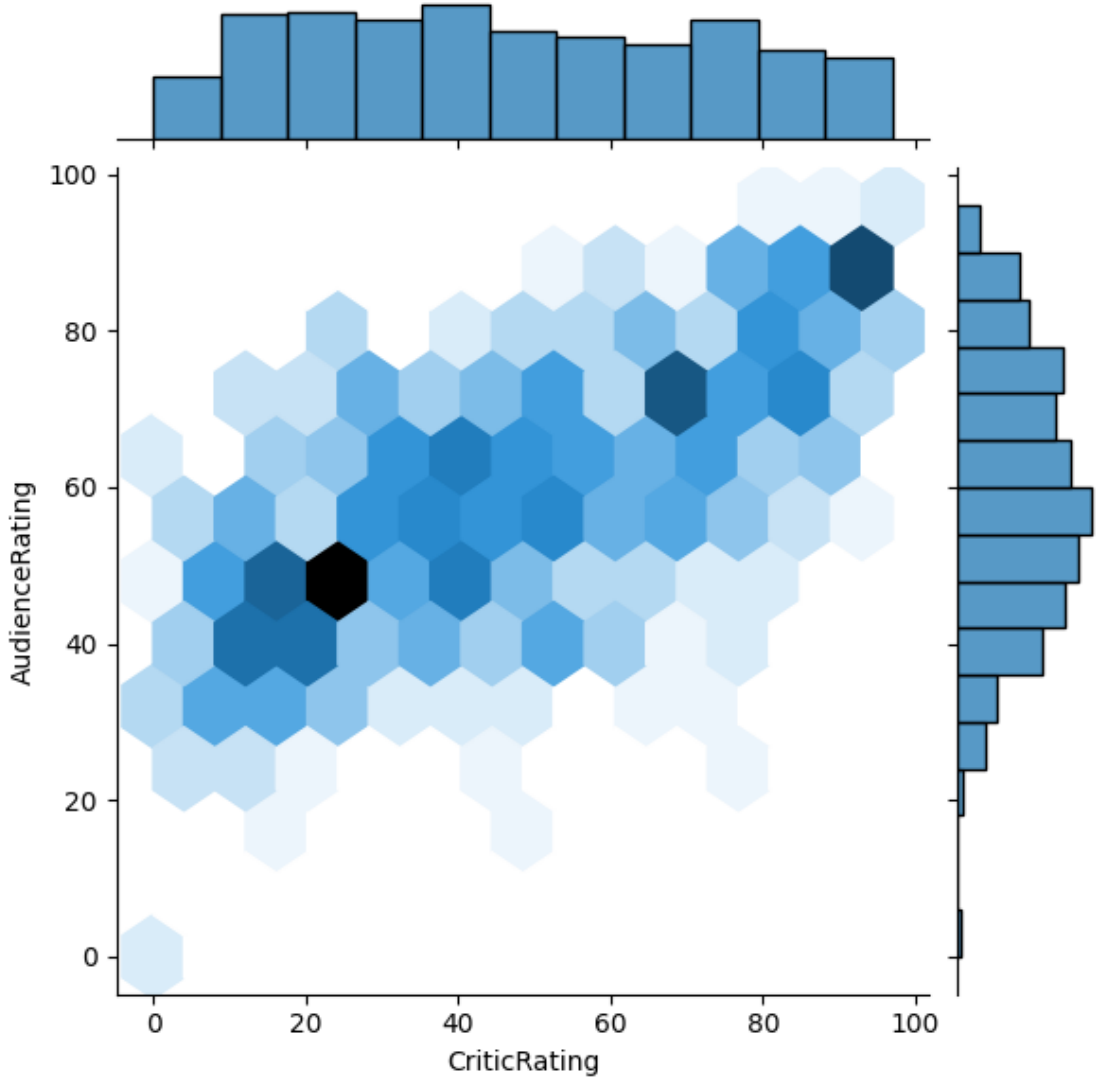
[31]:
```
j=sns.jointplot(data= movies, x = 'CriticRating', y='AudienceRating', kind =␣
↪'reg' )
```



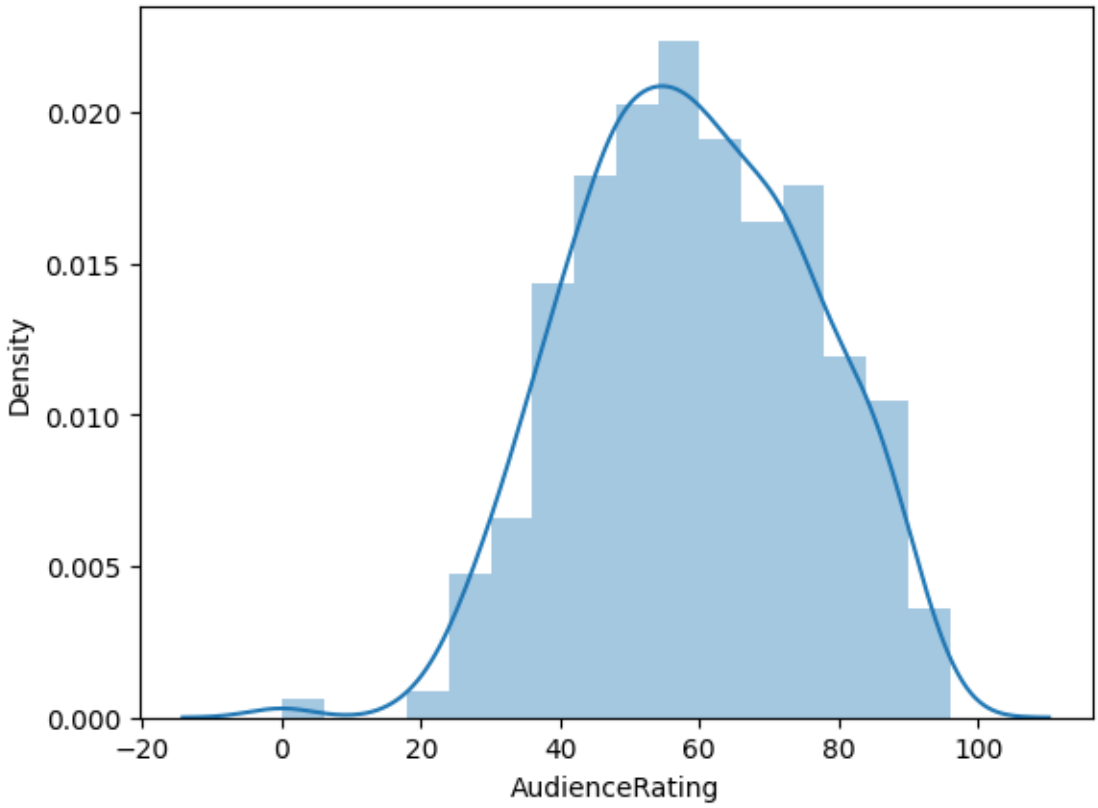
[36]:
```
j=sns.jointplot(data=movies,x='CriticRating', y='AudienceRating',kind=␣
↪'scatter')
```



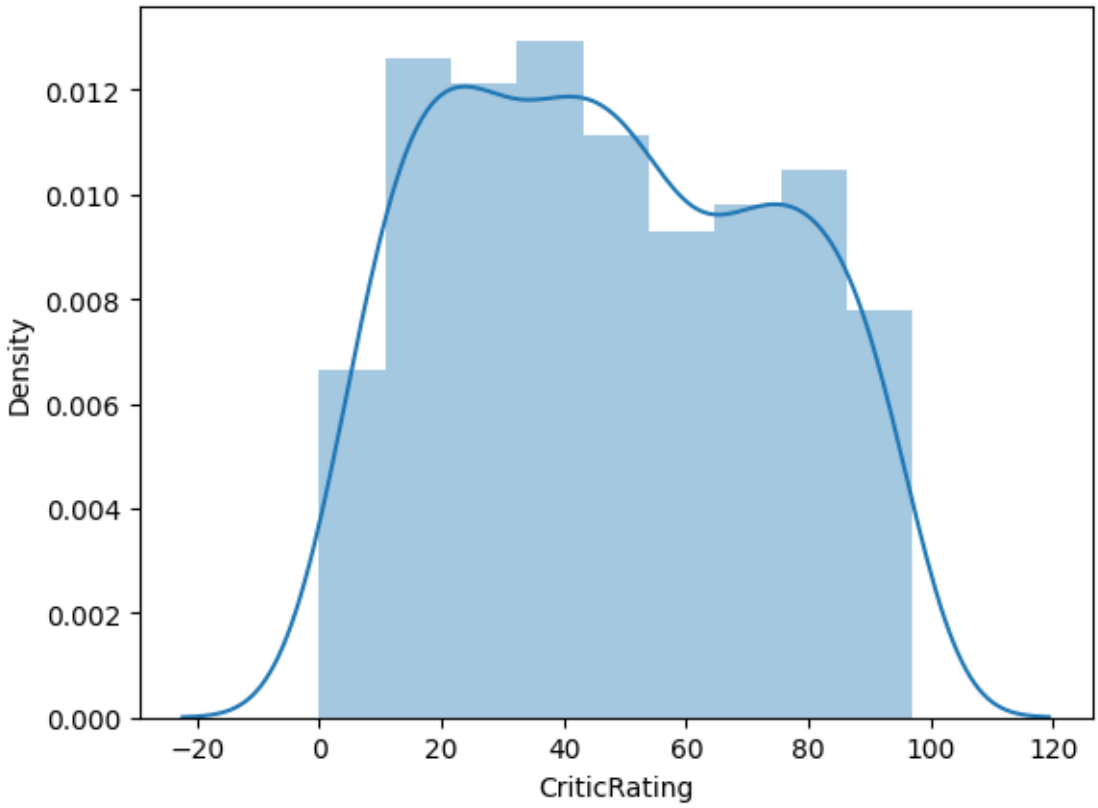
[37]:
```
j=sns.jointplot(data=movies,x='CriticRating',y='AudienceRating',kind='hex')
```



[38]:
```
# HISTOGRAMS
m1=sns.distplot(movies.AudienceRating)
```

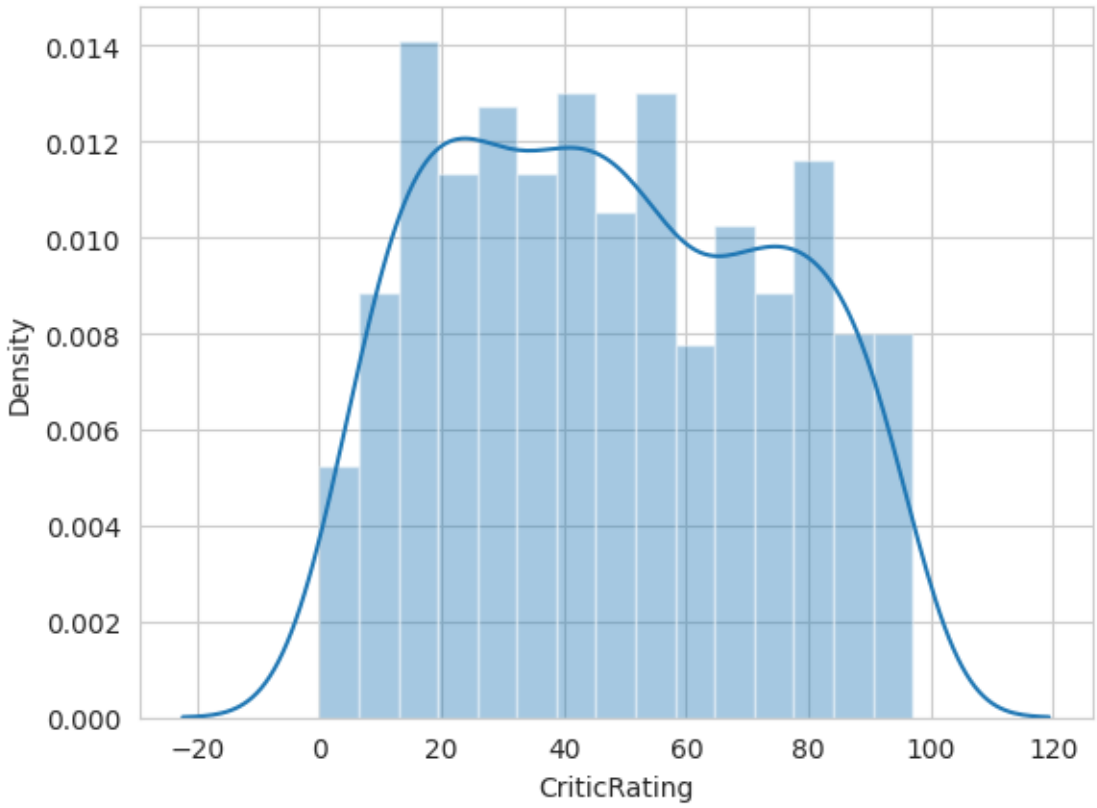


[39]:
```
# HISTOGRAMS
m1=sns.distplot(movies.CriticRating)
```

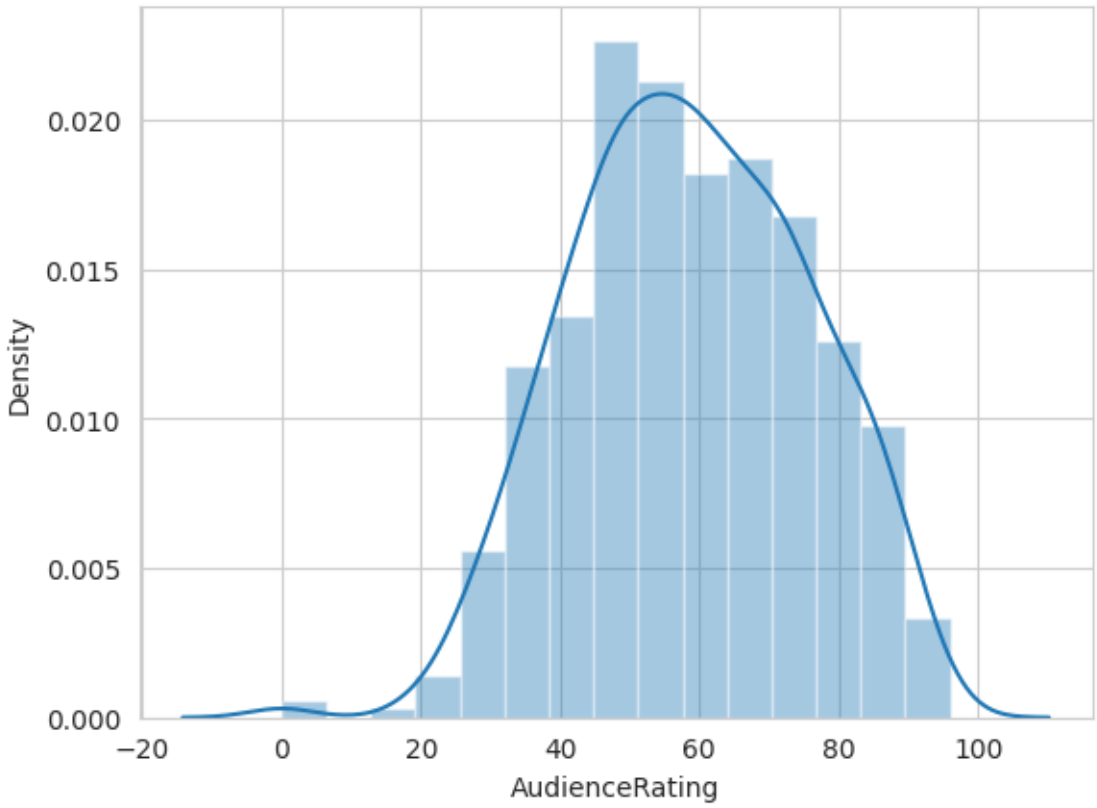


[40]:
```
sns.set_style('whitegrid')
```

[44]:
```
m2=sns.distplot(movies.CriticRating ,bins=15)
```

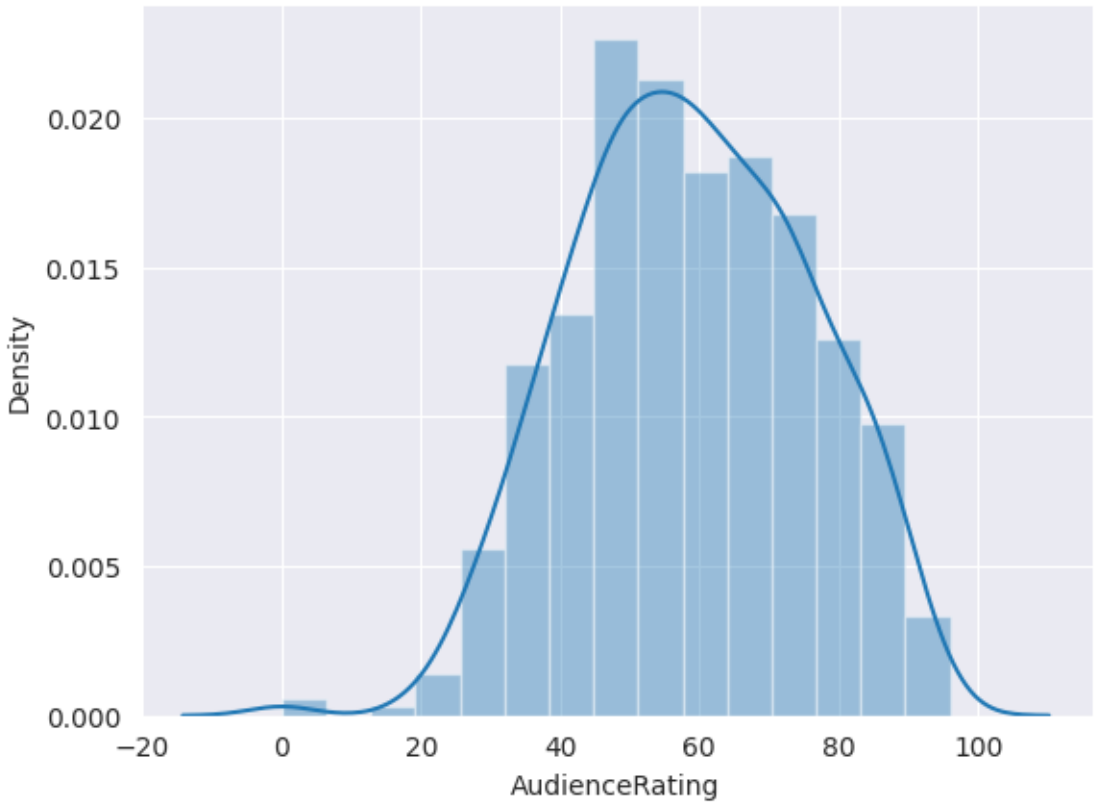


[45]:
```
m2=sns.distplot(movies.AudienceRating ,bins=15)
```

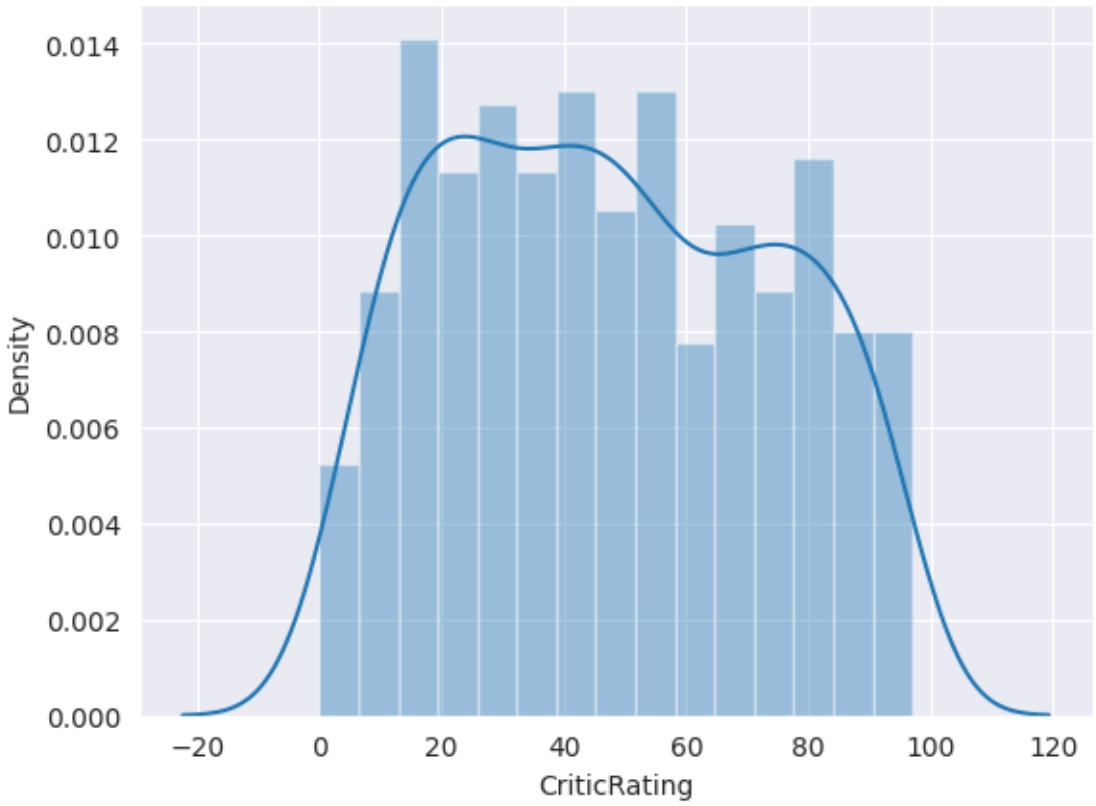


[46]:
```
sns.set_style('darkgrid')
```

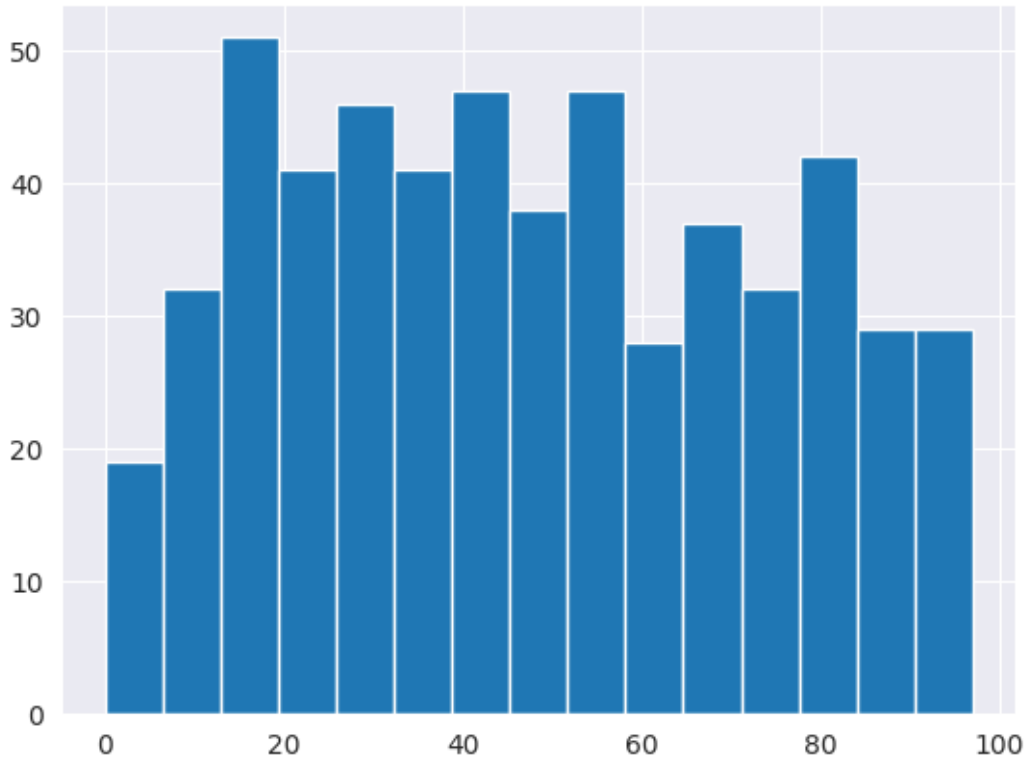
[47]:
```
m2=sns.distplot(movies.AudienceRating ,bins=15)
```



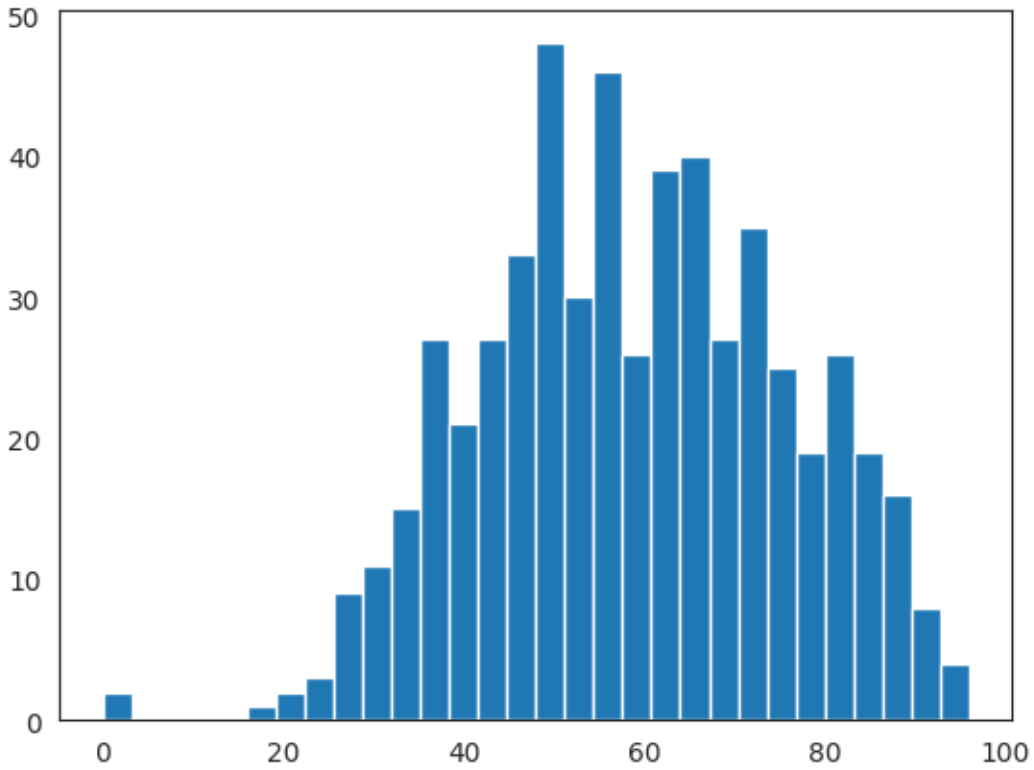
[48]:
```
m2=sns.distplot(movies.CriticRating ,bins=15)
```



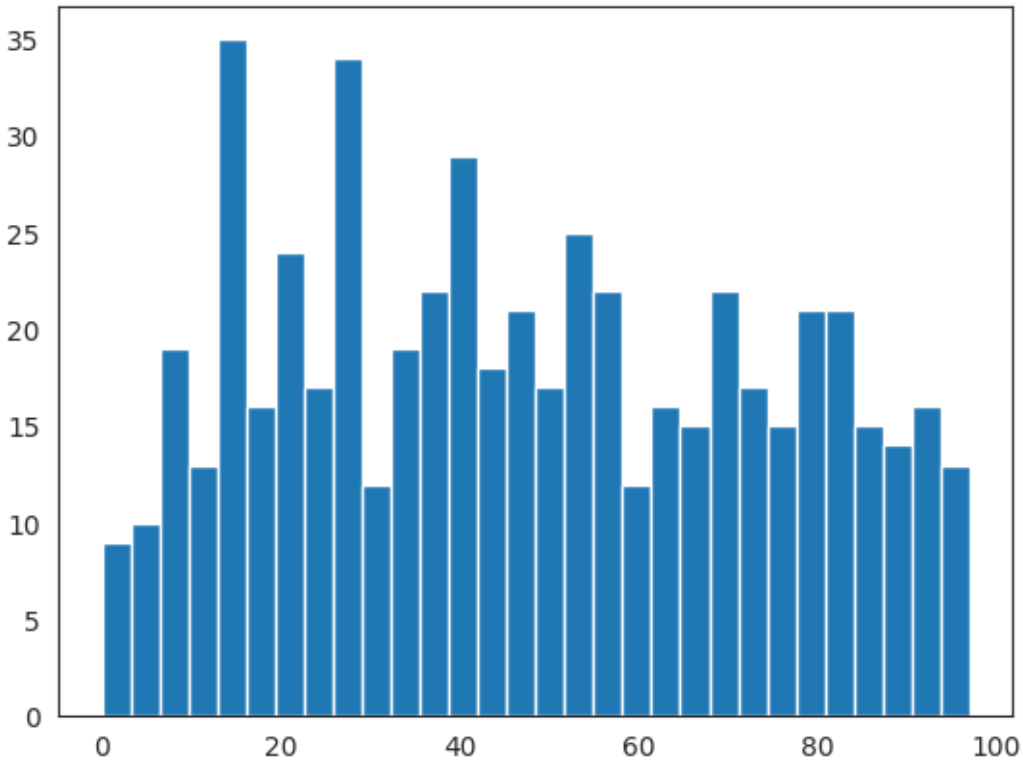
[50]:
```
import matplotlib.pyplot as plt
n1=plt.hist(movies.CriticRating,bins=15)
```



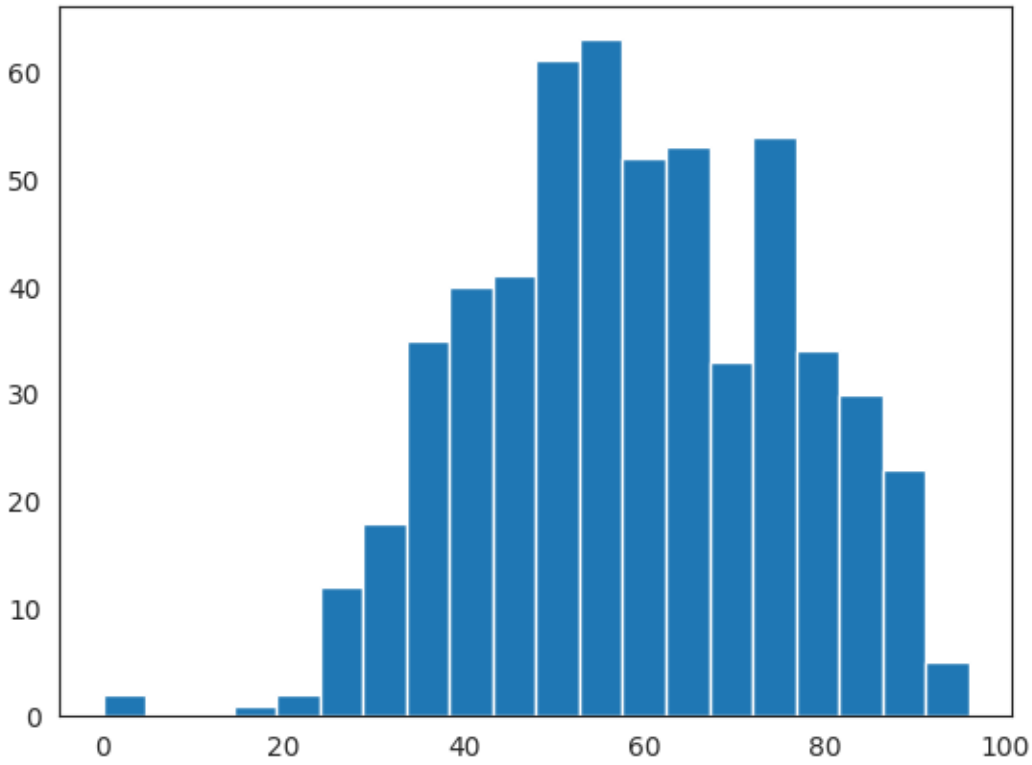
[51]:
```
sns.set_style('white')
n1=plt.hist(movies.AudienceRating,bins=30)
```



[52]:
```
n1=plt.hist(movies.CriticRating,bins=30)
```

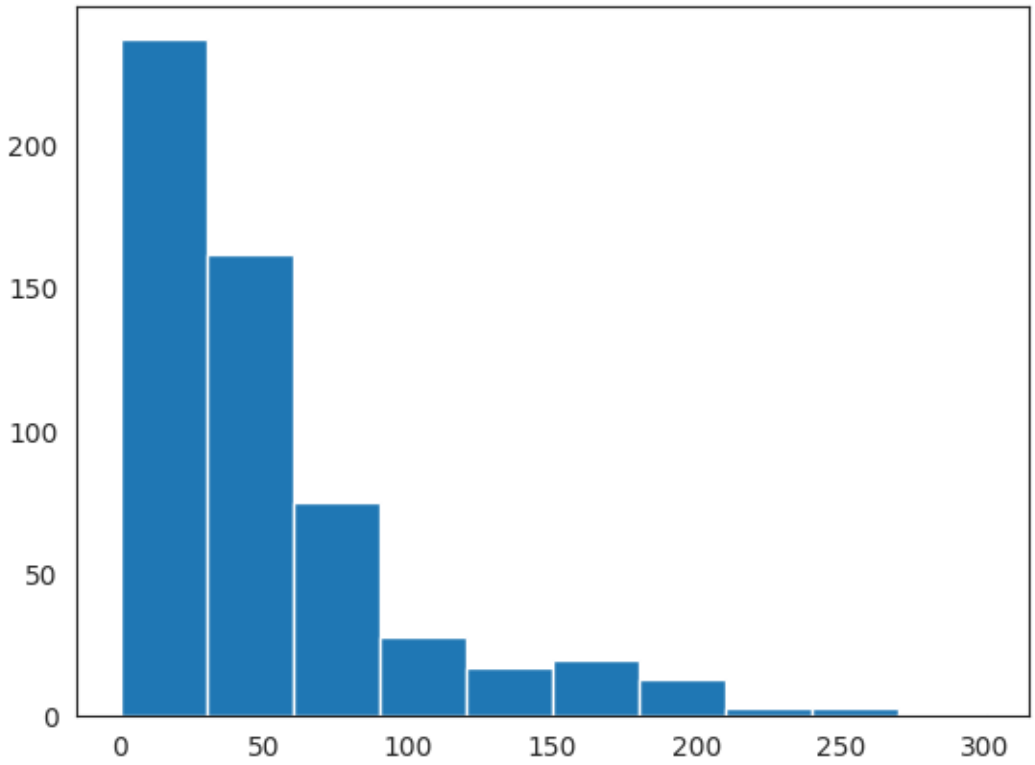


[54]:
```
n1=plt.hist(movies.AudienceRating,bins=20)
```

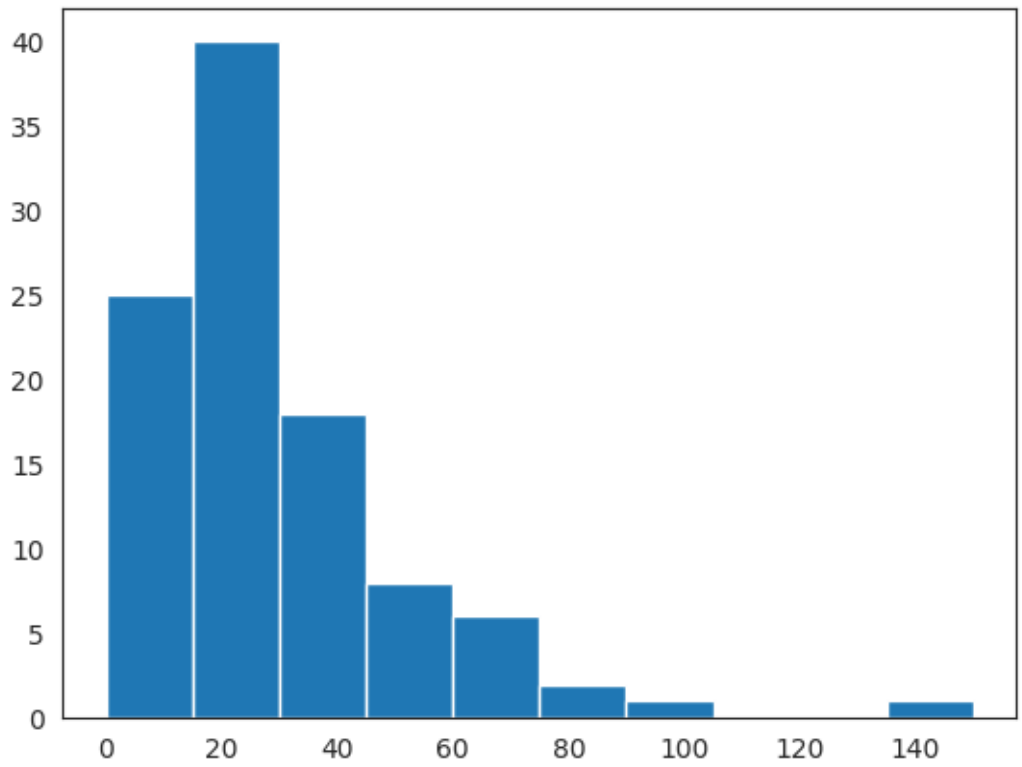


2 Creating stacked histograms & this is bit tough to understand

[56]:
```
plt.hist(movies.BudgetMillions)
plt.show()
```



[58]:
```
plt.hist(movies[movies.Genre == 'Drama'].BudgetMillions)
plt.show()
```



[59]:
```
movies.head()
```

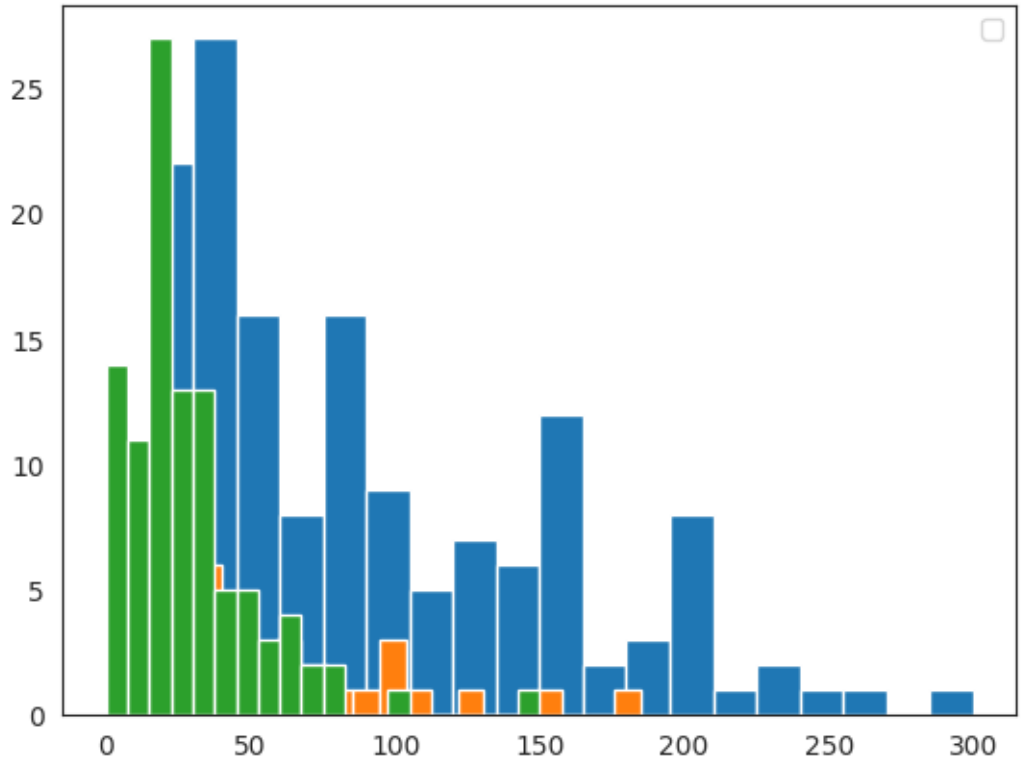
[59]:

	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

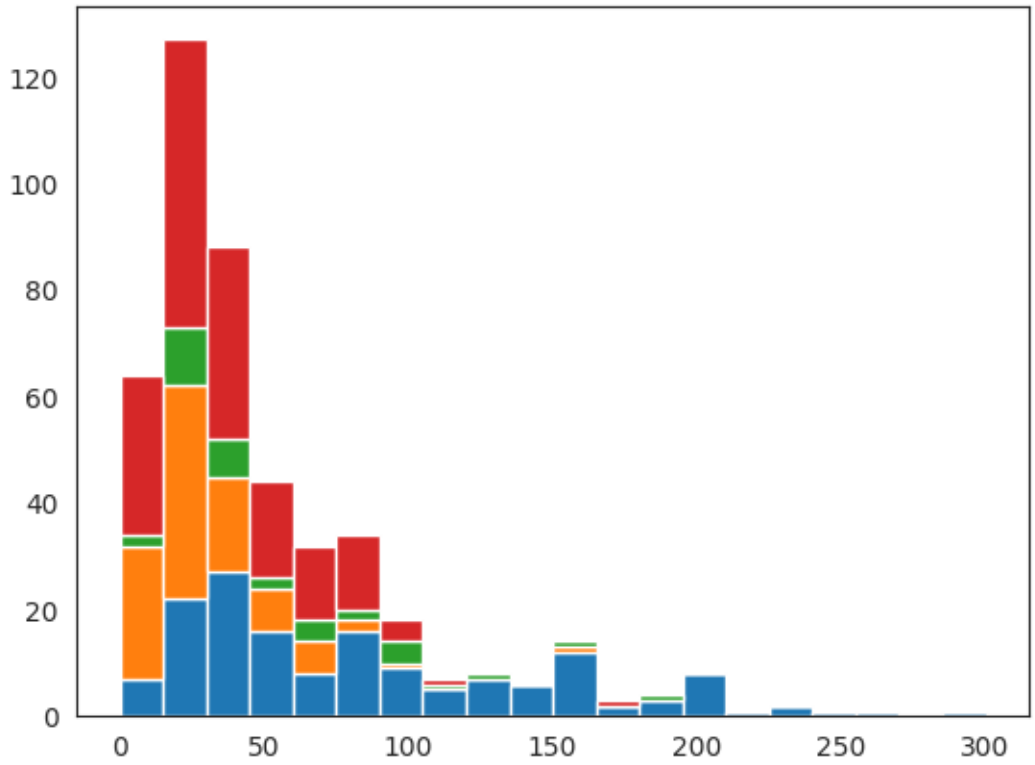
[70]:
```
# Below plots are stacked histogram becuase overlaped

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()
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note
that artists whose label start with an underscore are ignored when legend() is
called with no argument.
```



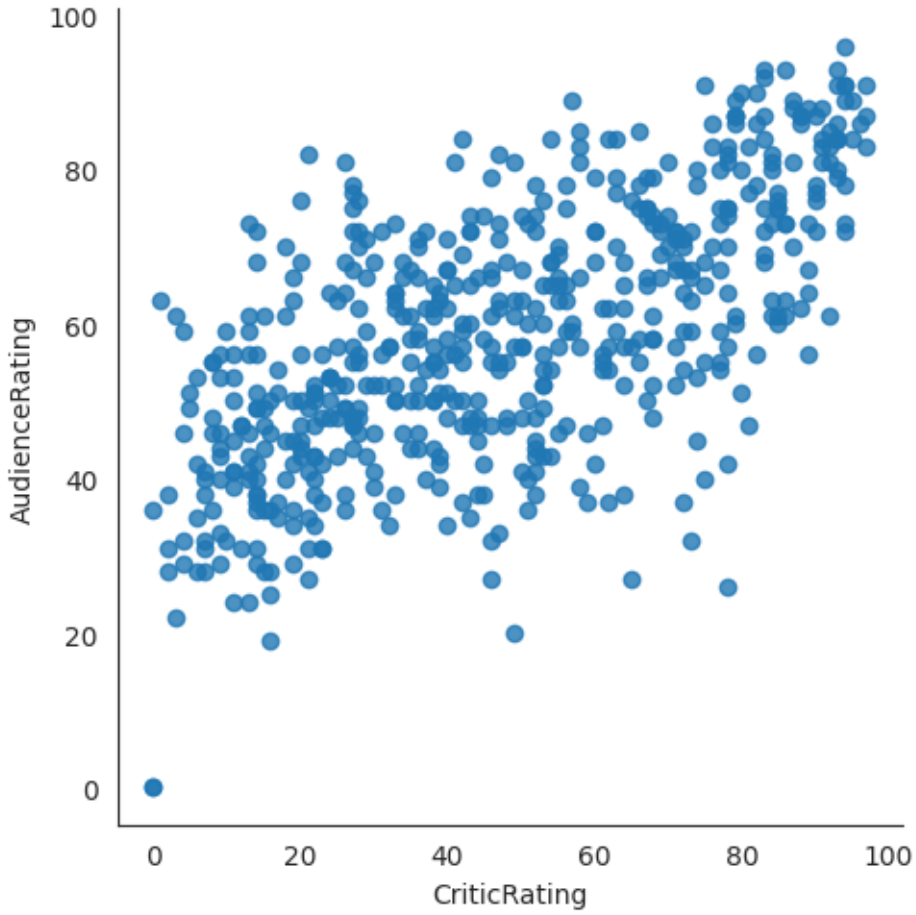
[71]:
```
plt.hist([movies[movies.Genre == 'Action'].BudgetMillions,\
          movies[movies.Genre == 'Drama'].BudgetMillions, \
          movies[movies.Genre == 'Thriller'].BudgetMillions, \
          movies[movies.Genre == 'Comedy'].BudgetMillions],
         bins = 20, stacked = True)
plt.show()
```



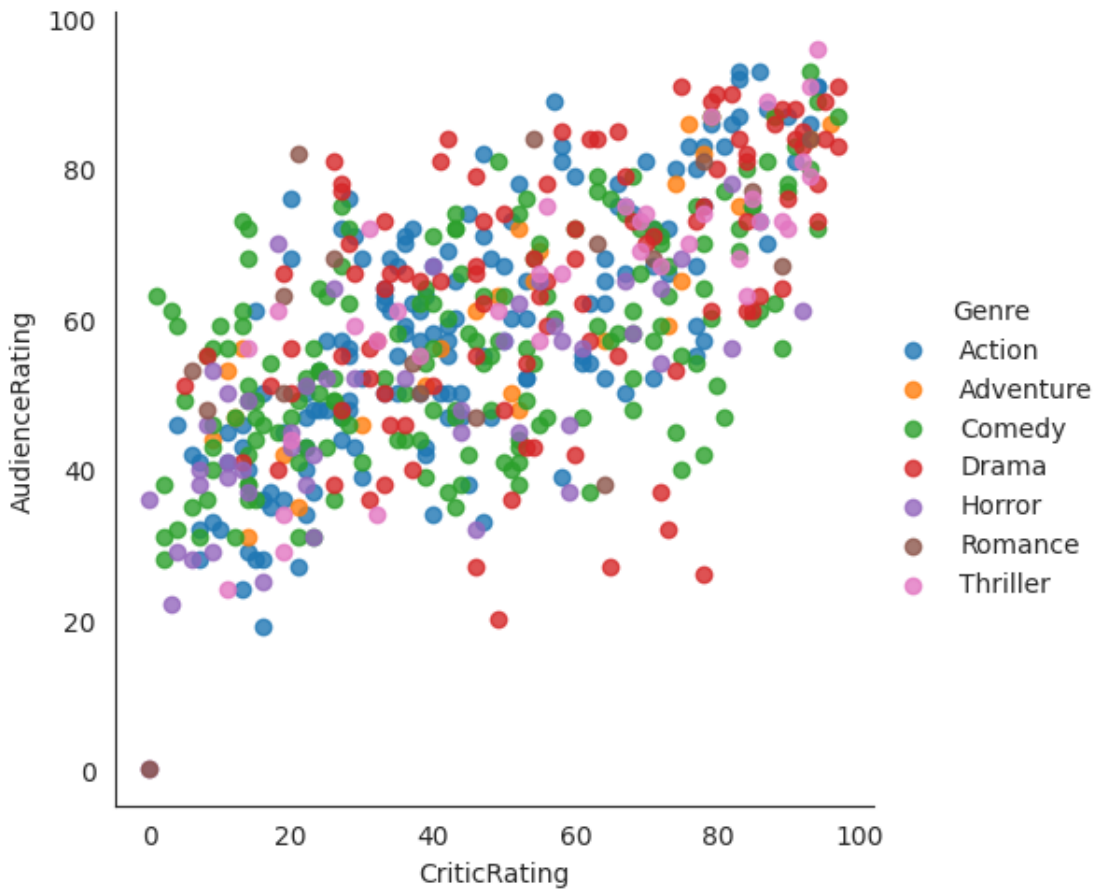
[73]:
```
# if you have 100 categories you cannot copy & paste all the things
for gen in movies.Genre.cat.categories:
  print(gen)
```

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

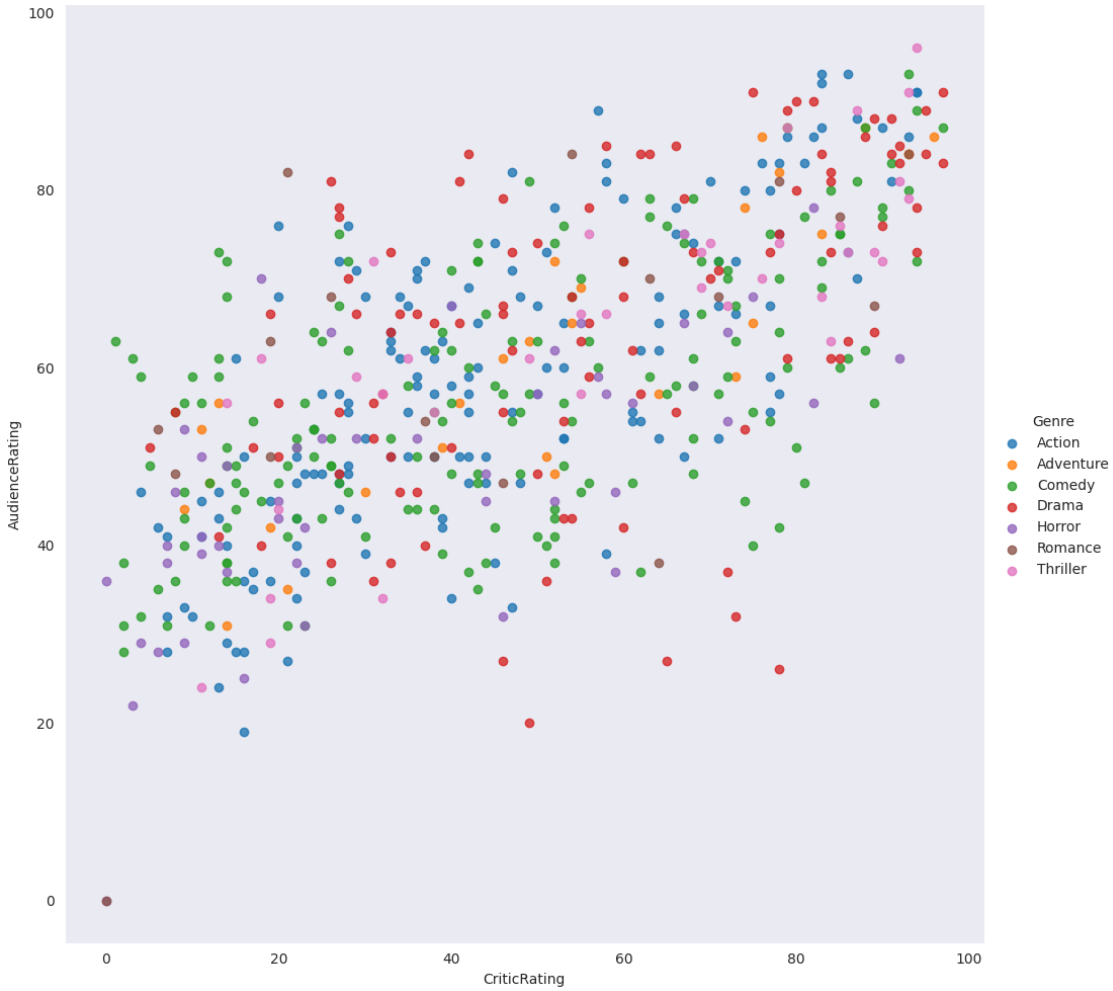
[74]:
```
vist1=sns.lmplot(data=movies,x='CriticRating',y='AudienceRating',fit_reg=False)
```



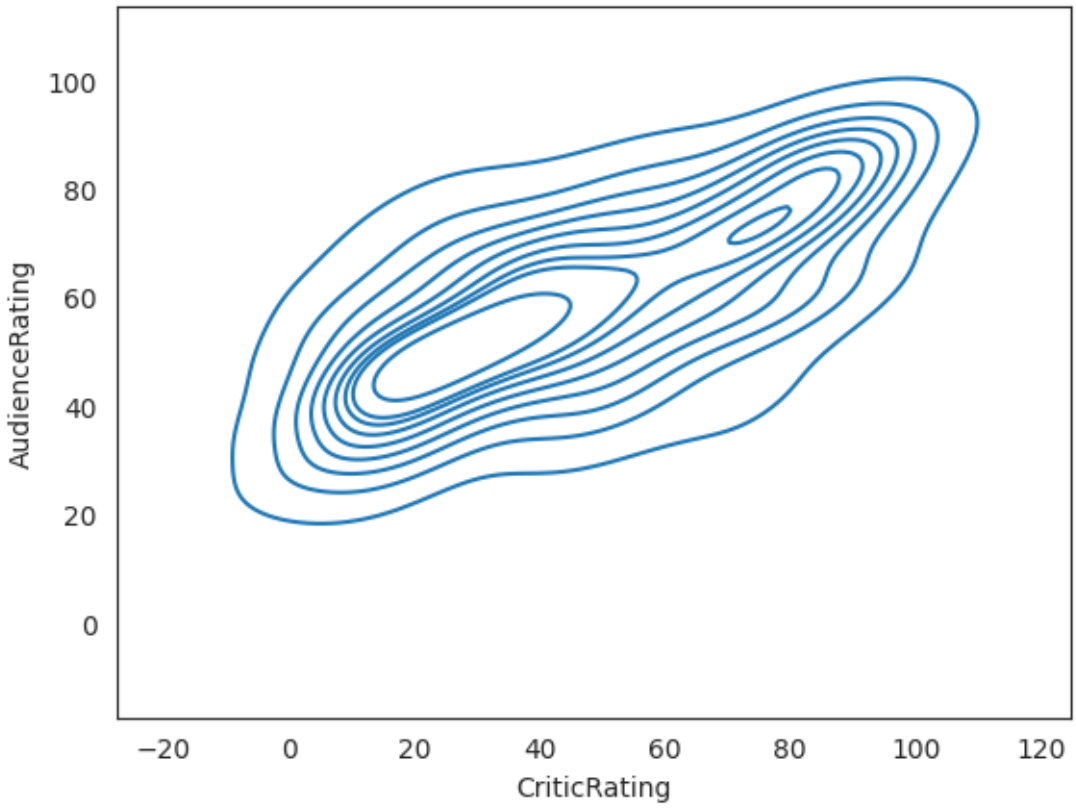
[82]:
```
vis1 = sns.lmplot(data=movies, x='CriticRating', y='AudienceRating',\
                  fit_reg=False, hue = 'Genre')
```



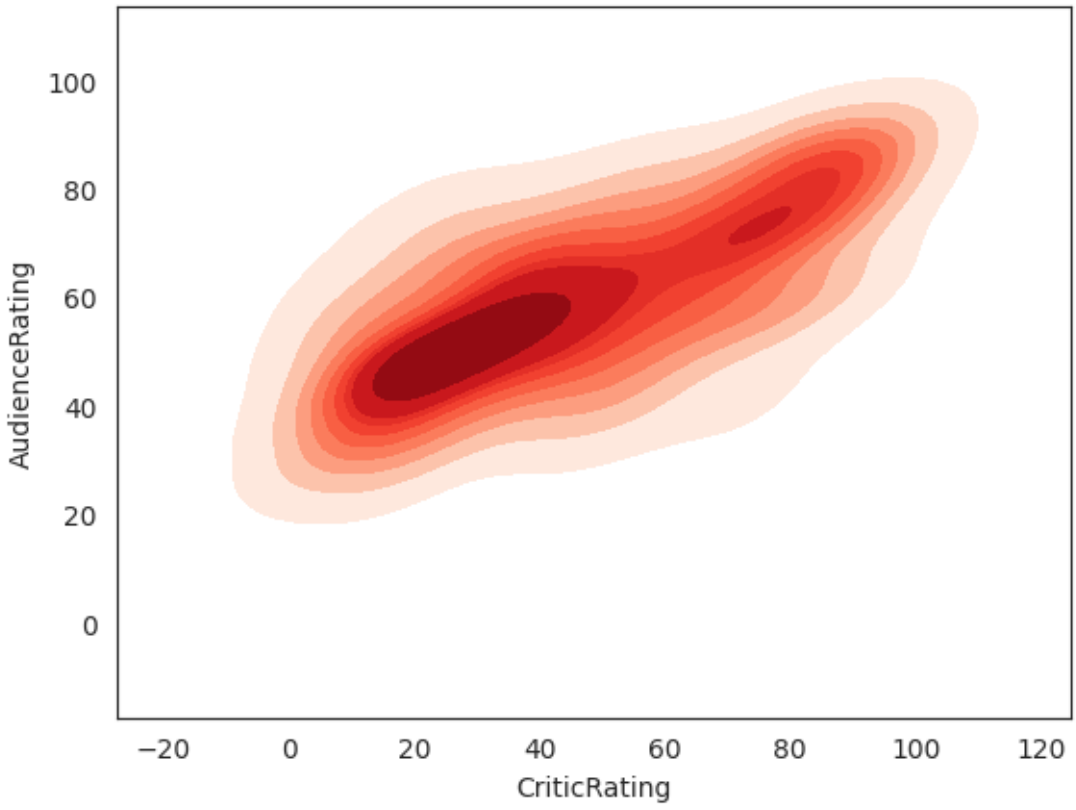
[90]:
```
# Fixing the lmplot code
vis1 = sns.lmplot(data=movies, x='CriticRating', y='AudienceRating',
                  fit_reg=False, hue='Genre', height=10, aspect=1)

# Display the plot
plt.show()
```



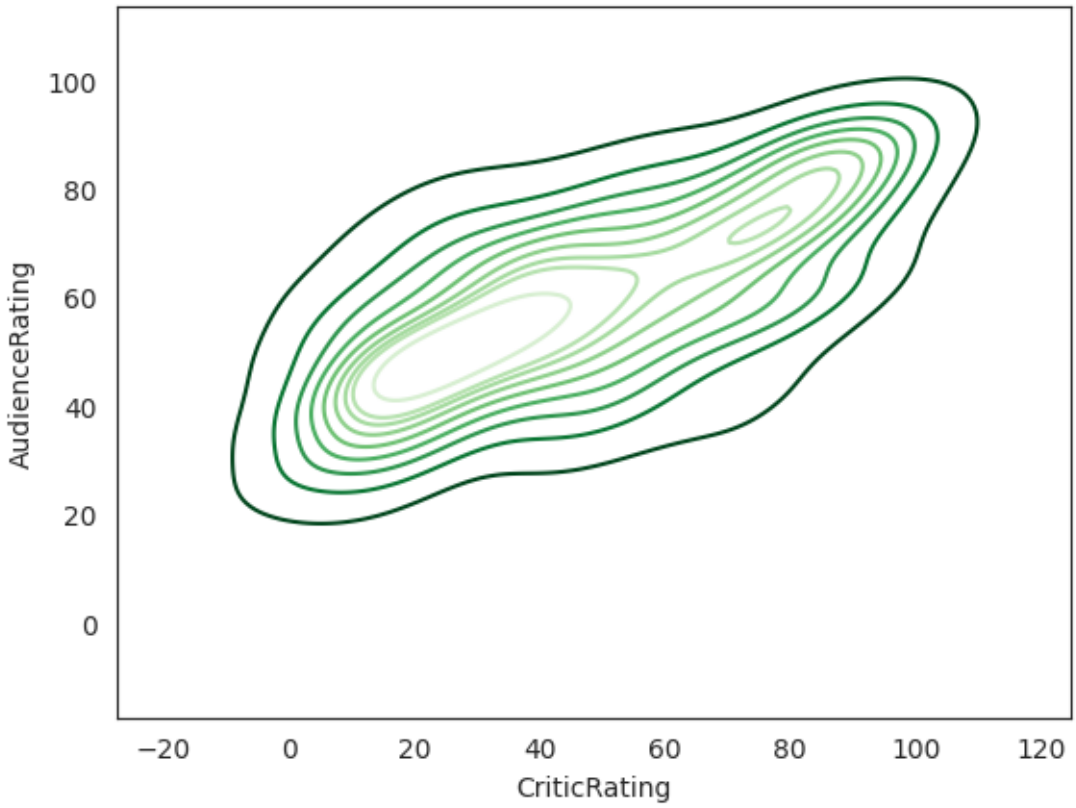
[93]:
```
sns.set_style('white')
k1 = sns.kdeplot(x=movies.CriticRating, y=movies.AudienceRating)

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



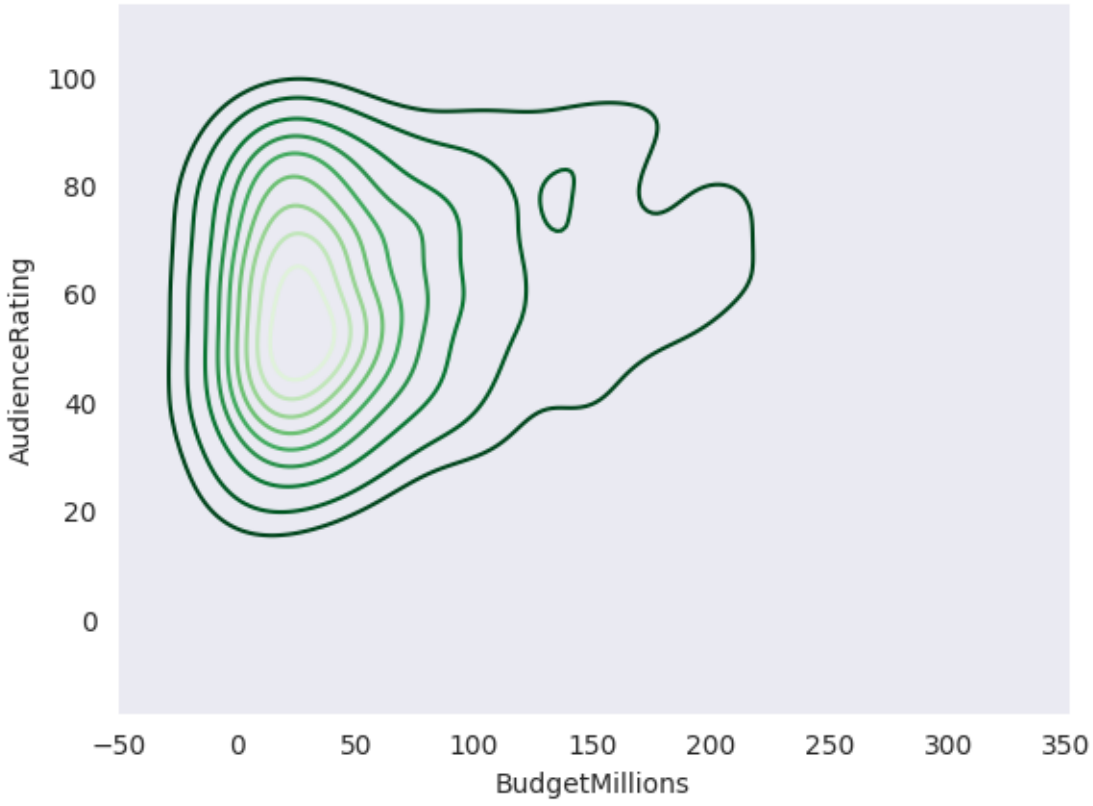
[95]:
```
k1 = sns.kdeplot(x=movies.CriticRating,y=movies.AudienceRating,shade =␣
↪True,shade_lowest=False,cmap='Reds')
```



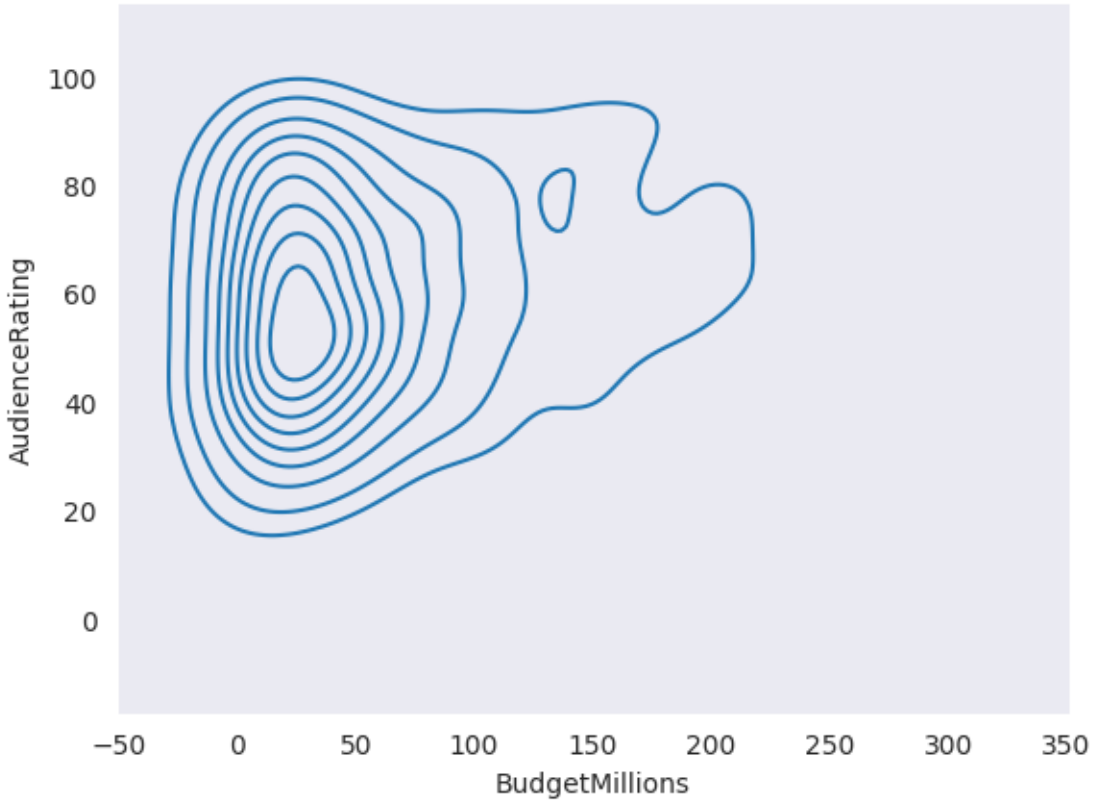
[98]:
```
k1 = sns.kdeplot(x=movies.CriticRating,y=movies.
↪AudienceRating,shade_lowest=False,cmap='Greens_r')
```



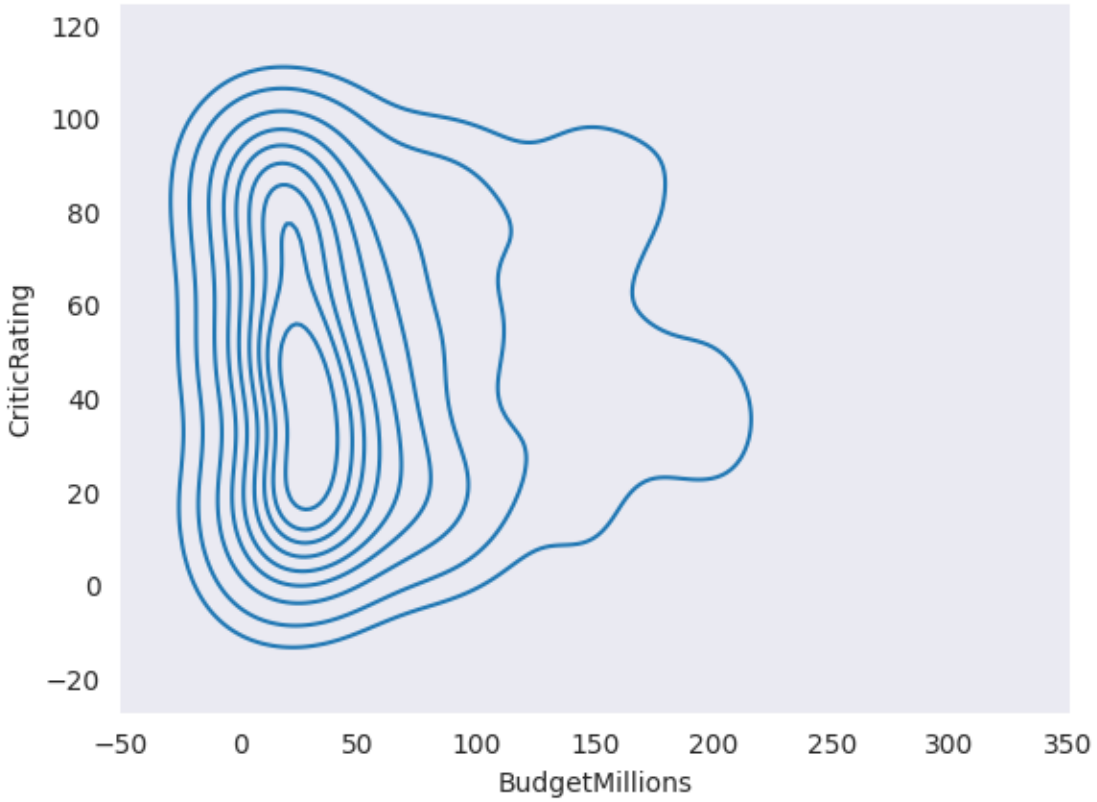
[100]:
```
sns.set_style('dark')
k1 = sns.kdeplot(x=movies.BudgetMillions,y=movies.
↪AudienceRating,shade_lowest=False,cmap='Greens_r')
```



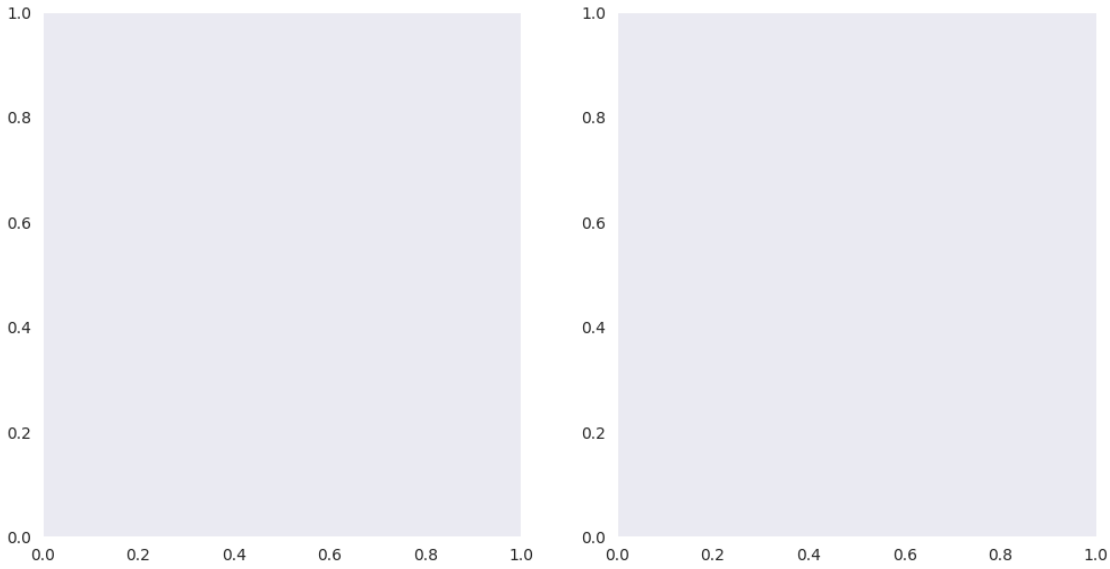
[102]:
```
sns.set_style('dark')
k1 = sns.kdeplot(x=movies.BudgetMillions,y=movies.AudienceRating)
```



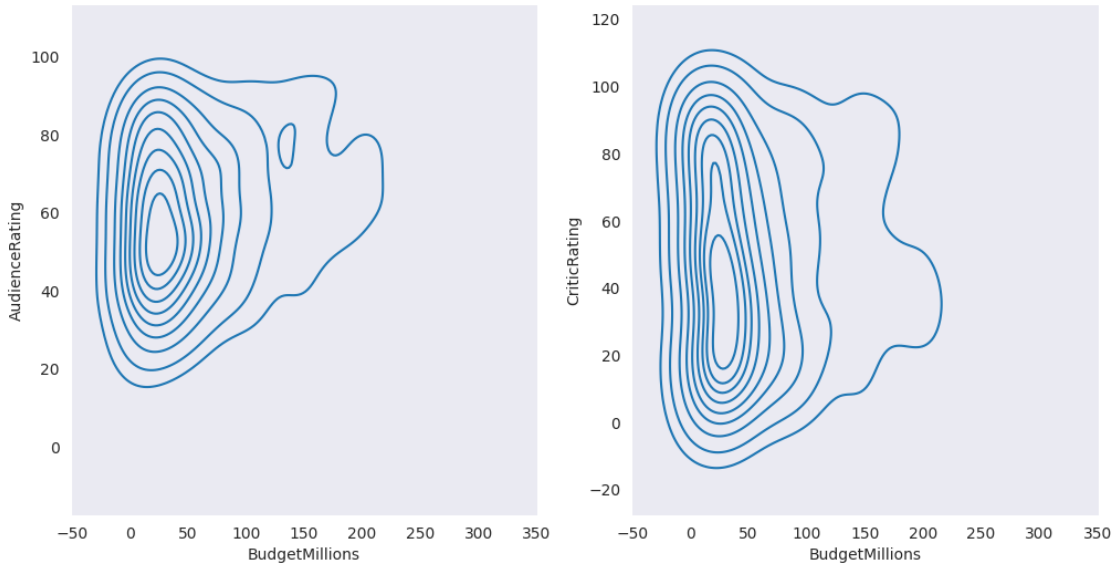
[103]:
```
k2 = sns.kdeplot(x=movies.BudgetMillions,y=movies.CriticRating)
```



[104]:
```
#subplots

f, ax = plt.subplots(1,2, figsize =(12,6))
#f, ax = plt.subplots(3,3, figsize =(12,6))
```



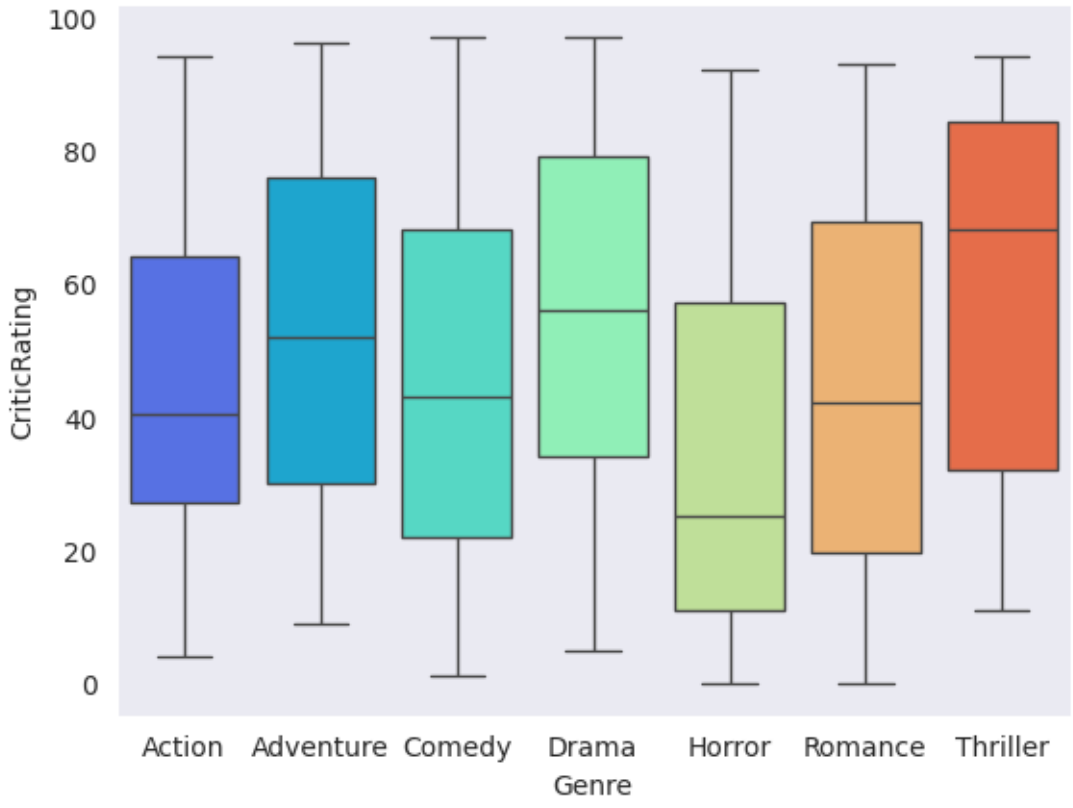
[106]:
```
f, axes = plt.subplots(1,2, figsize =(12,6))

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



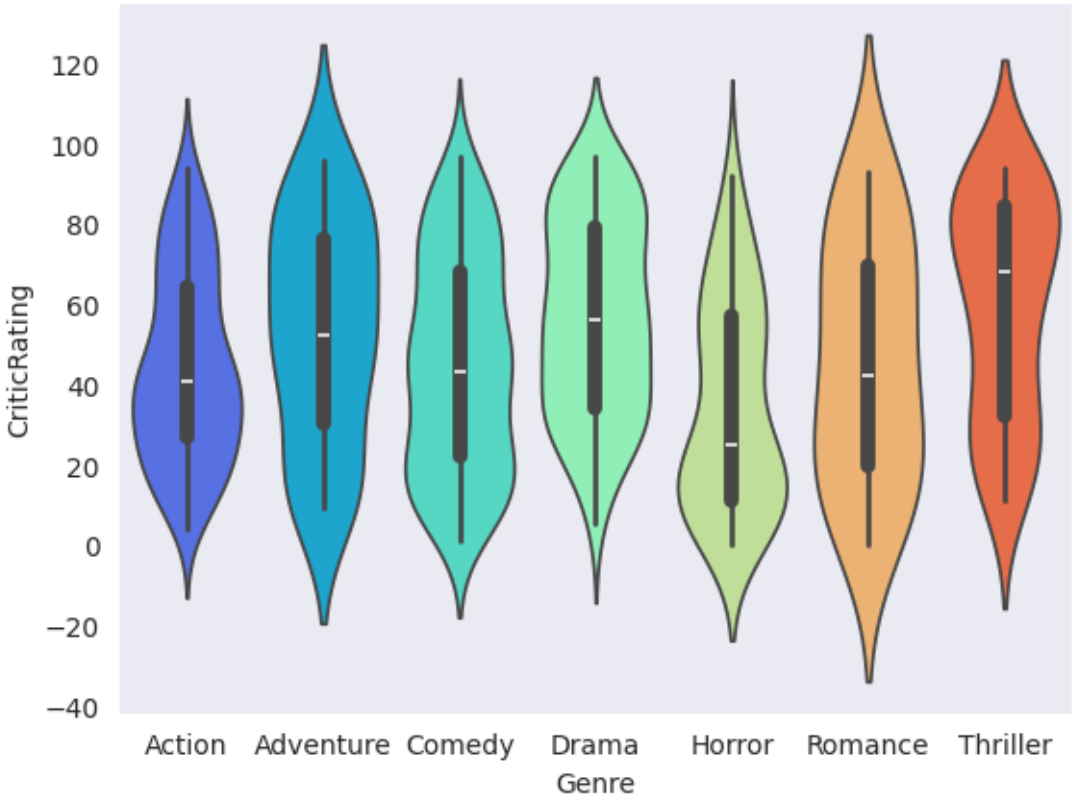
[107]:
```
axes
```

[107]:
```
array([<Axes: xlabel='BudgetMillions', ylabel='AudienceRating'>,
       <Axes: xlabel='BudgetMillions', ylabel='CriticRating'>],
      dtype=object)
```

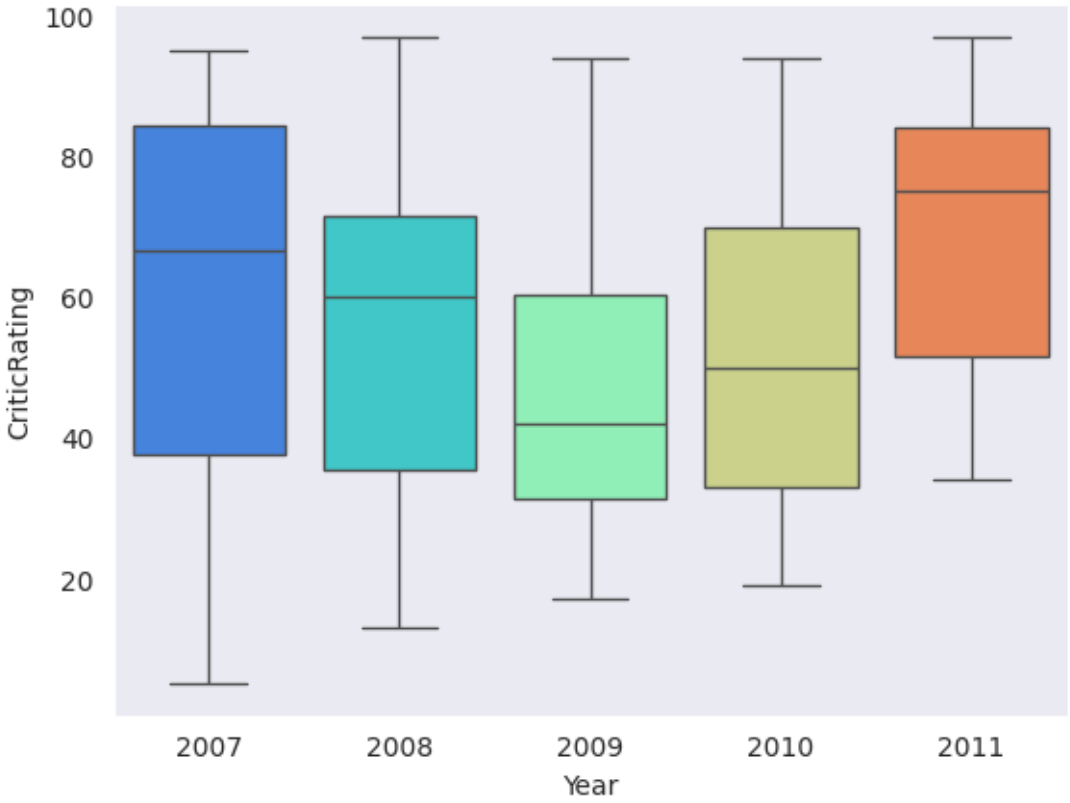
[111]:
```
#Box plots -

w = sns.boxplot(data=movies, x='Genre', y =␣
 ↪'CriticRating',hue='Genre',palette='rainbow')
```



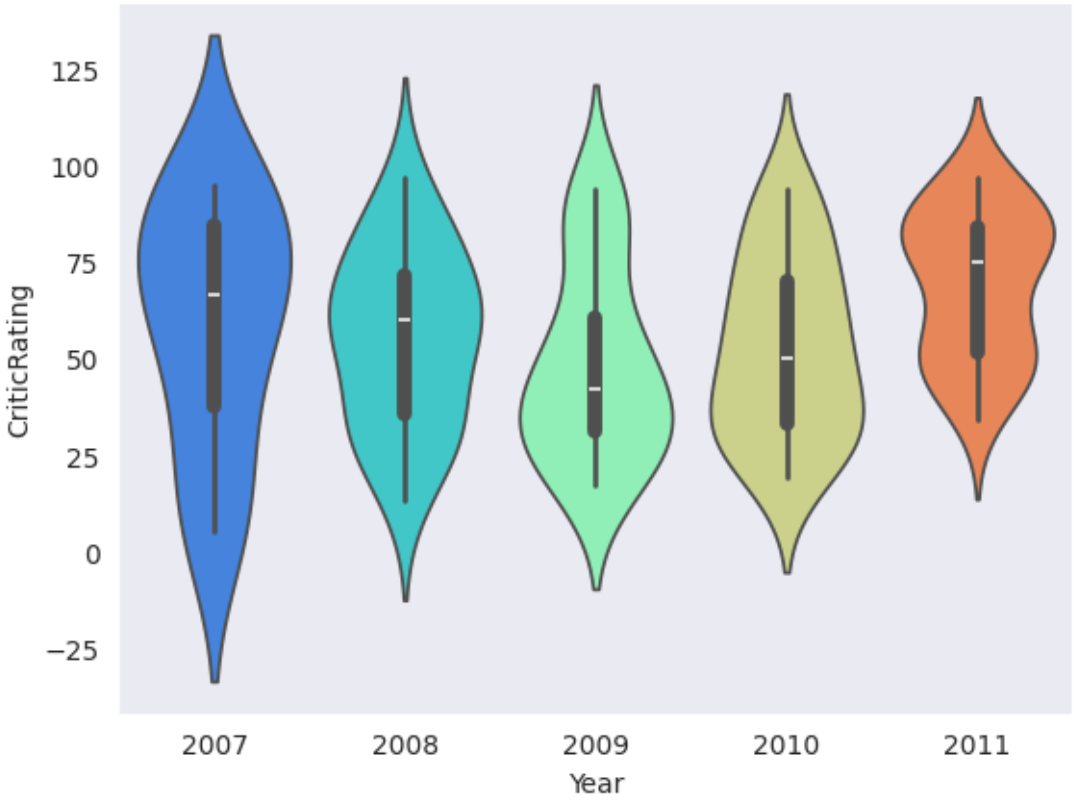
[112]:
```
#violin plot

z = sns.violinplot(data=movies, x='Genre', y = 'CriticRating',␣
↪palette='rainbow')
```



[114]:
```
w1 = sns.boxplot(data=movies[movies.Genre == 'Drama'], x='Year', y =␣
↪'CriticRating',palette='rainbow')
```

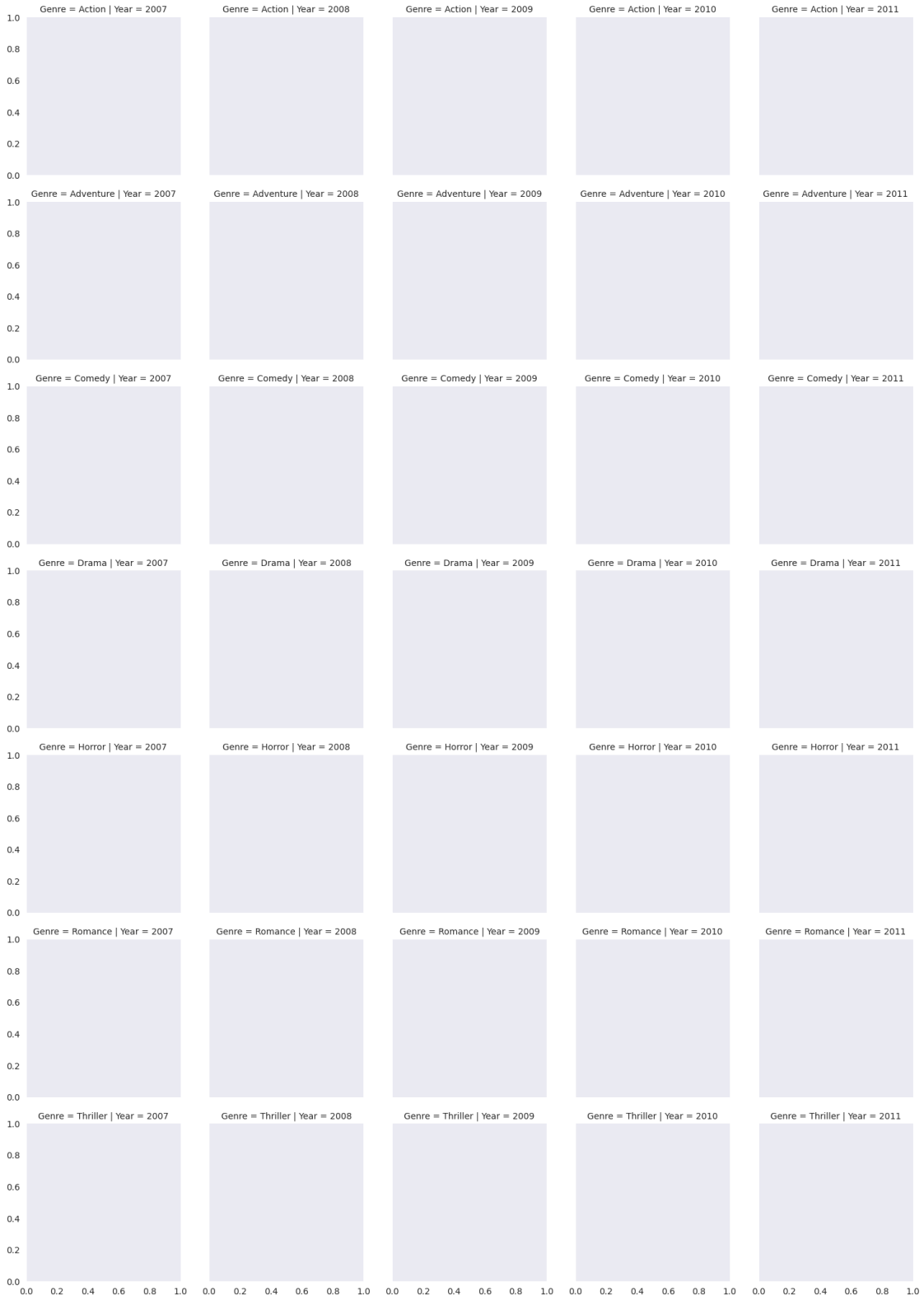


[115]:
```
z = sns.violinplot(data=movies[movies.Genre == 'Drama'], x='Year', y =␣
↪'CriticRating',palette='rainbow')
```



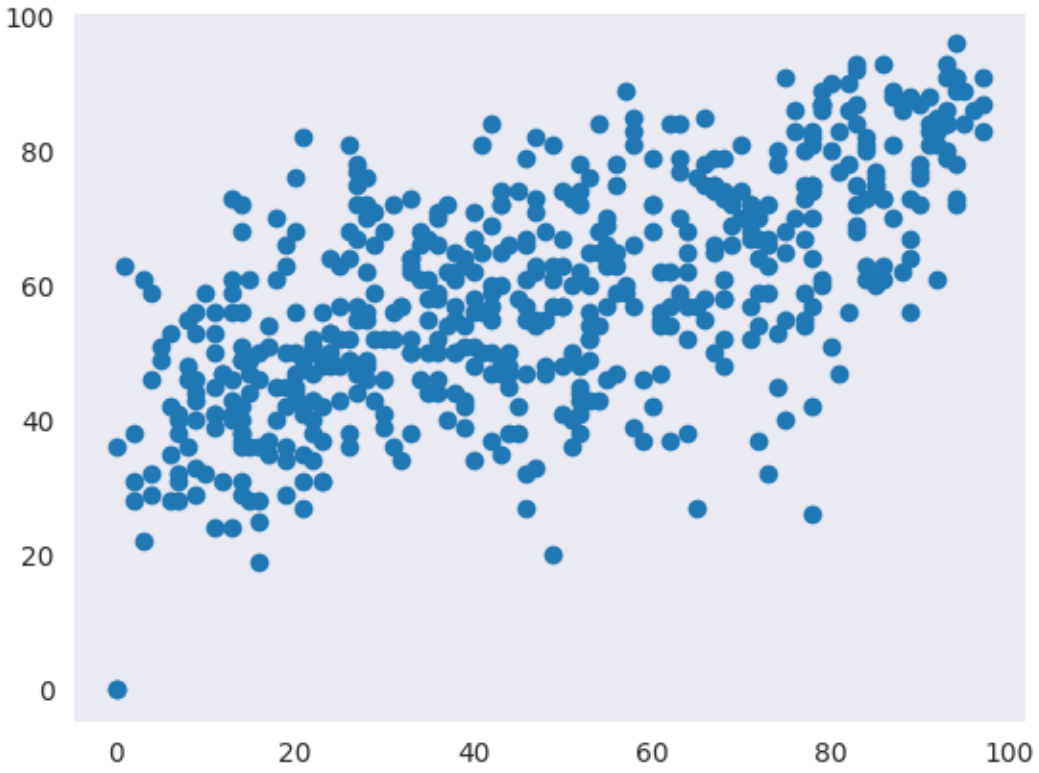
[116]:
```
# Createing a Facet grid
```

[117]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue = 'Genre') #kind of␣
↪subplots
```

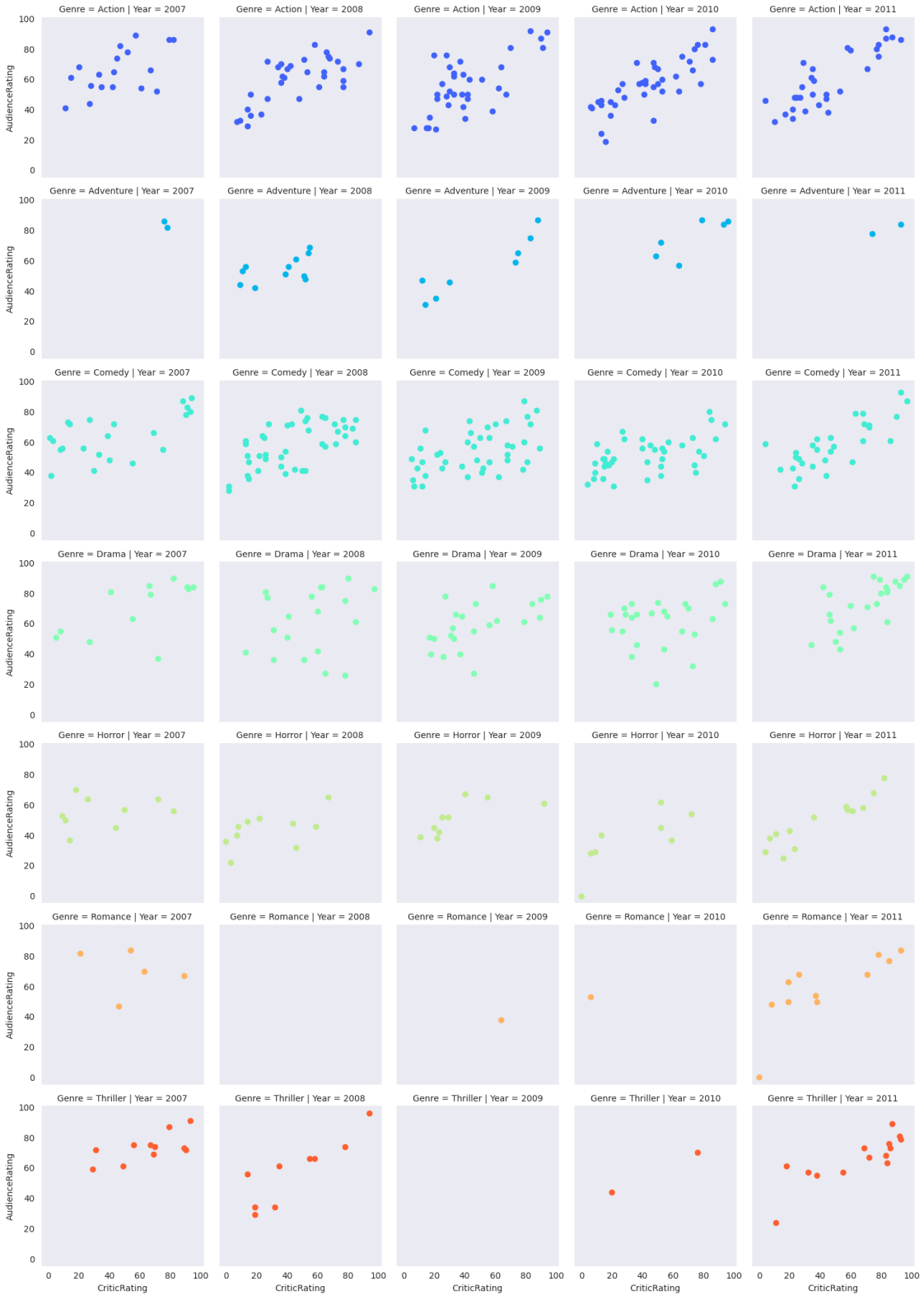

Genre = Action | Year = 2007
Genre = Action | Year = 2008
Genre = Action | Year = 2009
Genre = Action | Year = 2010
Genre = Action | Year = 2011
Genre = Adventure | Year = 2007
Genre = Adventure | Year = 2008
Genre = Adventure | Year = 2009
Genre = Adventure | Year = 2010
Genre = Adventure | Year = 2011
Genre = Comedy | Year = 2007
Genre = Comedy | Year = 2008
Genre = Comedy | Year = 2009
Genre = Comedy | Year = 2010
Genre = Comedy | Year = 2011
Genre = Drama | Year = 2007
Genre = Drama | Year = 2008
Genre = Drama | Year = 2009
Genre = Drama | Year = 2010
Genre = Drama | Year = 2011
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
0.0
0.2
0.4
0.6
0.8
1.0

[118]:
```
plt.scatter(movies.CriticRating,movies.AudienceRating)
```

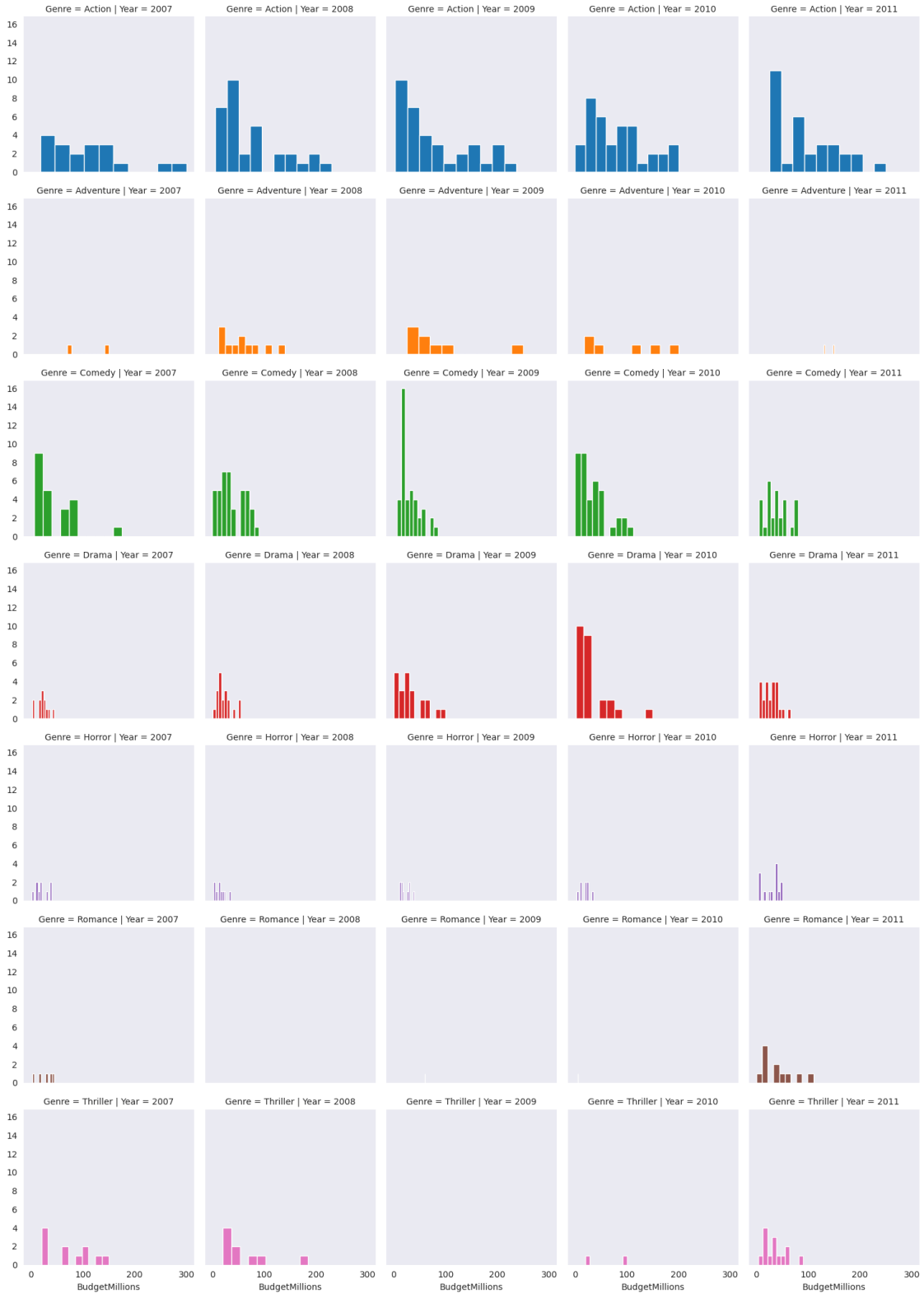
[118]:
```
<matplotlib.collections.PathCollection at 0x790bd2709840>
```



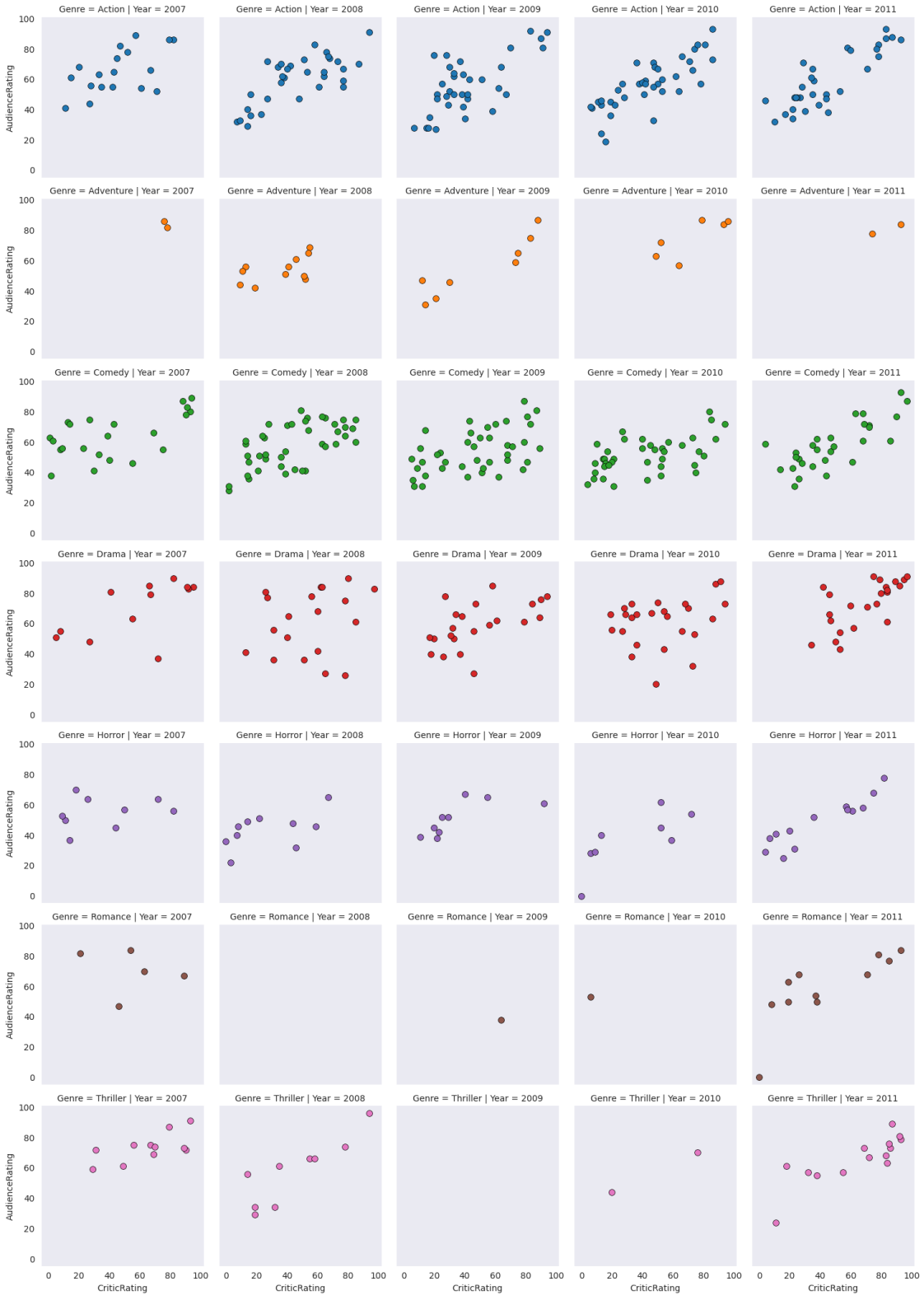
[121]:
```
g =sns.FacetGrid (movies, row = 'Genre', col = 'Year', hue =␣
↪'Genre',palette='rainbow')
g = g.map(plt.scatter, 'CriticRating', 'AudienceRating' ) #scatterplots are␣
↪mapped in facetgrid
```


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
AudienceRating
CriticRating
0
20
40
60
80
100

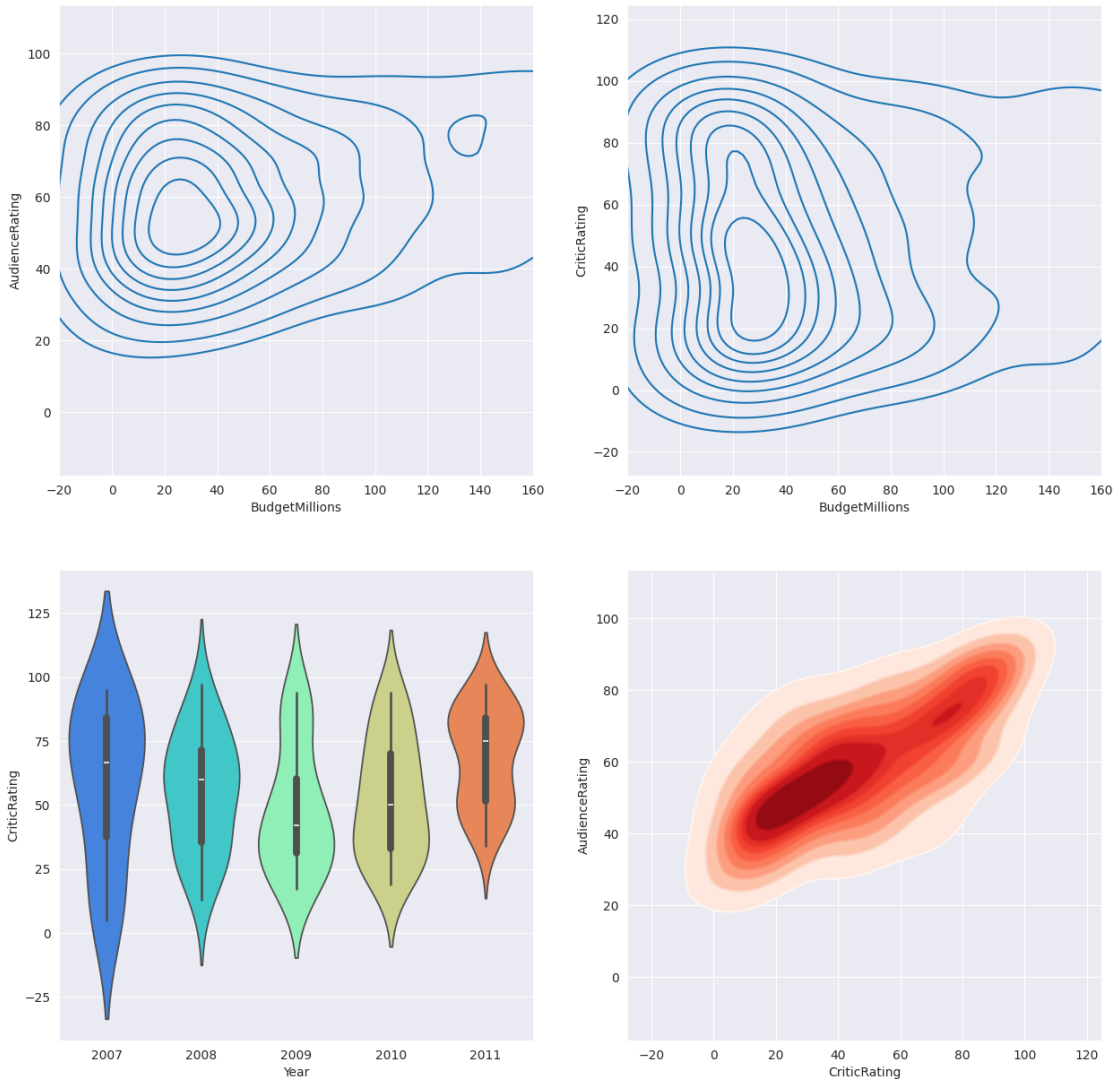
[122]:
```
# you can populated any type of chat.

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


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

[123]:
```
#
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 ) #scatterplots␣
 ↪are mapped in facetgrid
```

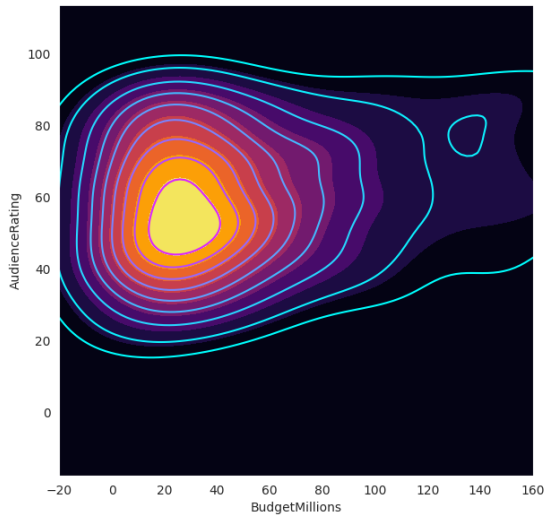

Genre = Action | Year = 2007
Genre = Action | Year = 2008
Genre = Action | Year = 2009
Genre = Action | Year = 2010
Genre = Action | Year = 2011
Genre = Adventure | Year = 2007
Genre = Adventure | Year = 2008
Genre = Adventure | Year = 2009
Genre = Adventure | Year = 2010
Genre = Adventure | Year = 2011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

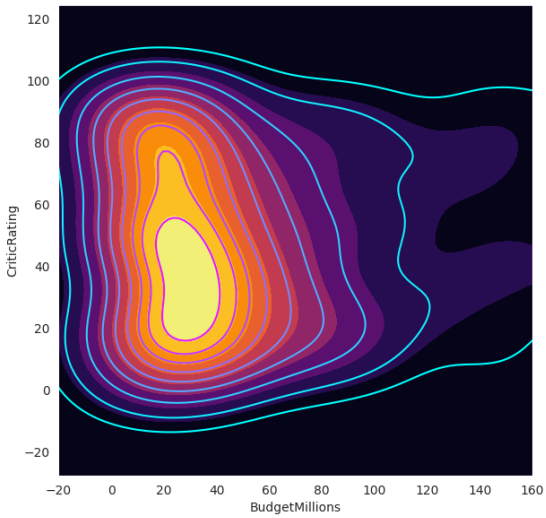
[127]:
```
# python is not vectorize programming language
# Building dashboards (dashboard - combination of chats)

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

k1 = sns.kdeplot(x=movies.BudgetMillions,y=movies.AudienceRating,ax=axes[0,0])
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 =␣
 ↪'CriticRating', ax=axes[1,0],palette='rainbow')

k4 = sns.kdeplot(x=movies.CriticRating,y=movies.AudienceRating,shade =␣
 ↪True,shade_lowest=False,cmap='Reds',ax=axes[1,1])

k4b = sns.kdeplot(x=movies.CriticRating,y= movies.AudienceRating,cmap='Reds',ax␣
 ↪= axes[1,1])

plt.show()
```

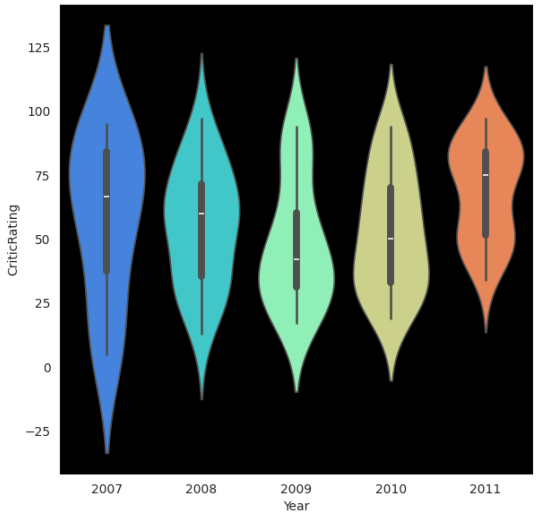


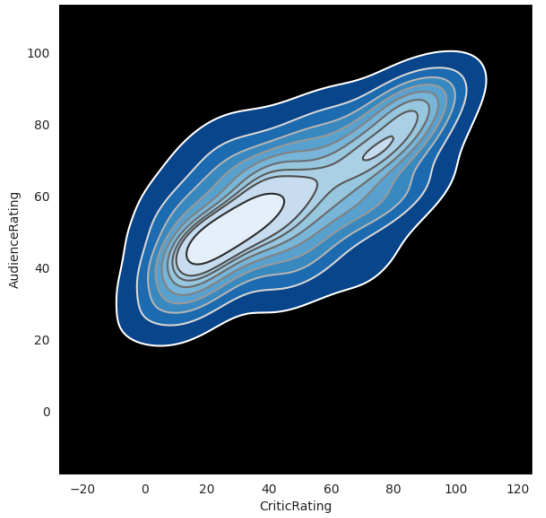
[131]:
```
# How can you style your dashboard  using different color map

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

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

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

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

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


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

plt.show()
```


AudienceRating
BudgetMillions


CriticRating
BudgetMillions


CriticRating
Year


AudienceRating
CriticRating

[ ]: