Playstore data analysis by Ayyappa Chowdary

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
data=pd.read_csv(r"B:\dump code\xl files\playstore-analysis.csv")
```

In [3]:
```
data.head(10)
```

Out[3]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10,000+	Free	0	Everyone	Art & Design	January 7, 2018
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167	5600.0	50,000+	Free	0	Everyone	Art & Design	March 26, 2017
6	Smoke Effect Photo Maker - Smoke Editor	ART_AND_DESIGN	3.8	178	19000.0	50,000+	Free	0	Everyone	Art & Design	April 26, 2018
7	Infinite Painter	ART_AND_DESIGN	4.1	36815	29000.0	1,000,000+	Free	0	Everyone	Art & Design	June 14, 2018
8	Garden Coloring Book	ART_AND_DESIGN	4.4	13791	33000.0	1,000,000+	Free	0	Everyone	Art & Design	September 20, 2017
9	Kids Paint Free - Drawing Fun	ART_AND_DESIGN	4.7	121	3100.0	10,000+	Free	0	Everyone	Art & Design;Creativity	July 3, 2018

Rename the columns

In [4]:
```
#We rename the column name
data.rename(columns={"Content Rating" : "ContentRating","Rating" : "UserRatings"},inplace = True)
```

In [5]:
```
data.head(5)
```

Out[5]:

	App	Category	UserRatings	Reviews	Size	Installs	Type	Price	ContentRating	Genres
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10,000+	Free	0	Everyone	Art & Design
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500,000+	Free	0	Everyone	Art & Design;Pretend Play
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5,000,000+	Free	0	Everyone	Art & Design
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50,000,000+	Free	0	Teen	Art & Design
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100,000+	Free	0	Everyone	Art & Design;Creativity

Understand and describe the data

In [6]:
```
data.describe()
```

Out[6]:

	UserRatings	Size
count	9367.000000	10841.000000
mean	4.193338	21516.529524
std	0.537431	20746.537567
min	1.000000	8.500000
25%	4.000000	5900.000000
50%	4.300000	18000.000000
75%	4.500000	26000.000000
max	19.000000	100000.000000

In [7]:
```
data.head()
```

Out[7]:

	App	Category	UserRatings	Reviews	Size	Installs	Type	Price	ContentRating	Genres
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10,000+	Free	0	Everyone	Art & Design
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500,000+	Free	0	Everyone	Art & Design;Pretend Play
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5,000,000+	Free	0	Everyone	Art & Design
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50,000,000+	Free	0	Teen	Art & Design
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100,000+	Free	0	Everyone	Art & Design;Creativity

In [8]:
```
#Shape  the data
data.shape
```

Out[8]:
```
(10841, 13)
```

In [9]:
```
data.dropna(how="any",subset=["UserRatings"],inplace=True)
```

In [10]:
```
data.shape
```

Out[10]:
```
(9367, 13)
```

In [11]:
```
print("Totoal Null values is ",data.isnull().sum())
data.isnull().sum()
```

```
Totoal Null values is  App              0
Category          0
UserRatings       0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
ContentRating     1
Genres            0
Last Updated      0
Current Ver       4
Android Ver       3
dtype: int64
```

Out[11]:
```
App               0
Category          0
UserRatings       0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
ContentRating     1
Genres            0
Last Updated      0
Current Ver       4
Android Ver       3
dtype: int64
```

In [12]:
```
data.loc[data["Android Ver"].isnull()]
```

Out[12]:

	App	Category	UserRatings	Reviews	Size	Installs	Type	Price	ContentRating
4453	[substratum] Vacuum: P	PERSONALIZATION	4.4	230	11000.000000	1,000+	Paid	$1.49	Everyone
4490	Pi Dark [substratum]	PERSONALIZATION	4.5	189	2100.000000	10,000+	Free	0	Everyone
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0M	21516.529524	Free	0	Everyone	NaN

In [13]:
```
data.loc[data["Current Ver"].isnull()]
```

Out[13]:

	App	Category	UserRatings	Reviews	Size	Installs	Type	Price	ContentRating	Genres
15	Learn To Draw Kawaii Characters	ART_AND_DESIGN	3.2	55	2700.0	5,000+	Free	0	Everyone	Art & Design
1553	Market Update Helper	LIBRARIES_AND_DEMO	4.1	20145	11.0	1,000,000+	Free	0	Everyone	Libraries & Demo
6322	Virtual DJ Sound Mixer	TOOLS	4.2	4010	8700.0	500,000+	Free	0	Everyone	Tools
7333	Dots puzzle	FAMILY	4.0	179	14000.0	50,000+	Paid	$0.99	Everyone	Puzzle

In [14]:
```
#Now we remove  one null values from data Android Ver
data.drop([10472], axis=0,inplace = True)
data.isnull().sum()
```

Out[14]:
```
App               0
Category          0
UserRatings       0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
ContentRating     0
Genres            0
Last Updated      0
Current Ver       4
Android Ver       2
dtype: int64
```

In [15]:
```
value = str(data["Android Ver"].mode())
data["Android Ver"].fillna(value = value, inplace = True)
data.isnull().sum()
```

Out[15]:
```
App               0
Category          0
UserRatings       0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
ContentRating     0
Genres            0
Last Updated      0
Current Ver       4
Android Ver       0
dtype: int64
```

In [16]:
```
#now we do for Current Ver
value1 = data["Current Ver"].mode()[0]
data["Current Ver"].fillna(value = value1, inplace = True)
print("Total null values: ",data.isnull().sum().sum())
```

```
Total null values:  0
```

So we removed all the Null values in dataset, Now we clean the data and we have to correct the data types

In [17]:
```
numeric=data.select_dtypes(include=np.number).columns
numeric
```

Out[17]:
```
Index(['UserRatings', 'Size'], dtype='object')
```

In [18]:
```
data.dtypes
```

Out[18]:
```
App               object
Category          object
UserRatings      float64
Reviews           object
Size             float64
Installs          object
Type              object
Price             object
ContentRating     object
Genres            object
Last Updated      object
Current Ver       object
Android Ver       object
dtype: object
```

Correcting Price , Reviews , Install datatypes : because we cannot use when it in object fromat, That's why we changing to float format

In [19]:
```
data["Price"] = data["Price"].str.replace("$","")
data["Price"] = data["Price"].str.replace(",","")
data["Price"] = data["Price"].astype("float")
```

In [20]:
```
data["Installs"] = data["Installs"].str.replace("+","")
data["Installs"] = data["Installs"].str.replace(",","")
data["Installs"] = data["Installs"].astype("int")
```

In [21]:
```
data["Reviews"] = data["Reviews"].astype("int")
```

In [22]:
```
data.dtypes
```

Out[22]:
```
App               object
Category          object
UserRatings      float64
Reviews            int32
Size             float64
Installs           int32
Type              object
Price            float64
ContentRating     object
Genres            object
Last Updated      object
Current Ver       object
Android Ver       object
dtype: object
```

Sanity check : Means, It is a basic test to quickly evaluate whether a claim or the result of a calculation can possibly be true. It is a simple check to see if the produced material is rational

In [23]:
```
data.drop(data[data.UserRatings>5].index, inplace = True)
data.drop(data[data.UserRatings<1].index,inplace = True)
data.shape
```

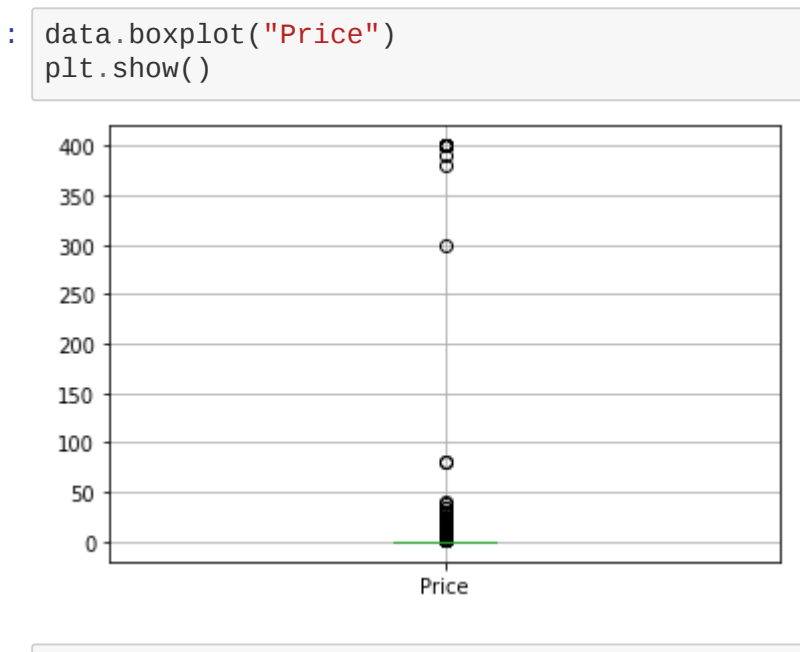
Out[23]:
```
(9366, 13)
```

In [24]:
```
data.drop(data[data.Reviews>data.Installs].index,inplace = True)
data.shape
```

Out[24]:
```
(9359, 13)
```

outliers :- here the main part arrives, we have to remove outliers. outliers means unwanted or unstructured data in dataset.

In [25]:
```
data.boxplot("Price")
plt.show()
```



In [26]:
```
data.drop(data[data.Price>200].index,inplace=True)
data.shape
```

Out[26]:
```
(9344, 13)
```

Draw the boxplot for price

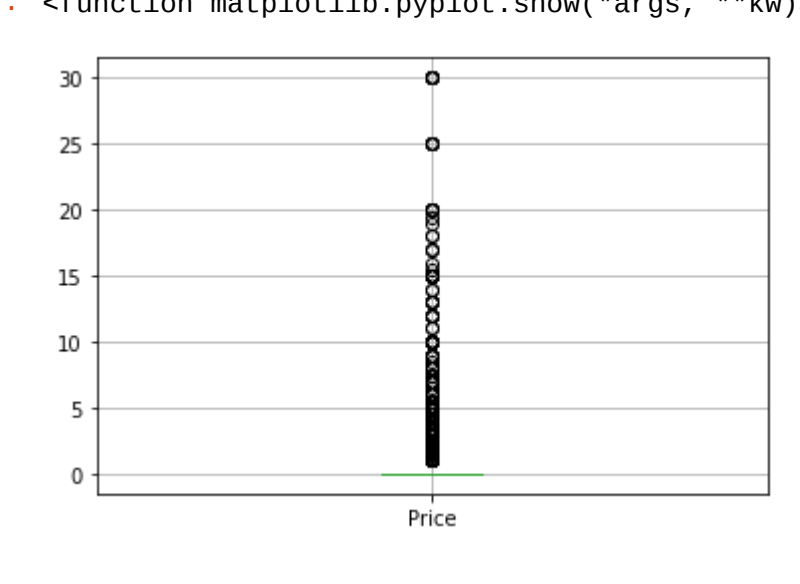
In [27]:
```
data.boxplot("Price")
plt.show()
```

In [28]:
```
data.drop(data[data.Price>30].index, inplace=True)
data.shape
```

Out[28]:
```
(9338, 13)
```

In [29]:
```
data.boxplot("Price")
plt.show
```

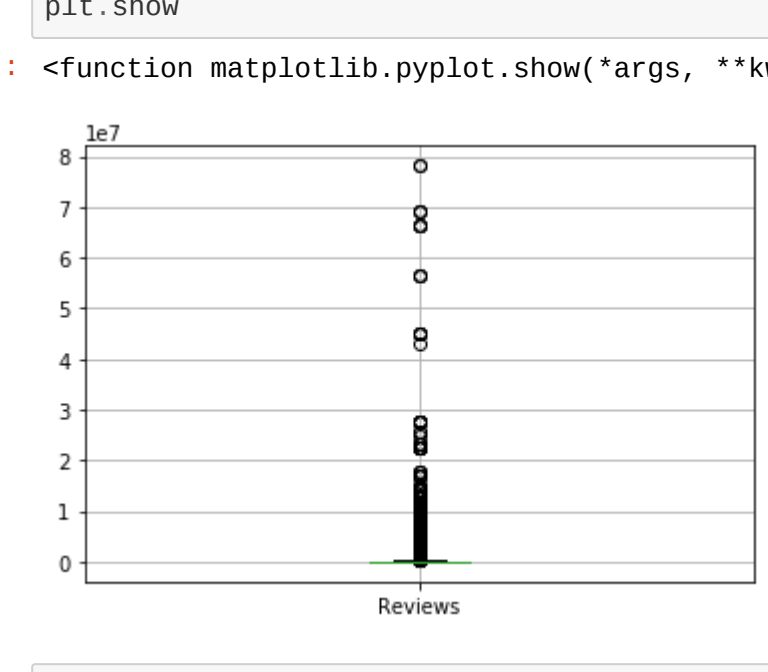
Out[29]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



Draw the boxplot for Reviews

In [30]:
```
#Reviews
data.boxplot("Reviews")
plt.show
```

Out[30]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```

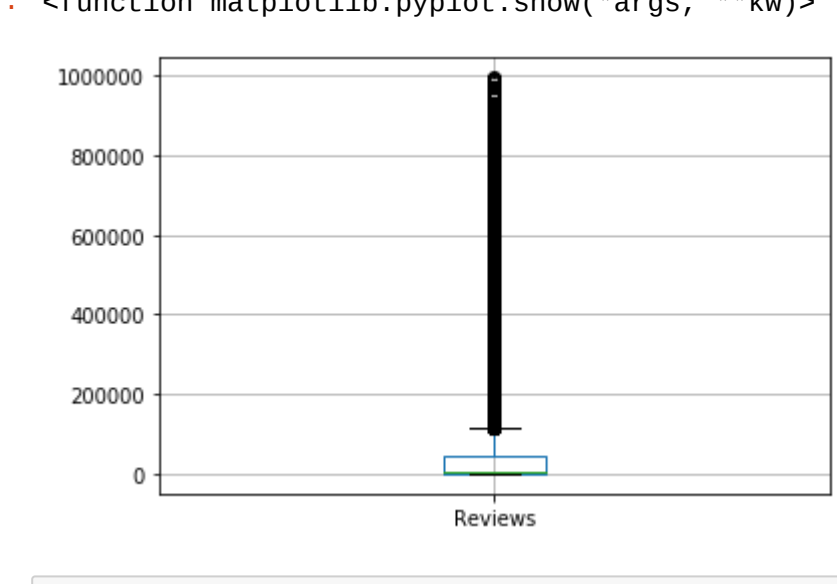


In [31]:
```
data.drop(data[data.Reviews>1000000].index,inplace=True)
data.shape
```

Out[31]:
```
(8634, 13)
```

In [32]:
```
data.boxplot("Reviews")
plt.show
```

Out[32]:
```
<function matplotlib.pyplot.show(*args, **kw)>
```



In [33]:
```
#installs
percentail=np.percentile(data["Installs"],95)
percentail
data.shape
```

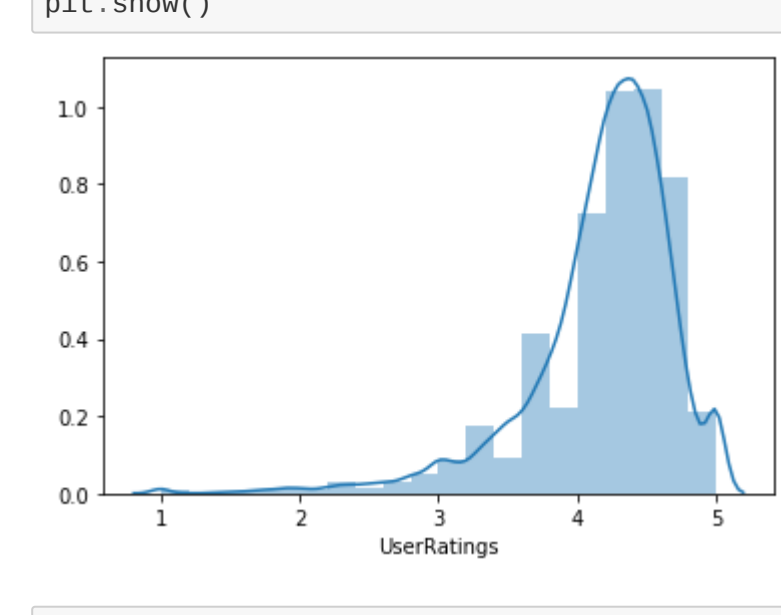
Out[33]:
```
(8634, 13)
```

In [34]:
```
data.drop(data[data.Installs>percentail].index,inplace=True)
data.shape
```

Out[34]:
```
(8435, 13)
```

In [60]:
```
#Data analysis to answer business questions or models
```

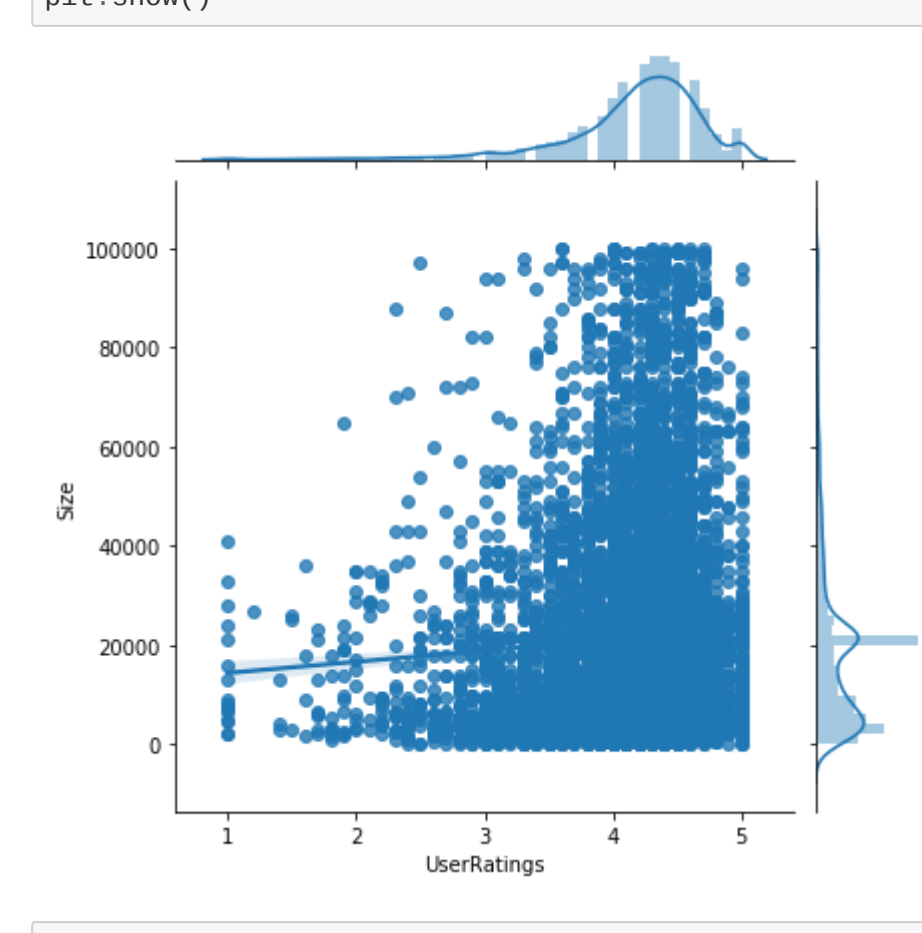
In [35]:
```
sns.distplot(data["UserRatings"],bins=20)
plt.show()
```



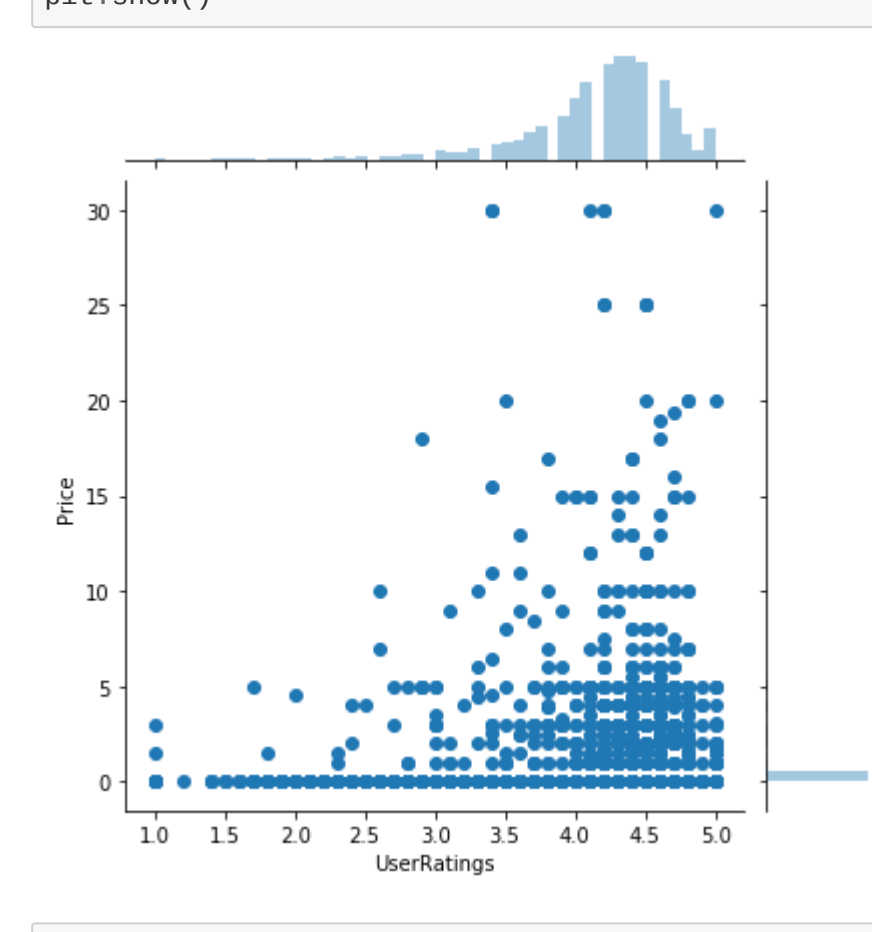
In [36]:
```
data["ContentRating"].value_counts()
```

Out[36]:
```
Everyone           6782
Teen                900
Mature 17+          417
Everyone 10+        332
Adults only 18+       3
Unrated               1
Name: ContentRating, dtype: int64
```

In [37]:
```
sns.jointplot("UserRatings", "Size", data=data, kind="reg")
plt.show()
```



In [38]:
```
sns.jointplot("UserRatings","Price",data=data)
plt.show()
```

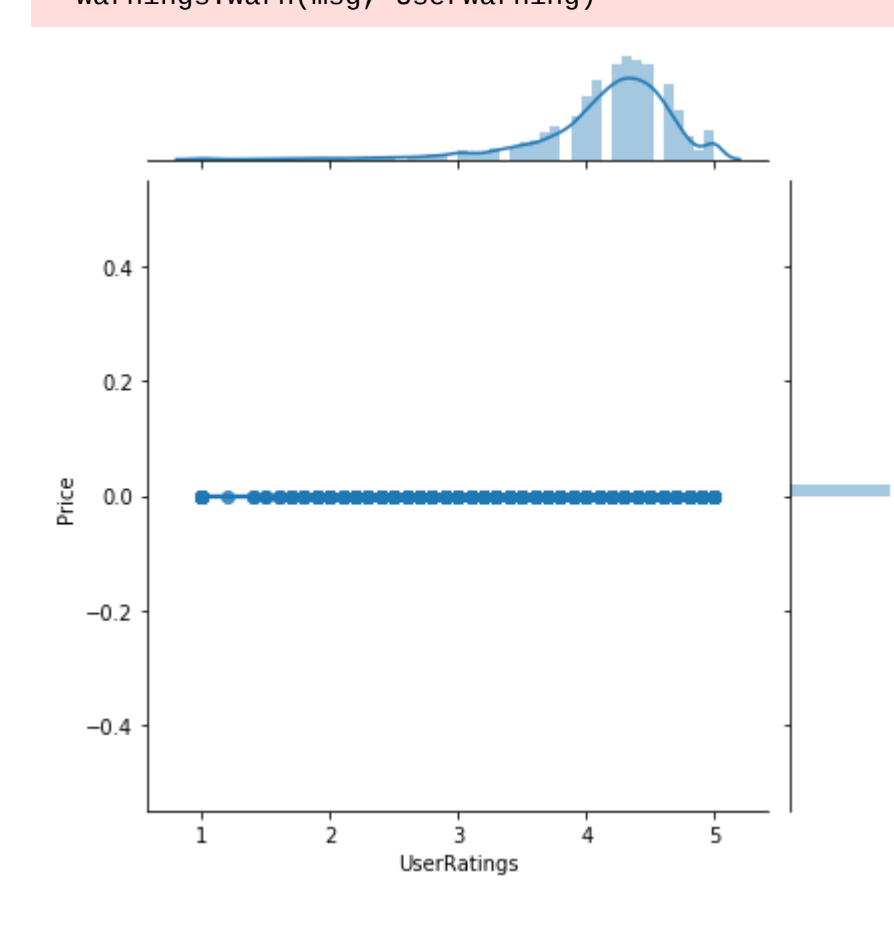


In [39]:
```
print(data.shape)
data.drop(data[data.Price>0].index,inplace=True)
print(data.shape)
```

```
(8435, 13)
(7814, 13)
```

In [40]:
```
sns.jointplot("UserRatings","Price",data=data,kind="reg")
plt.show()
```

```
C:\Users\Ayyappa\Anaconda3\lib\site-packages\seaborn\distributions.py:288: UserWarning: Data must have variance to compute a kernel density estimate.
  warnings.warn(msg, UserWarning)
```



For Free apps the rating starts from 2.5 to 5 Pattern changes to straight line from the graph for free apps got 4-5 rating Overall infernce from price and rating is when price is 0 rating is high from 4-5 when price is greater than zero same as like price zero there is no much difference customer prefer free apps than paid apps

Now we can draw the Pairplot with Numerical Columns

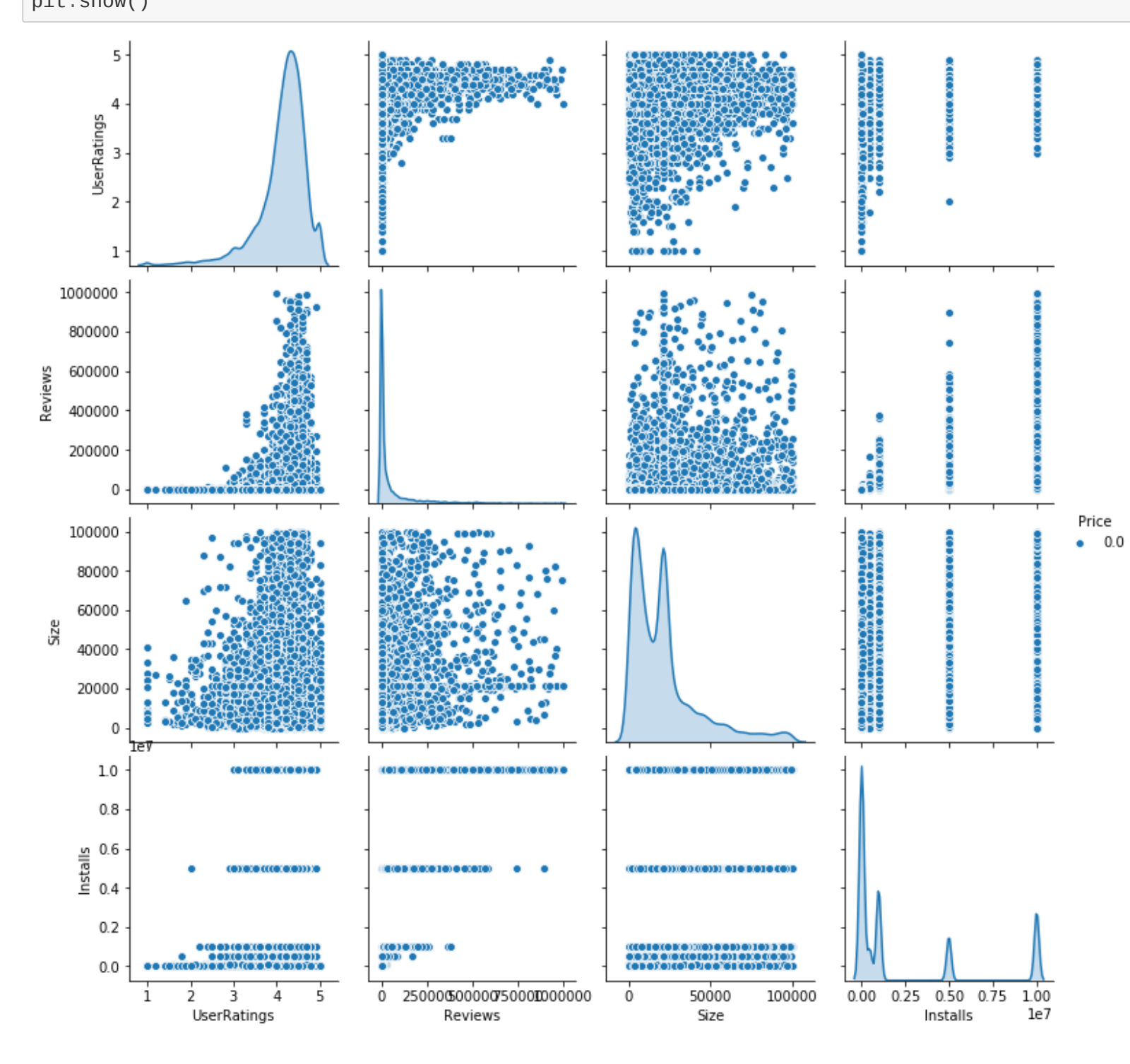
In [41]:
```
numeric=data.select_dtypes(include=np.number)
numeric
```

Out[41]:

	UserRatings	Reviews	Size	Installs	Price
0	4.1	159	19000.000000	10000	0.0
1	3.9	967	14000.000000	500000	0.0
2	4.7	87510	8700.000000	5000000	0.0
4	4.3	967	2800.000000	100000	0.0
5	4.4	167	5600.000000	50000	0.0
...	...	...	...	...	...
10834	4.0	7	2600.000000	500	0.0
10836	4.5	38	53000.000000	5000	0.0
10837	5.0	4	3600.000000	100	0.0
10839	4.5	114	21516.529524	1000	0.0
10840	4.5	398307	19000.000000	10000000	0.0

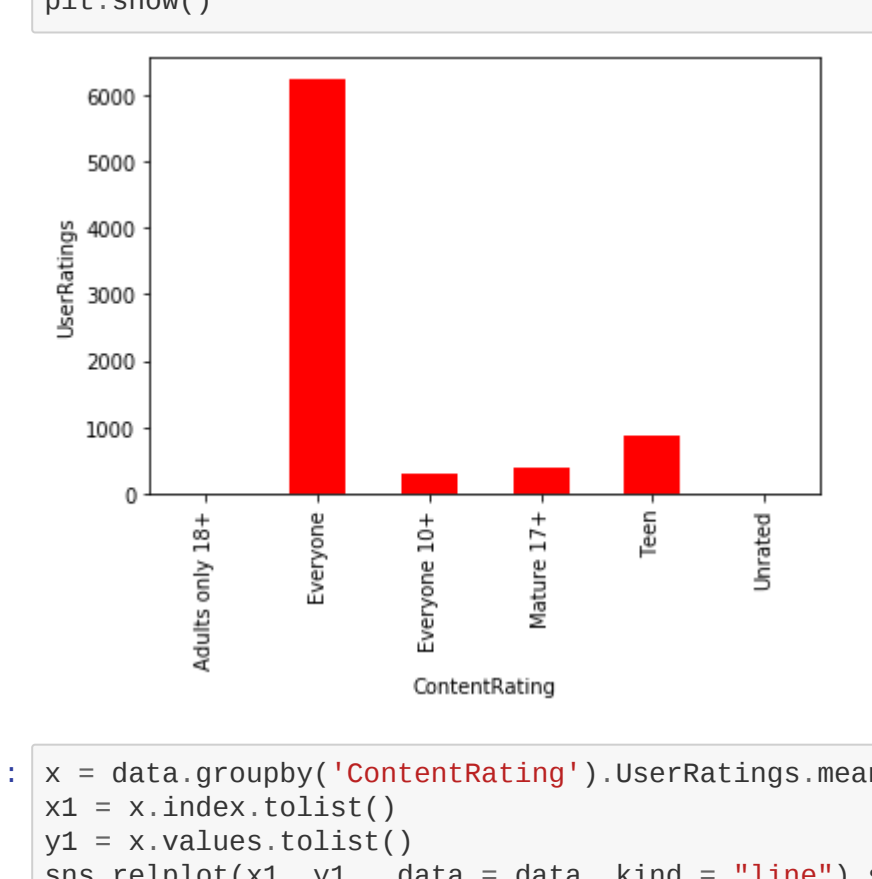
7814 rows × 5 columns

In [42]:
```
sns.pairplot(numeric,hue="Price")
plt.show()
```

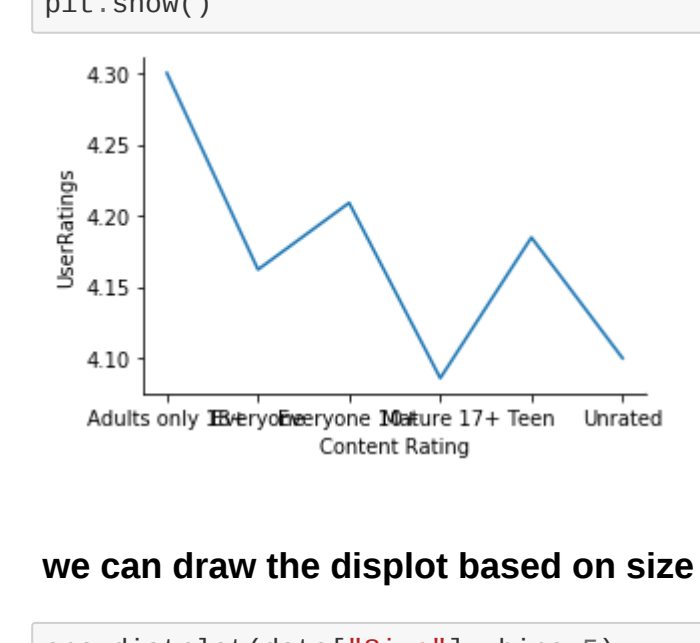


ContentRating vs UserRatings

In [43]:
```
data.groupby(['ContentRating'])['UserRatings'].count().plot.bar(color="red")
plt.ylabel('UserRatings')
plt.show()
```

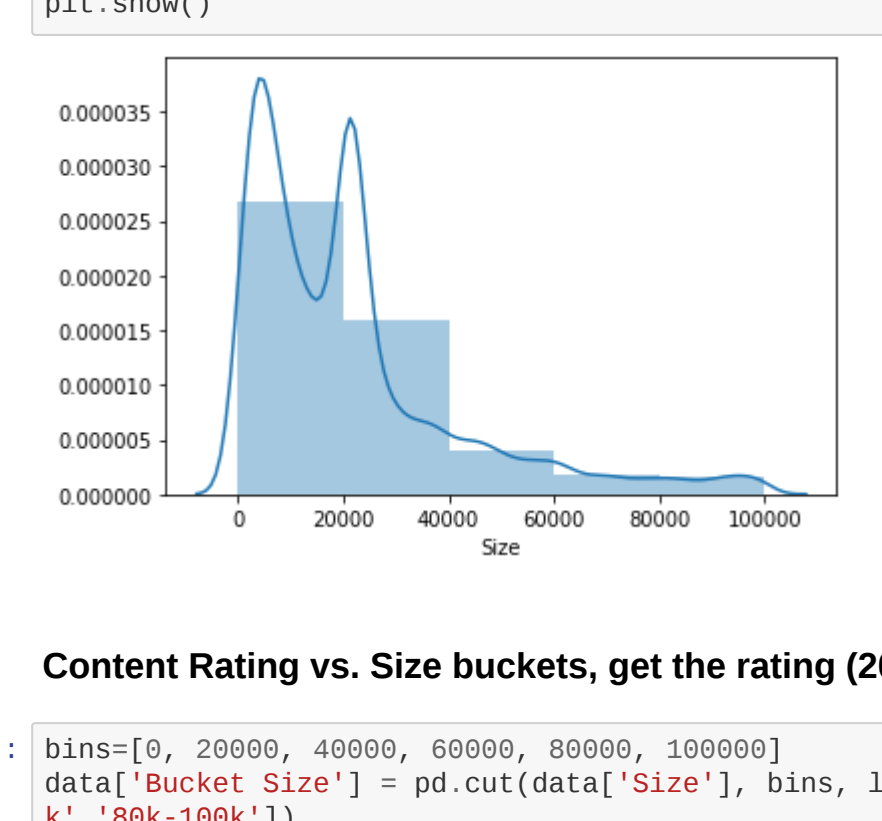


In [46]:
```
x = data.groupby('ContentRating').UserRatings.mean()
x1 = x.index.tolist()
y1 = x.values.tolist()
sns.relplot(x1 ,y1 , data = data, kind = "line").set(xlabel = "Content Rating", ylabel = "UserRatings")
plt.show()
```



we can draw the displot based on size

In [48]:
```
sns.distplot(data["Size"], bins=5)
plt.show()
```



Content Rating vs. Size buckets, get the rating (20th percentile) for each combination

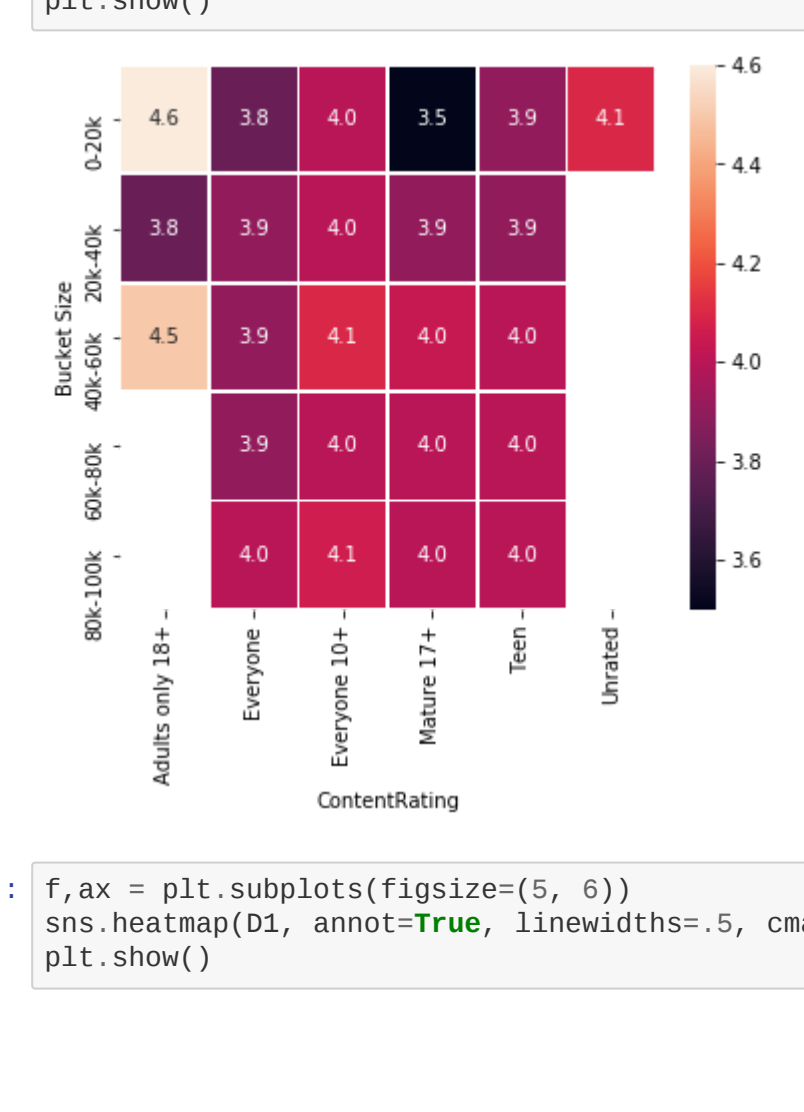
In [56]:
```
bins=[0, 20000, 40000, 60000, 80000, 100000]
data['Bucket Size'] = pd.cut(data['Size'], bins, labels=['0-20k','20k-40k','40k-60k','60k-80k','80k-100k'])
D1=pd.pivot_table(data, values='UserRatings', index='Bucket Size', columns='ContentRating',
                  aggfunc=lambda x:np.quantile(x,0.2))
D1
```

Out[56]:

ContentRating / Bucket Size	Adults only 18+	Everyone	Everyone 10+	Mature 17+	Teen	Unrated
0-20k	4.6	3.8	4.00	3.50	3.9	4.1
20k-40k	3.8	3.9	4.00	3.90	3.9	NaN
40k-60k	4.5	3.9	4.10	4.04	4.0	NaN
60k-80k	NaN	3.9	4.00	4.00	4.0	NaN
80k-100k	NaN	4.0	4.06	4.00	4.0	NaN

Now can plot with heatmap

In [57]:
```
f,ax = plt.subplots(figsize=(6, 5))
sns.heatmap(D1, annot=True, linewidths=.8, fmt='1.1f',ax=ax)
plt.show()
```



In [59]:
```
f,ax = plt.subplots(figsize=(5, 6))
sns.heatmap(D1, annot=True, linewidths=.5, cmap='Greens',fmt='1.1f',ax=ax)
plt.show()
```