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

In [2]:
```
movies = pd.read_csv(r"C:\Users\Jan Saida\Downloads\Movie-Rating.csv")
```

In [3]:
```
movies
```

Out[3]:

	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 [4]:
```
movies.head()
```

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

In [5]:
```
movies.tail()
```

Out[5]:

	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 [6]:
```
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 [7]:
```
type('movies')
```

Out[7]:
```
str
```

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

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

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

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

```
In [10]: id(movies)
```

```
Out[10]: 2895782033504
```

```
In [11]: len(movies)
```

```
Out[11]: 559
```

```
In [12]: movies.columns
```

```
Out[12]: Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
                'Budget (million $)', 'Year of release'],
               dtype='object')
```

```
In [13]: movies.columns=['Film','Genre','CriticRating','AudienceRating','BudgetMillions','Year']
```

```
In [14]: movies.columns
```

```
Out[14]: Index(['Film', 'Genre', 'CriticRating', 'AudienceRating', 'BudgetMillions',
                'Year'],
               dtype='object')
```

```
In [15]: movies.head(1)
```

Out[15]:

	Film	Genre	CriticRating	AudienceRating	BudgetMillions	Year
0	(500) Days of Summer	Comedy	87	81	8	2009

```
In [16]: movies.head()
```

Out[16]:

	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

```
In [17]: movies.shape
```

```
Out[17]: (559, 6)
```

```
In [18]: movies.describe()
```

Out[18]:

	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

```
In [19]: movies.transpose()
```

Out[19]:

	0	1	2	3	4	5	6	7	8	9	...	549	550
Film	(500) Days of Summer	10,000 B.C.	12 Rounds	127 Hours	17 Again	2012	27 Dresses	30 Days of Night	30 Minutes or Less	50/50	...	Yes Man	Yogi Bear
Genre	Comedy	Adventure	Action	Adventure	Comedy	Action	Comedy	Horror	Comedy	Comedy	...	Comedy	Comedy
CriticRating	87	9	30	93	55	39	40	50	43	93	...	43	14
AudienceRating	81	44	52	84	70	63	71	57	48	93	...	72	36
BudgetMillions	8	105	20	18	20	200	30	32	28	8	...	70	80
Year	2009	2008	2009	2010	2009	2009	2008	2007	2011	2011	...	2008	2010

6 rows × 559 columns

In [20]:
```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 559 entries, 0 to 558
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Film            559 non-null    object
 1   Genre           559 non-null    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 [21]:
```
movies.Film=movies.Film.astype('category')
```

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

Out[22]:
```
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 Re
volt', 'Zodiac', 'Zombieland ', 'Zookeeper']
```

In [23]:
```
movies.head()
```

Out[23]:

	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

In [24]:
```
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
```

In [25]:
```
movies.describe()
```

Out[25]:

	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

In [26]:
```
movies.Genre=movies.Genre.astype('category')
movies.Year=movies.Year.astype('category')
```

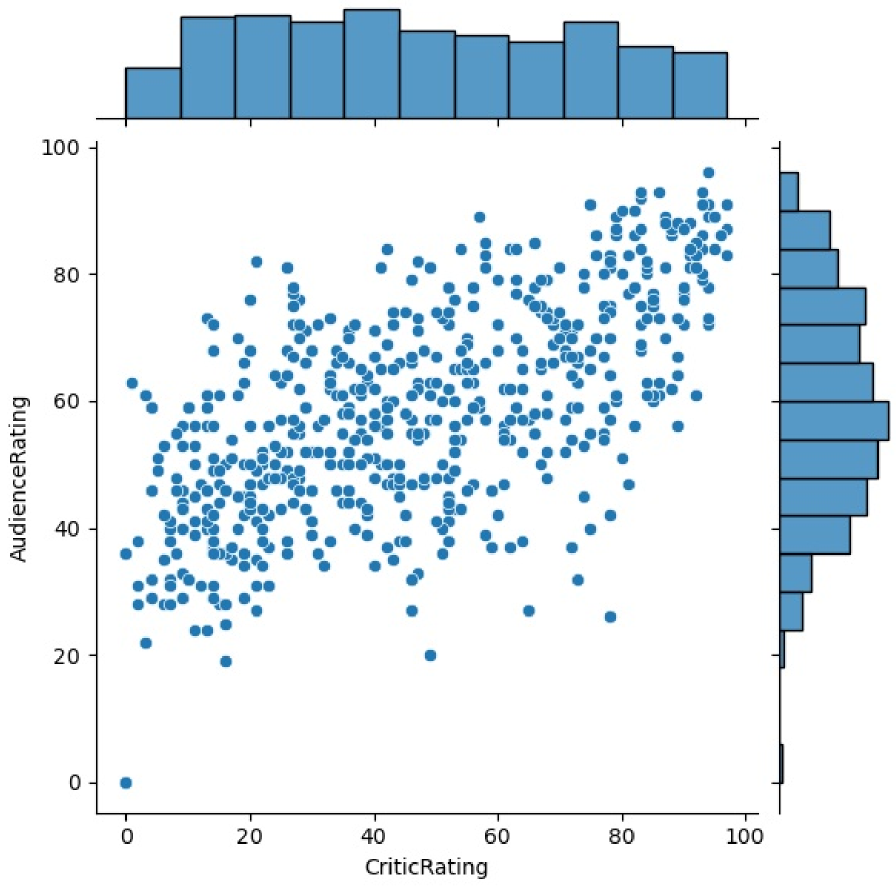
In [27]:
```
movies.describe()
```

Out[27]:

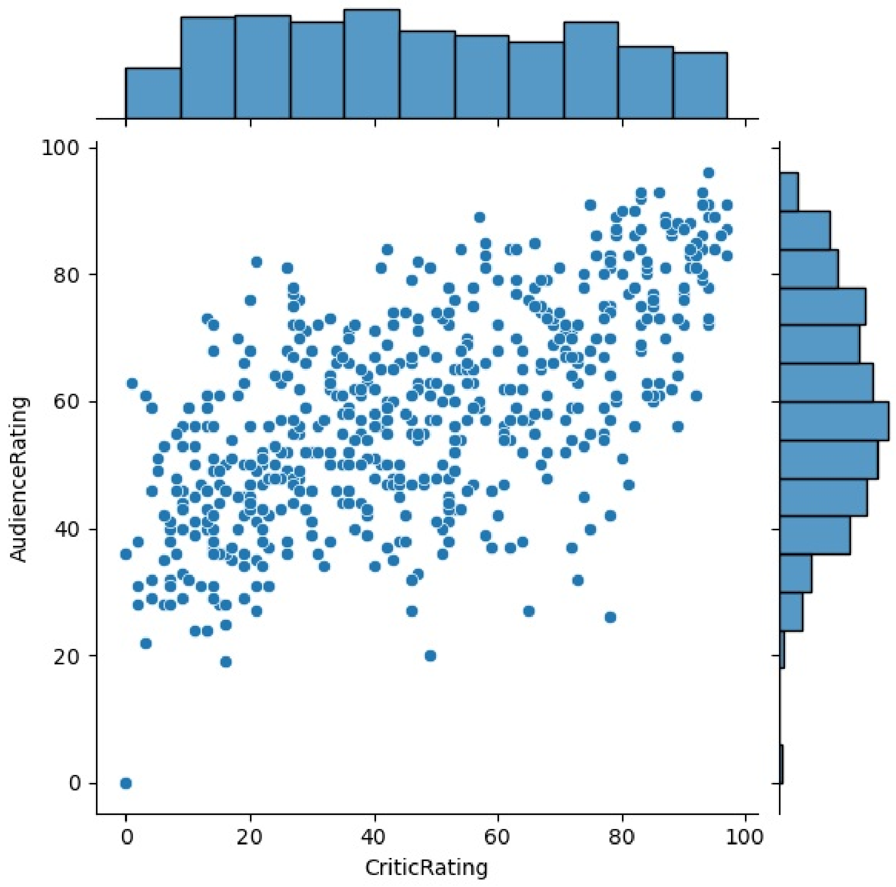
	CriticRating	AudienceRating	BudgetMillions
count	559.000000	559.000000	559.000000
mean	47.309481	58.744186	50.236136
std	26.413091	16.826887	48.731817
min	0.000000	0.000000	0.000000
25%	25.000000	47.000000	20.000000
50%	46.000000	58.000000	35.000000
75%	70.000000	72.000000	65.000000
max	97.000000	96.000000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

In [28]:
```
from matplotlib import pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

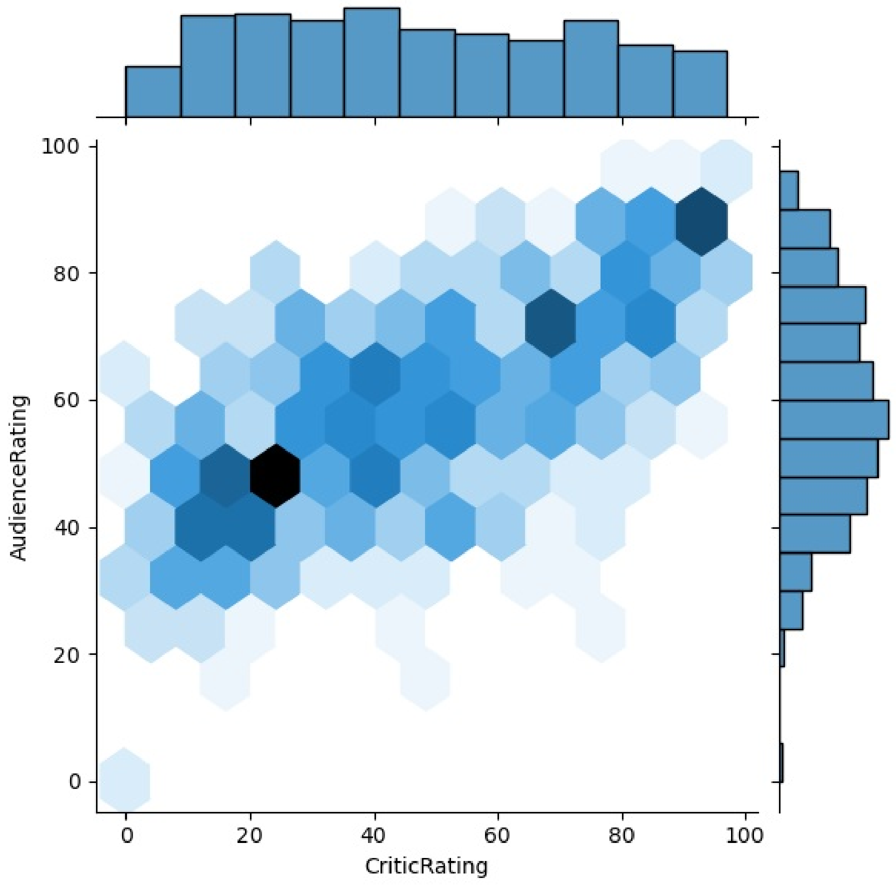
In [29]:
```
j=sns.jointplot(data=movies, x='CriticRating',y='AudienceRating')
```



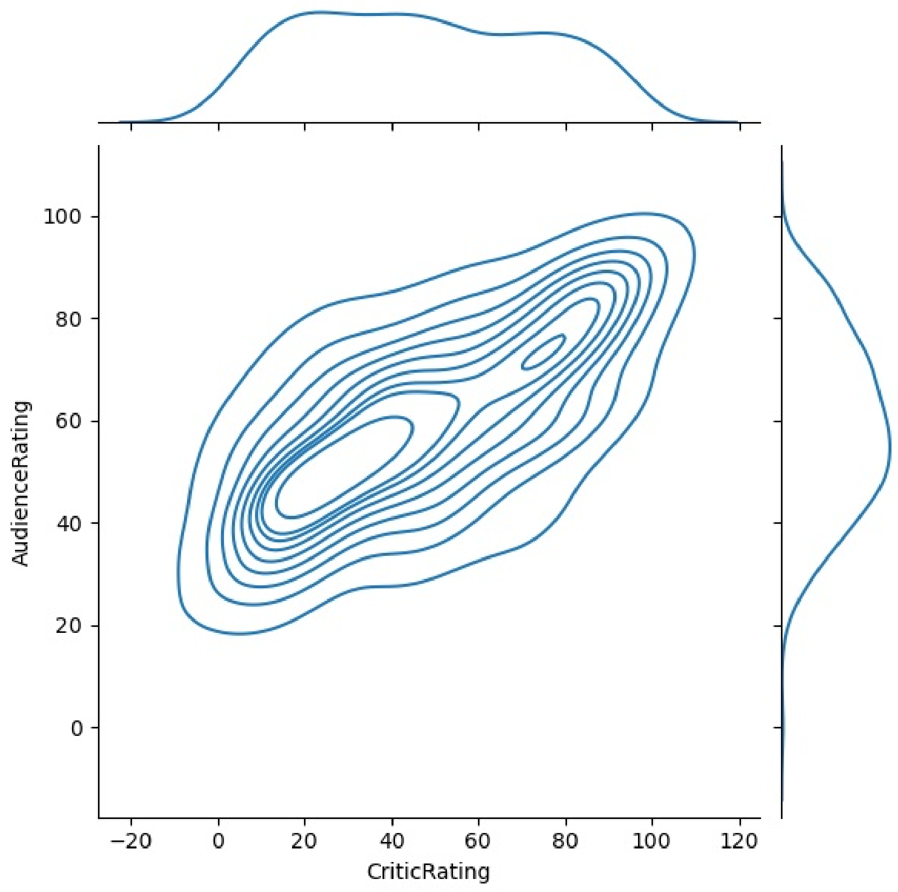
In [30]:
```
j=sns.jointplot(data=movies, x='CriticRating',y='AudienceRating',kind='scatter')
```



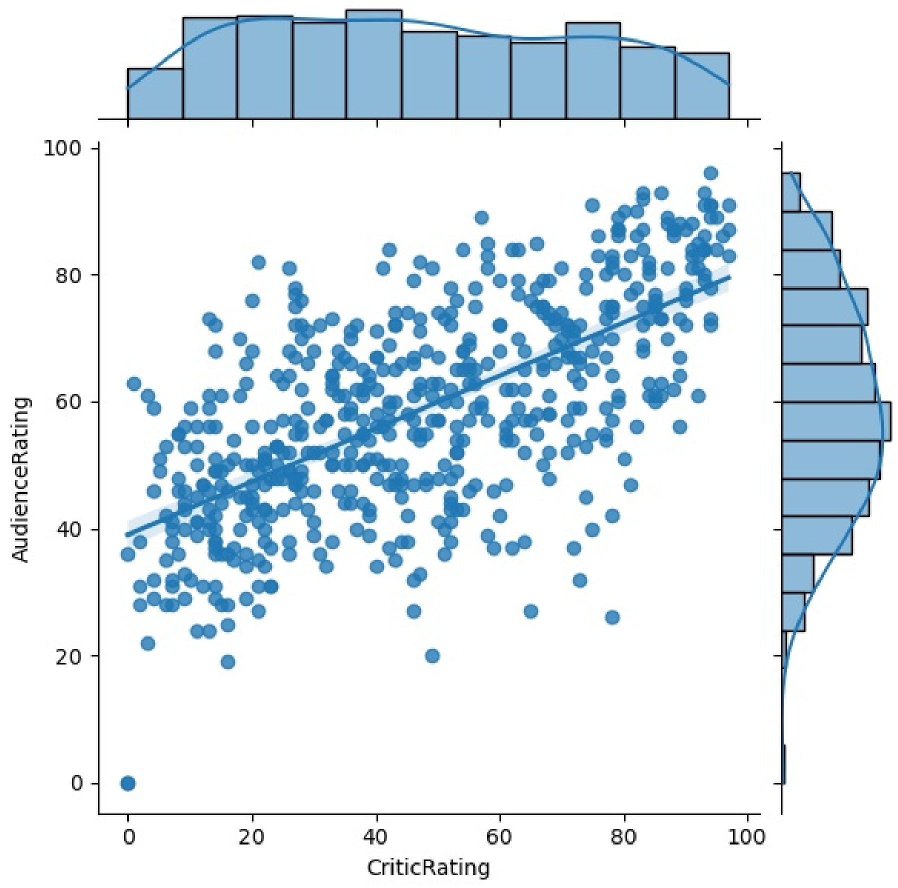
In [31]:
```
j=sns.jointplot(data=movies, x='CriticRating',y='AudienceRating',kind='hex')
```



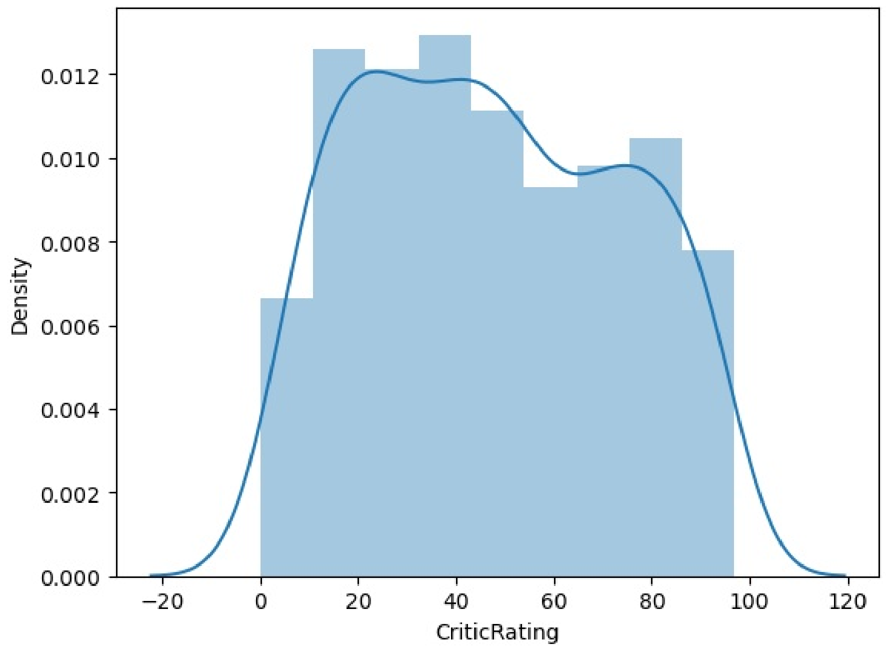
In [32]:
```
j=sns.jointplot(data=movies, x='CriticRating',y='AudienceRating',kind='kde')
```



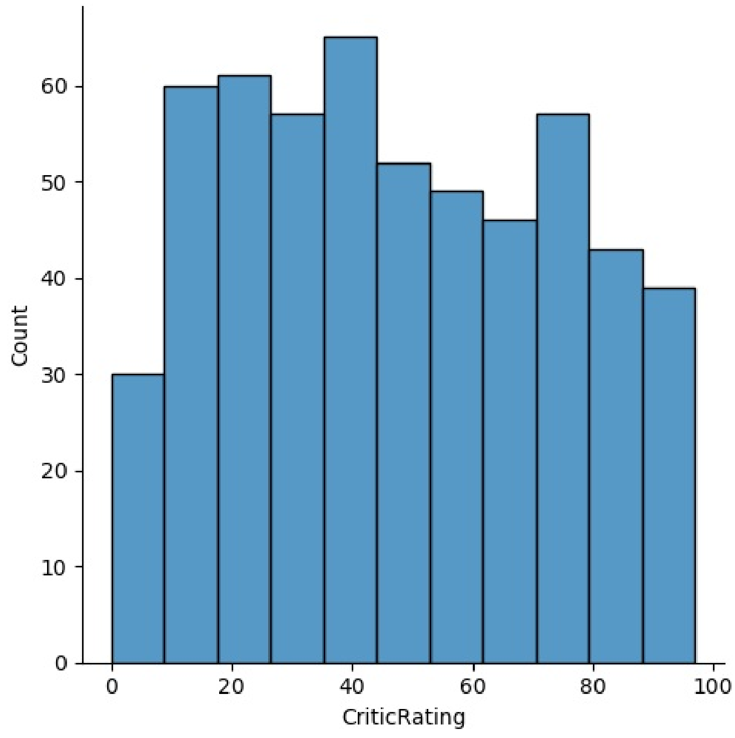
In [33]:
```
j=sns.jointplot(data=movies, x='CriticRating',y='AudienceRating',kind='reg')
```



In [34]:
```
m1=sns.distplot(movies.CriticRating)
```

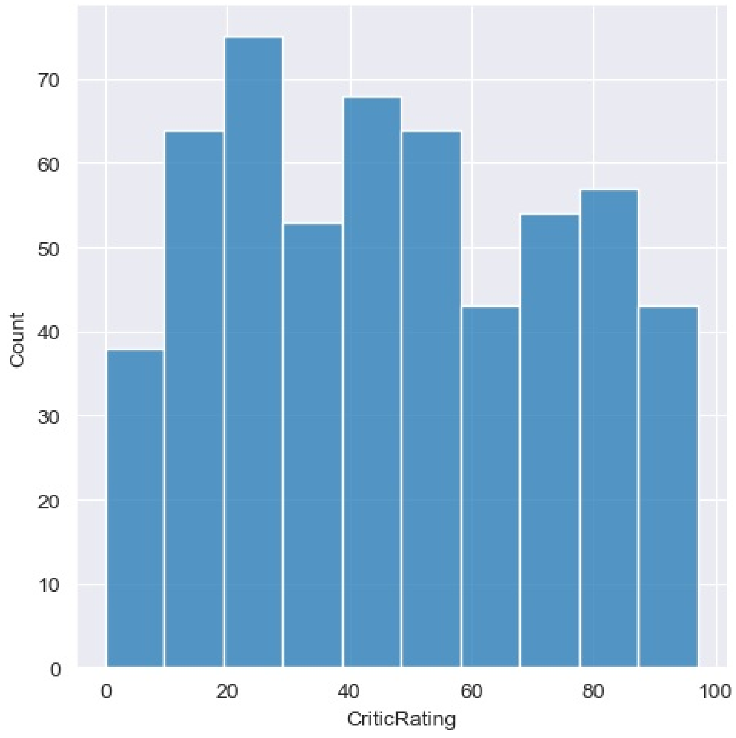


```
In [35]: m1=sns.displot(movies.CriticRating)
```

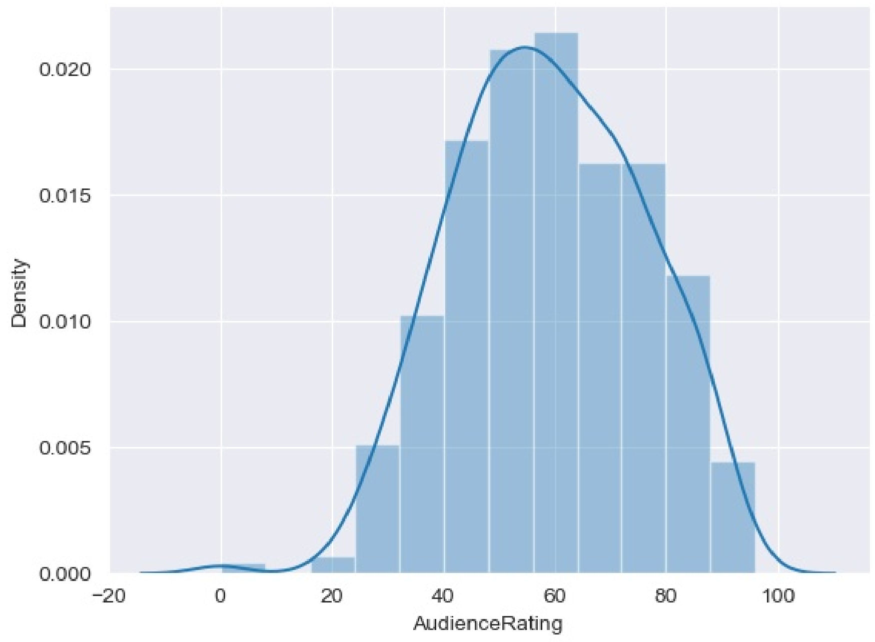


```
In [36]: sns.set_style('darkgrid')
```

```
In [37]: m1=sns.displot(movies.CriticRating,bins=10)
```



```
In [38]: m2=sns.distplot(movies.AudienceRating,bins=12)
```



```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js