1. Load the data file using pandas.

In [1]:

```
import pandas as pd
```

In [2]:

```
import numpy as np
```

In [3]:

```
data = pd.read_csv('googleplaystore.csv')
```

In [4]:

```
data.head()
```

Out[4]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	G
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & D
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Design;Pr
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & D
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & D
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Design;Cre

In [5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10841 non-null  object 
 1   Category        10841 non-null  object 
 2   Rating          9367 non-null   float64
 3   Reviews         10841 non-null  object 
 4   Size            10841 non-null  object 
 5   Installs        10841 non-null  object 
 6   Type            10840 non-null  object 
 7   Price           10841 non-null  object 
 8   Content Rating  10840 non-null  object 
 9   Genres          10841 non-null  object 
 10  Last Updated    10841 non-null  object 
 11  Current Ver     10833 non-null  object 
 12  Android Ver     10838 non-null  object 
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

In [6]:

```
data.shape
```

Out[6]:

```
(10841, 13)
```

2.Check for null values in the data. Get the number of null values for each column.

In [7]:

```
data.isnull().any()
```

Out[7]:

```
App               False
Category          False
Rating             True
Reviews           False
Size              False
Installs          False
Type               True
Price             False
Content Rating     True
Genres            False
Last Updated      False
Current Ver        True
Android Ver        True
dtype: bool
```

In [8]:

```
data.isnull().sum()
```

Out[8]:
```
App                  0
Category             0
Rating            1474
Reviews              0
Size                 0
Installs             0
Type                 1
Price                0
Content Rating       1
Genres               0
Last Updated         0
Current Ver          8
Android Ver          3
dtype: int64
```

3. Drop records with nulls in any of the columns.

In [9]:
```
data = data.dropna()
```

In [10]:
```
data.isnull().any()
```

Out[10]:
```
App               False
Category          False
Rating            False
Reviews           False
Size              False
Installs          False
Type              False
Price             False
Content Rating    False
Genres            False
Last Updated      False
Current Ver       False
Android Ver       False
dtype: bool
```

In [11]:
```
data.shape
```

Out[11]:
```
(9360, 13)
```

4(1) Variables seem to have incorrect type and inconsistent formatting. You need to fix them:

Size column has sizes in Kb as well as Mb. To analyze, you’ll need to convert these to numeric.

In [12]:
```
data["Size"] = [ float(i.split('M')[0]) if 'M' in i else float(0) for i in data["Size"
```

In [13]:
```
data.head()
```

Out[13]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	G
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19.0	10,000+	Free	0	Everyone	Art & D
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14.0	500,000+	Free	0	Everyone	Design;Pr
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5,000,000+	Free	0	Everyone	Art & D
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25.0	50,000,000+	Free	0	Teen	Art & D
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100,000+	Free	0	Everyone	Design;Cre

Extract the numeric value from the column

In [14]:
```
data["Size"] = 1000 * data["Size"]
```

In [15]:
```
data
```

Out[15]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Conte Rati
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10,000+	Free	0	Everyo
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500,000+	Free	0	Everyo
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5,000,000+	Free	0	Everyo
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50,000,000+	Free	0	Te
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100,000+	Free	0	Everyo
...	...	...	...	...	...	...	...	...	
10834	FR Calculator	FAMILY	4.0	7	2600.0	500+	Free	0	Everyo
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53000.0	5,000+	Free	0	Everyo
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3600.0	100+	Free	0	Everyo
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	0.0	1,000+	Free	0	Mat 1
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19000.0	10,000,000+	Free	0	Everyo

9360 rows × 13 columns

4.(2)Reviews is a numeric field that is loaded as a string field. Convert it to numeric (int/float).

In [16]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9360 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             9360 non-null   object 
 1   Category        9360 non-null   object 
 2   Rating          9360 non-null   float64
 3   Reviews         9360 non-null   object 
 4   Size            9360 non-null   float64
 5   Installs        9360 non-null   object 
 6   Type            9360 non-null   object 
 7   Price           9360 non-null   object 
 8   Content Rating  9360 non-null   object 
 9   Genres          9360 non-null   object 
 10  Last Updated    9360 non-null   object 
 11  Current Ver     9360 non-null   object 
 12  Android Ver     9360 non-null   object 
dtypes: float64(2), object(11)
memory usage: 1023.8+ KB
```

In [17]:

```
data["Reviews"] = data["Reviews"].astype(float)
```

In [18]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9360 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             9360 non-null   object 
 1   Category        9360 non-null   object 
 2   Rating          9360 non-null   float64
 3   Reviews         9360 non-null   float64
 4   Size            9360 non-null   float64
 5   Installs        9360 non-null   object 
 6   Type            9360 non-null   object 
 7   Price           9360 non-null   object 
 8   Content Rating  9360 non-null   object 
 9   Genres          9360 non-null   object 
 10  Last Updated    9360 non-null   object 
 11  Current Ver     9360 non-null   object 
 12  Android Ver     9360 non-null   object 
dtypes: float64(3), object(10)
memory usage: 1023.8+ KB
```

4.(3)Installs field is currently stored as string and has values like 1,000,000+.

In [19]:

```
data["Installs"] = [ float(i.replace('+','').replace(',', '')) if '+' in i or ',' in i
```

In [20]:

```
data.head()
```

Out[20]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19000.0	10000.0	Free	0	Everyone	Art &
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000.0	Free	0	Everyone	Design
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000.0	Free	0	Everyone	Art &
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25000.0	50000000.0	Free	0	Teen	Art &
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000.0	Free	0	Everyone	Design;(

In [21]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9360 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             9360 non-null   object 
 1   Category        9360 non-null   object 
 2   Rating          9360 non-null   float64
 3   Reviews         9360 non-null   float64
 4   Size            9360 non-null   float64
 5   Installs        9360 non-null   float64
 6   Type            9360 non-null   object 
 7   Price           9360 non-null   object 
 8   Content Rating  9360 non-null   object 
 9   Genres          9360 non-null   object 
 10  Last Updated    9360 non-null   object 
 11  Current Ver     9360 non-null   object 
 12  Android Ver     9360 non-null   object 
dtypes: float64(4), object(9)
memory usage: 1023.8+ KB
```

In [22]:
```
data["Installs"] = data["Installs"].astype(int)
```

In [23]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9360 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             9360 non-null   object 
 1   Category        9360 non-null   object 
 2   Rating          9360 non-null   float64
 3   Reviews         9360 non-null   float64
 4   Size            9360 non-null   float64
 5   Installs        9360 non-null   int32  
 6   Type            9360 non-null   object 
 7   Price           9360 non-null   object 
 8   Content Rating  9360 non-null   object 
 9   Genres          9360 non-null   object 
 10  Last Updated    9360 non-null   object 
 11  Current Ver     9360 non-null   object 
 12  Android Ver     9360 non-null   object 
dtypes: float64(3), int32(1), object(9)
memory usage: 987.2+ KB
```

4.(4)Price field is a string and has $symbol. Remove“'$ sign, and convert it to numeric.

In [24]:
```
data['Price'] = [ float(i.split('$')[1]) if '$' in i else float(0) for i in data['Pric
```

In [25]:
```
data.head()
```

Out[25]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19000.0	10000	Free	0.0	Everyone	Art &
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000	Free	0.0	Everyone	Design;F
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000	Free	0.0	Everyone	Art &
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25000.0	50000000	Free	0.0	Teen	Art &
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000	Free	0.0	Everyone	Design;Cr

In [26]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9360 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             9360 non-null   object 
 1   Category        9360 non-null   object 
 2   Rating          9360 non-null   float64
 3   Reviews         9360 non-null   float64
 4   Size            9360 non-null   float64
 5   Installs        9360 non-null   int32  
 6   Type            9360 non-null   object 
 7   Price           9360 non-null   float64
 8   Content Rating  9360 non-null   object 
 9   Genres          9360 non-null   object 
 10  Last Updated    9360 non-null   object 
 11  Current Ver     9360 non-null   object 
 12  Android Ver     9360 non-null   object 
dtypes: float64(4), int32(1), object(8)
memory usage: 987.2+ KB
```

In [27]:
```
data["Price"] = data["Price"].astype(int)
```

In [28]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9360 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             9360 non-null   object 
 1   Category        9360 non-null   object 
 2   Rating          9360 non-null   float64
 3   Reviews         9360 non-null   float64
 4   Size            9360 non-null   float64
 5   Installs        9360 non-null   int32  
 6   Type            9360 non-null   object 
 7   Price           9360 non-null   int32  
 8   Content Rating  9360 non-null   object 
 9   Genres          9360 non-null   object 
 10  Last Updated    9360 non-null   object 
 11  Current Ver     9360 non-null   object 
 12  Android Ver     9360 non-null   object 
dtypes: float64(3), int32(2), object(8)
memory usage: 950.6+ KB
```

5 Sanity checks:

5(1) Average rating should be between 1 and 5 as only these values are allowed on the play store.

Drop the rows that have a value outside this range.

In [29]:
```
data.shape
```

Out[29]:
```
(9360, 13)
```

In [30]:
```
data.drop(data[(data['Reviews'] < 1) & (data['Reviews'] > 5 )].index, inplace = True)
```

In [31]:
```
data.shape
```

Out[31]:
```
(9360, 13)
```

5 (2) Reviews should not be more than installs as only those who installed can review the app.

If there are any such records, drop them.

In [32]:
```
data.shape
```

Out[32]:
```
(9360, 13)
```

In [33]:
```
data.drop(data[data['Installs'] < data['Reviews'] ].index, inplace = True)
```

In [34]:
```
data.shape
```

Out[34]:
```
(9353, 13)
```

5(3) For free apps (type = “Free”), the price should not be >0. Drop any such rows.

In [35]:
```
data.shape
```

Out[35]:
```
(9353, 13)
```

In [36]:
```
data.drop(data[(data['Type'] =='Free') & (data['Price'] > 0 )].index, inplace = True)
```

In [37]:
```
data.shape
```

Out[37]:
```
(9353, 13)
```

5. Performing univariate analysis:

Boxplot for Price

Are there any outliers? Think about the price of usual apps on Play Store.

In [38]:
```
pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\rakesh\anaconda3\lib\site-packages
(0.11.2)
Requirement already satisfied: matplotlib>=2.2 in c:\users\rakesh\anaconda3\lib\site-
packages (from seaborn) (3.5.2)
Requirement already satisfied: numpy>=1.15 in c:\users\rakesh\anaconda3\lib\site-pack
ages (from seaborn) (1.21.5)
Requirement already satisfied: pandas>=0.23 in c:\users\rakesh\anaconda3\lib\site-pac
kages (from seaborn) (1.4.4)
Requirement already satisfied: scipy>=1.0 in c:\users\rakesh\anaconda3\lib\site-packa
ges (from seaborn) (1.9.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\rakesh\anaconda3\lib\sit
e-packages (from matplotlib>=2.2->seaborn) (1.4.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\rakesh\anaconda3\lib\site-pa
ckages (from matplotlib>=2.2->seaborn) (9.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\rakesh\anaconda3\lib\site-pac
kages (from matplotlib>=2.2->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\rakesh\anaconda3\lib\sit
e-packages (from matplotlib>=2.2->seaborn) (4.25.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\rakesh\anaconda3\lib\site
-packages (from matplotlib>=2.2->seaborn) (3.0.9)
Requirement already satisfied: packaging>=20.0 in c:\users\rakesh\anaconda3\lib\site-
packages (from matplotlib>=2.2->seaborn) (21.3)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\rakesh\anaconda3\lib
\site-packages (from matplotlib>=2.2->seaborn) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\rakesh\anaconda3\lib\site-pac
kages (from pandas>=0.23->seaborn) (2022.1)
Requirement already satisfied: six>=1.5 in c:\users\rakesh\anaconda3\lib\site-package
s (from python-dateutil>=2.7->matplotlib>=2.2->seaborn) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [39]:
```
pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\rakesh\anaconda3\lib\site-packa
ges (3.5.2)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\rakesh\anaconda3\lib\sit
e-packages (from matplotlib) (4.25.0)
Requirement already satisfied: numpy>=1.17 in c:\users\rakesh\anaconda3\lib\site-pack
ages (from matplotlib) (1.21.5)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\rakesh\anaconda3\lib
\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: cycler>=0.10 in c:\users\rakesh\anaconda3\lib\site-pac
kages (from matplotlib) (0.11.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\rakesh\anaconda3\lib\site
-packages (from matplotlib) (3.0.9)
Requirement already satisfied: pillow>=6.2.0 in c:\users\rakesh\anaconda3\lib\site-pa
ckages (from matplotlib) (9.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\rakesh\anaconda3\lib\sit
e-packages (from matplotlib) (1.4.2)
Requirement already satisfied: packaging>=20.0 in c:\users\rakesh\anaconda3\lib\site-
packages (from matplotlib) (21.3)
Requirement already satisfied: six>=1.5 in c:\users\rakesh\anaconda3\lib\site-package
s (from python-dateutil>=2.7->matplotlib) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [41]:

```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [42]:

```
data.head()
```

Out[42]:

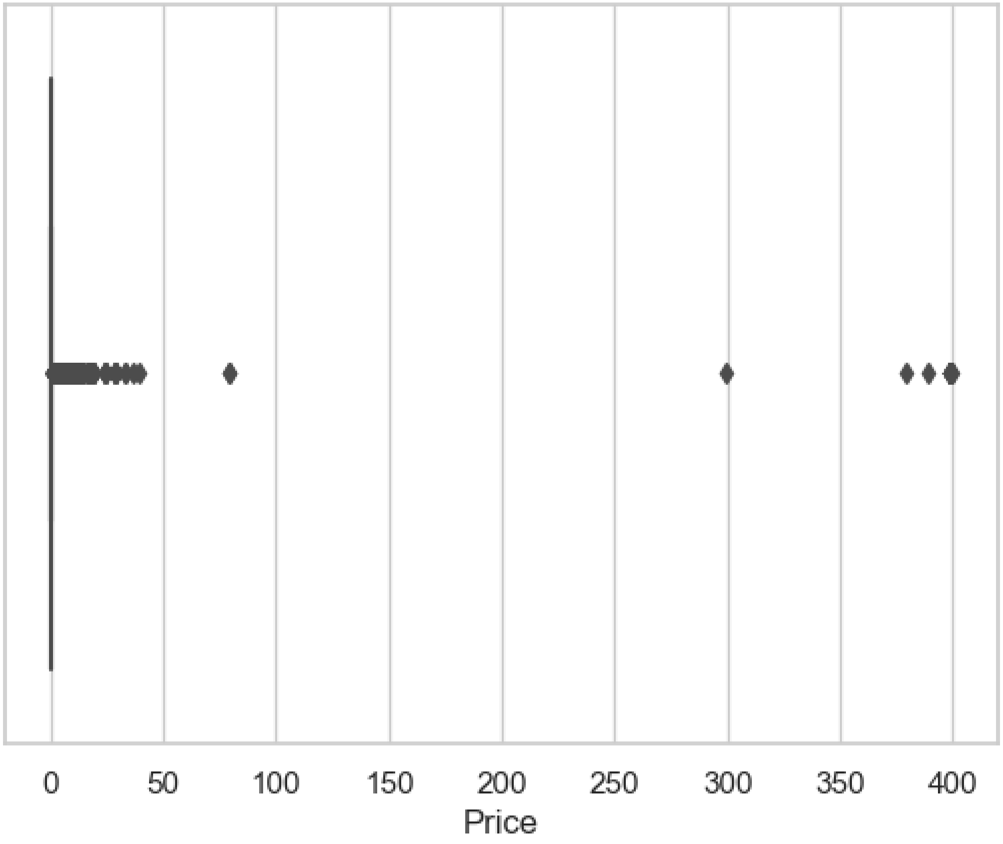
	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	(
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19000.0	10000	Free	0	Everyone	Art &
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000	Free	0	Everyone	Design;F
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000	Free	0	Everyone	Art &
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25000.0	50000000	Free	0	Teen	Art &
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000	Free	0	Everyone	Design;Cr

```
In [43]: sns.set(style='whitegrid')
```

```
In [44]: # Box Plot for Price
         sns.boxplot(data['Price'])
```

```
C:\Users\Rakesh\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

```
Out[44]: <AxesSubplot:xlabel='Price'>
```



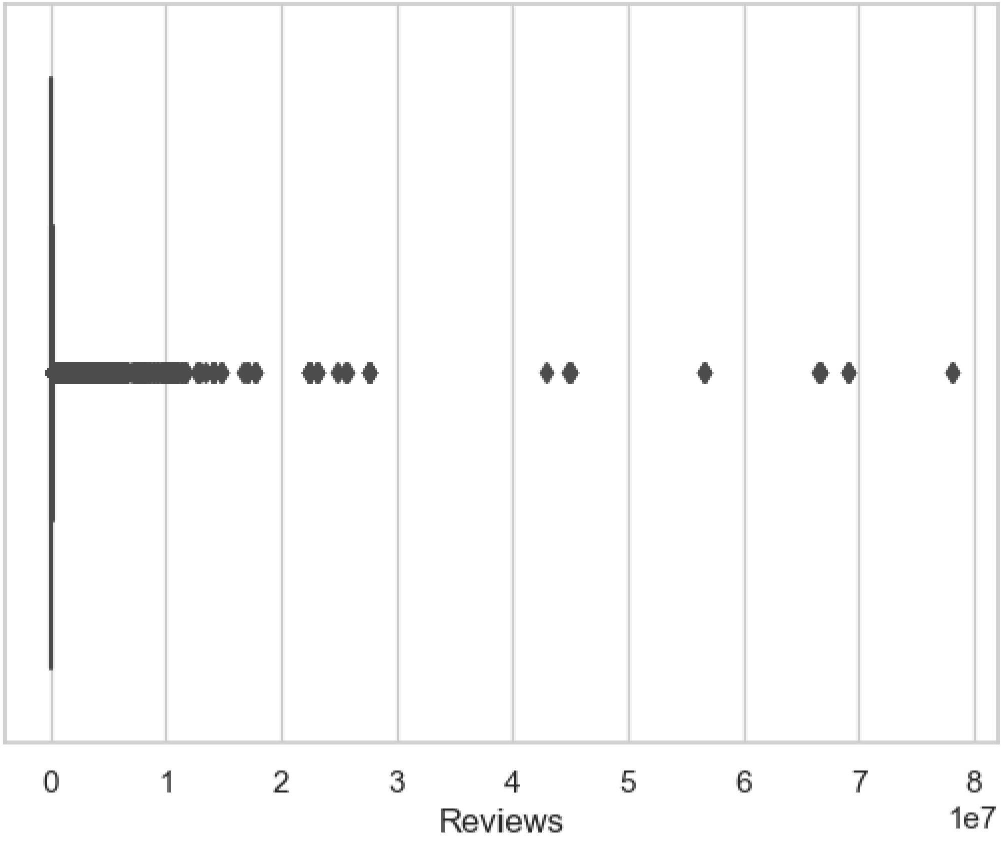
5(2) Boxplot for Reviews

Are there any apps with very high number of reviews? Do the values seem right?

```
In [45]: sns.boxplot(data['Reviews'])
```

```
C:\Users\Rakesh\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only valid po
sitional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```

```
Out[45]: <AxesSubplot:xlabel='Reviews'>
```

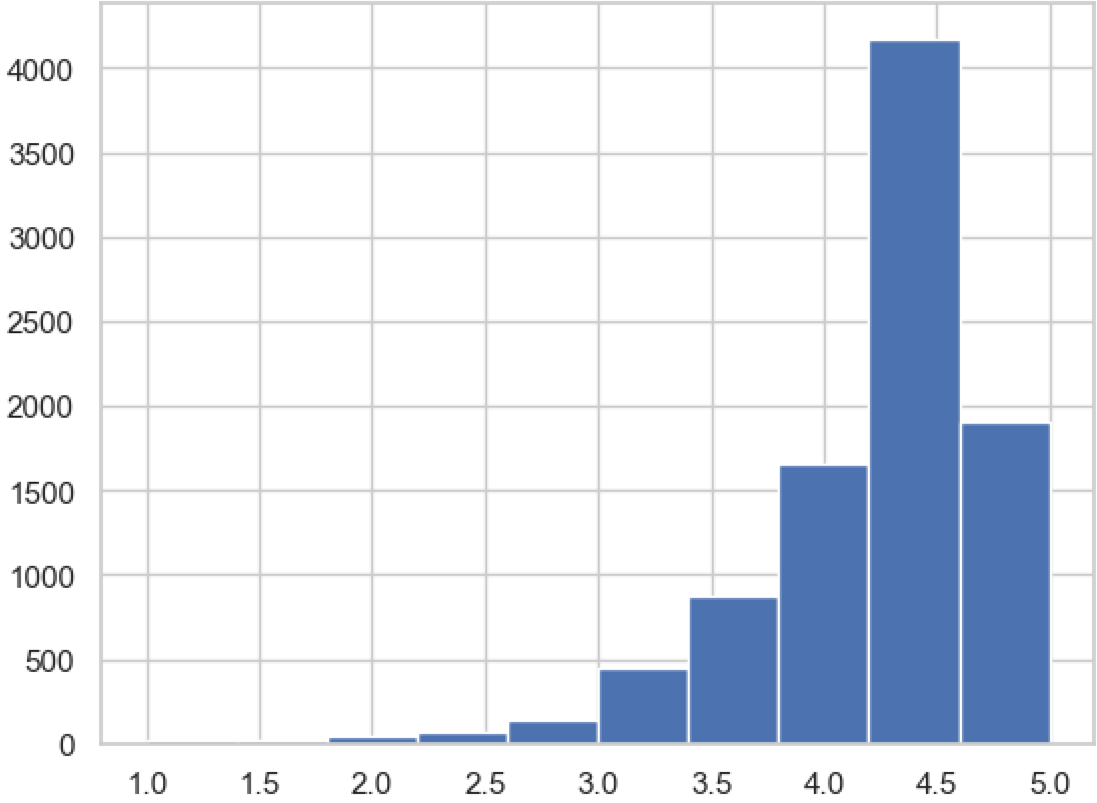


Histogram for Rating

How are the ratings distributed? Is it more toward higher ratings?

In [48]:
```
plt.hist(data['Rating'])
```

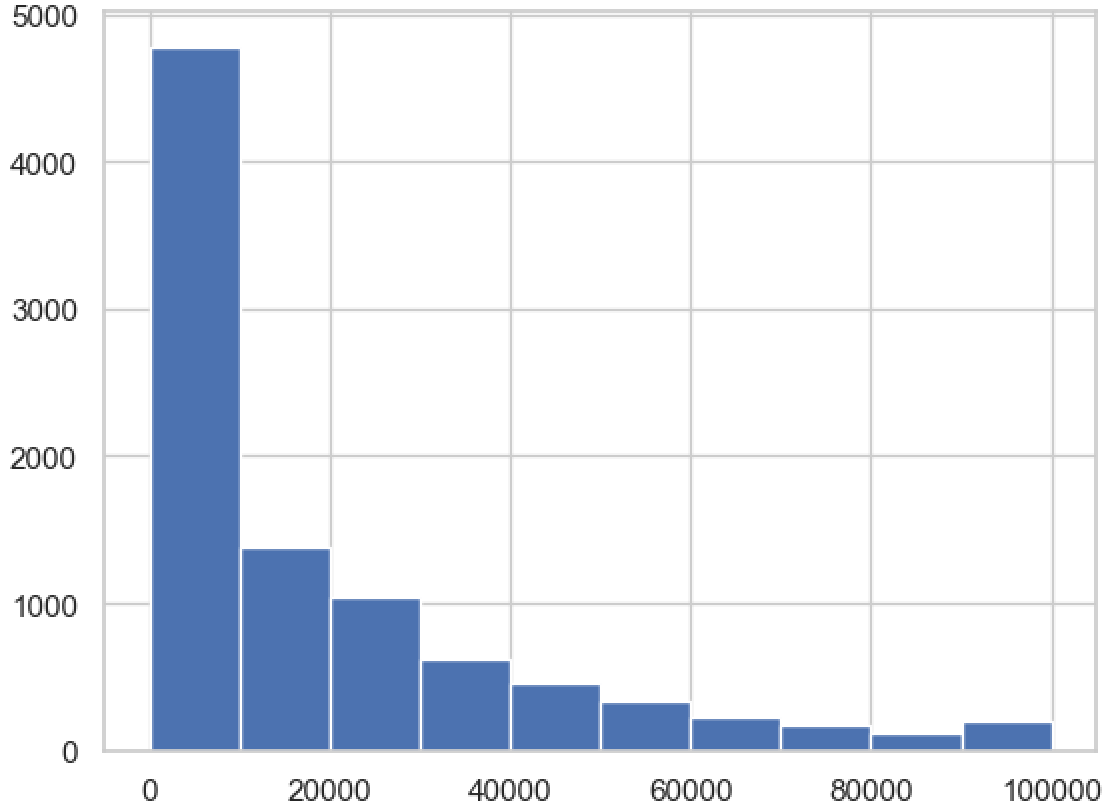
Out[48]:
```
(array([  17.,   18.,   41.,   74.,  137.,  445.,  879., 1660., 4172.,
        1910.]),
 array([1. , 1.4, 1.8, 2.2, 2.6, 3. , 3.4, 3.8, 4.2, 4.6, 5. ]),
 <BarContainer object of 10 artists>)
```



5(4) Histogram for Size

```
In [49]: plt.hist(data['Size'])
```

```
Out[49]: (array([4779., 1386., 1036.,  617.,  464.,  334.,  234.,  174.,  125.,
                 204.]),
          array([     0.,  10000.,  20000.,  30000.,  40000.,  50000.,  60000.,
                 70000.,  80000.,  90000., 100000.]),
          <BarContainer object of 10 artists>)
```



6. Outlier treatment:

6(1) Price: From the box plot, it seems like there are some apps with very high price.

A price of $200 for an application on the Play Store is very high and suspicious!

6(I) Check out the records with very high price

Is 200 indeed a high price?

```
In [50]: data[data['Price']>200].index.shape[0]
Out[50]: 15
```

6(II) Drop these as most seem to be junk apps

```
In [51]: data.drop(data[data['Price']>200].index, inplace=True)
```

```
In [52]: data.shape
Out[52]: (9338, 13)
```

6(2) Reviews: Very few apps have very high number of reviews. These are all star apps that don't help with the analysis and, in fact, will skew it.

Drop records having more than 2 million reviews.

In [53]:

```
data.drop(data[data['Reviews'] > 2000000].index, inplace = True)
```

In [54]:

```
data.shape
```

Out[54]:

```
(8885, 13)
```

6(3)Installs: There seems to be some outliers in this field too.

Apps having very high number of installs should be dropped from the analysis.

Find out the different percentiles – 10, 25, 50, 70, 90, 95, 99

In [55]:

```
data.quantile([.1, .25, .5, .70, .90, .95, .99], axis = 0)
```

Out[55]:

	Rating	Reviews	Size	Installs	Price
0.10	3.5	18.00	0.0	1000.0	0.0
0.25	4.0	159.00	2600.0	10000.0	0.0
0.50	4.3	4290.00	9500.0	500000.0	0.0
0.70	4.5	35930.40	23000.0	1000000.0	0.0
0.90	4.7	296771.00	50000.0	10000000.0	0.0
0.95	4.8	637298.00	68000.0	10000000.0	1.0
0.99	5.0	1462800.88	95000.0	100000000.0	7.0

6(II) Decide a threshold as cutoff for outlier and drop records having values more than that

In [56]:

```
data.drop(data[data['Installs'] > 10000000].index, inplace = True)
```

In [57]:

```
data.shape
```

Out[57]:

```
(8496, 13)
```

7. Bivariate analysis: Let's look at how the available predictors relate to the variable of interest, i.e., our target variable rating.

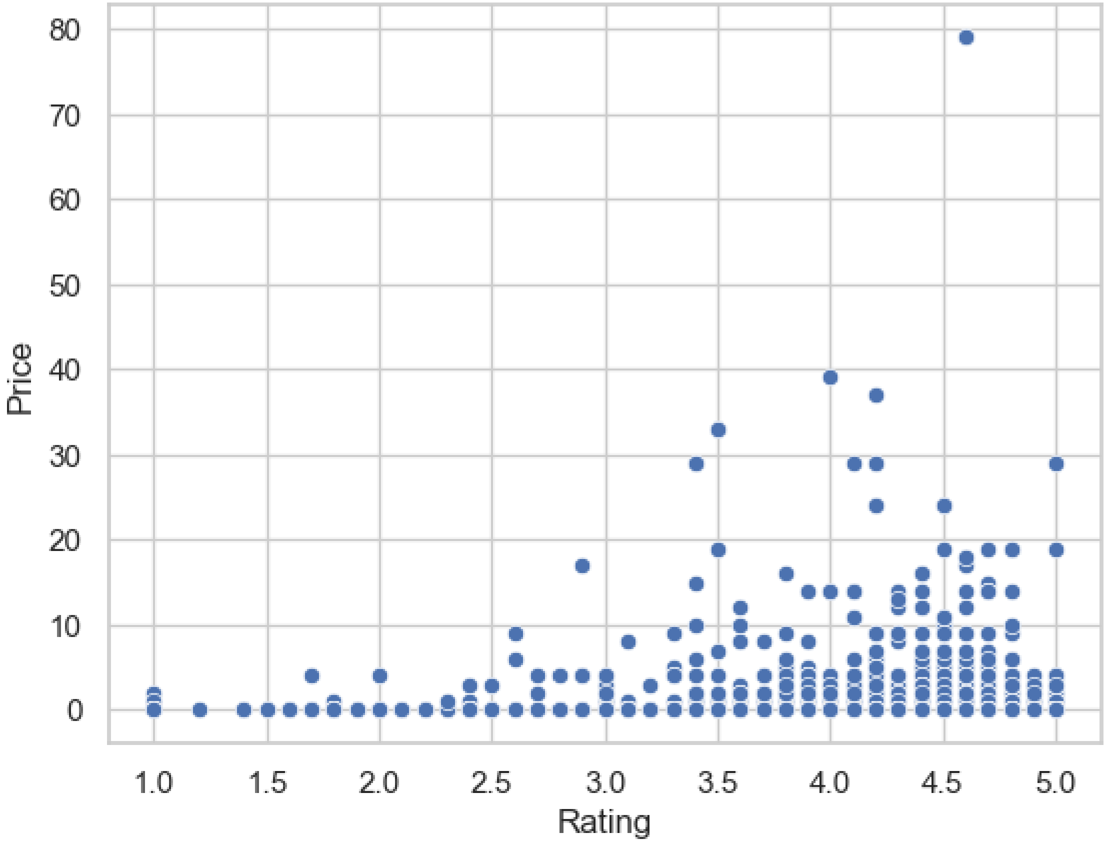
Make scatter plots (for numeric features) and box plots (for character features) to assess the relations between rating and the other features.

Make scatter plot/joinplot for Rating vs. Price

What pattern do you observe? Does rating increase with price?

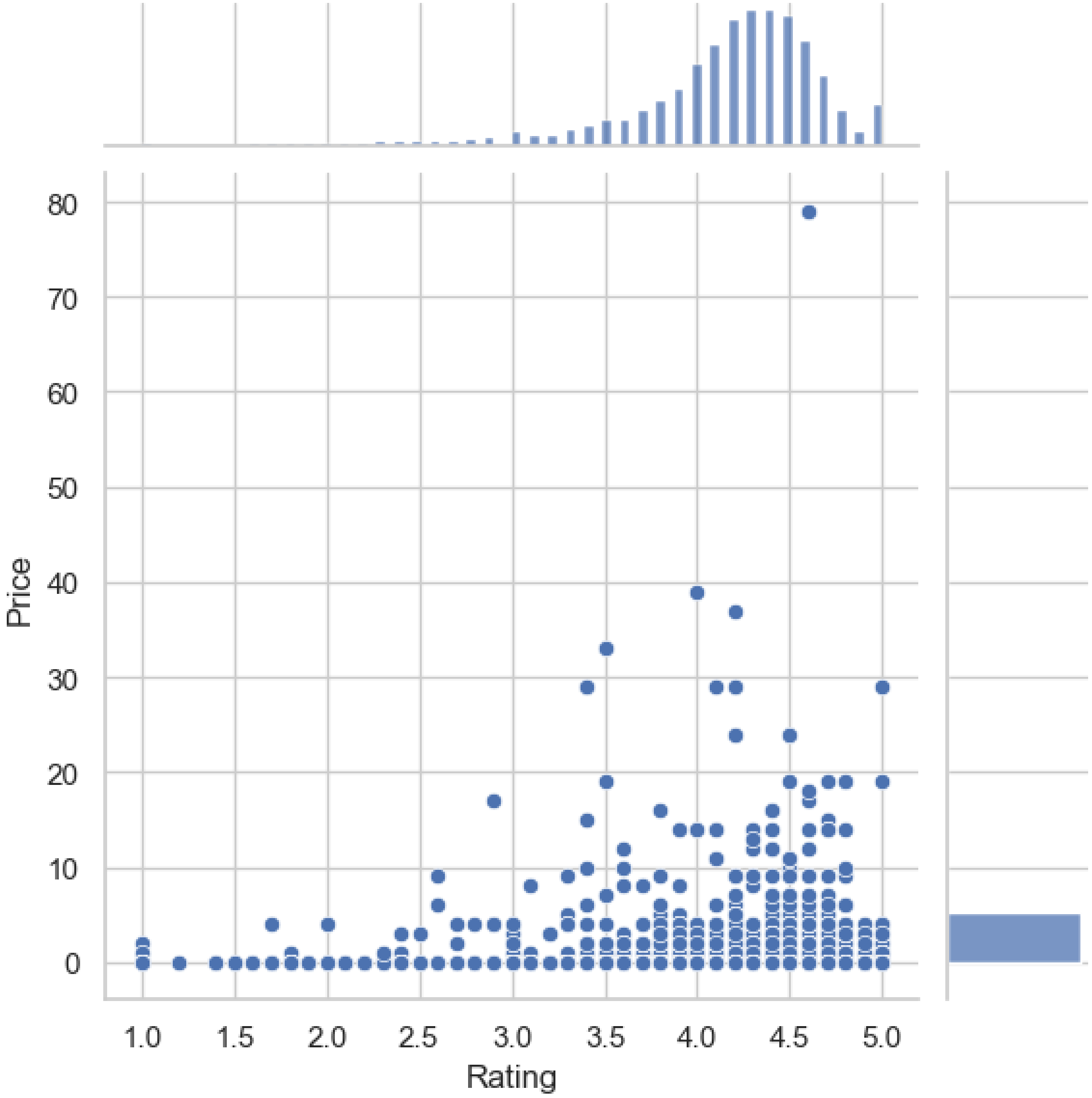
```
In [58]: sns.scatterplot(x='Rating', y='Price', data=data)
```

```
Out[58]: <AxesSubplot:xlabel='Rating', ylabel='Price'>
```



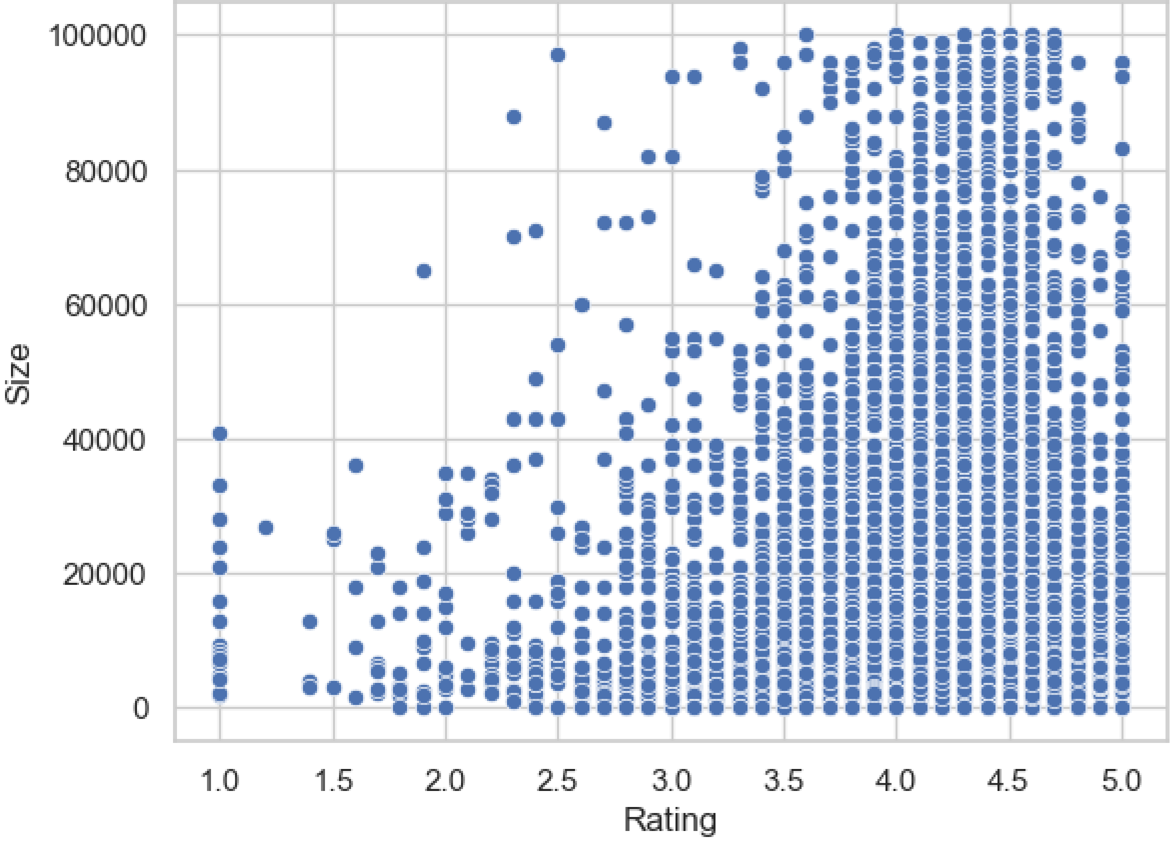
```
In [59]: sns.jointplot(x= 'Rating',y= 'Price',data= data)
```

```
Out[59]: <seaborn.axisgrid.JointGrid at 0x2063f2c22e0>
```



In [60]:
```
sns.scatterplot(x= 'Rating',y= 'Size', data = data)
```

Out[60]:
```
<AxesSubplot:xlabel='Rating', ylabel='Size'>
```

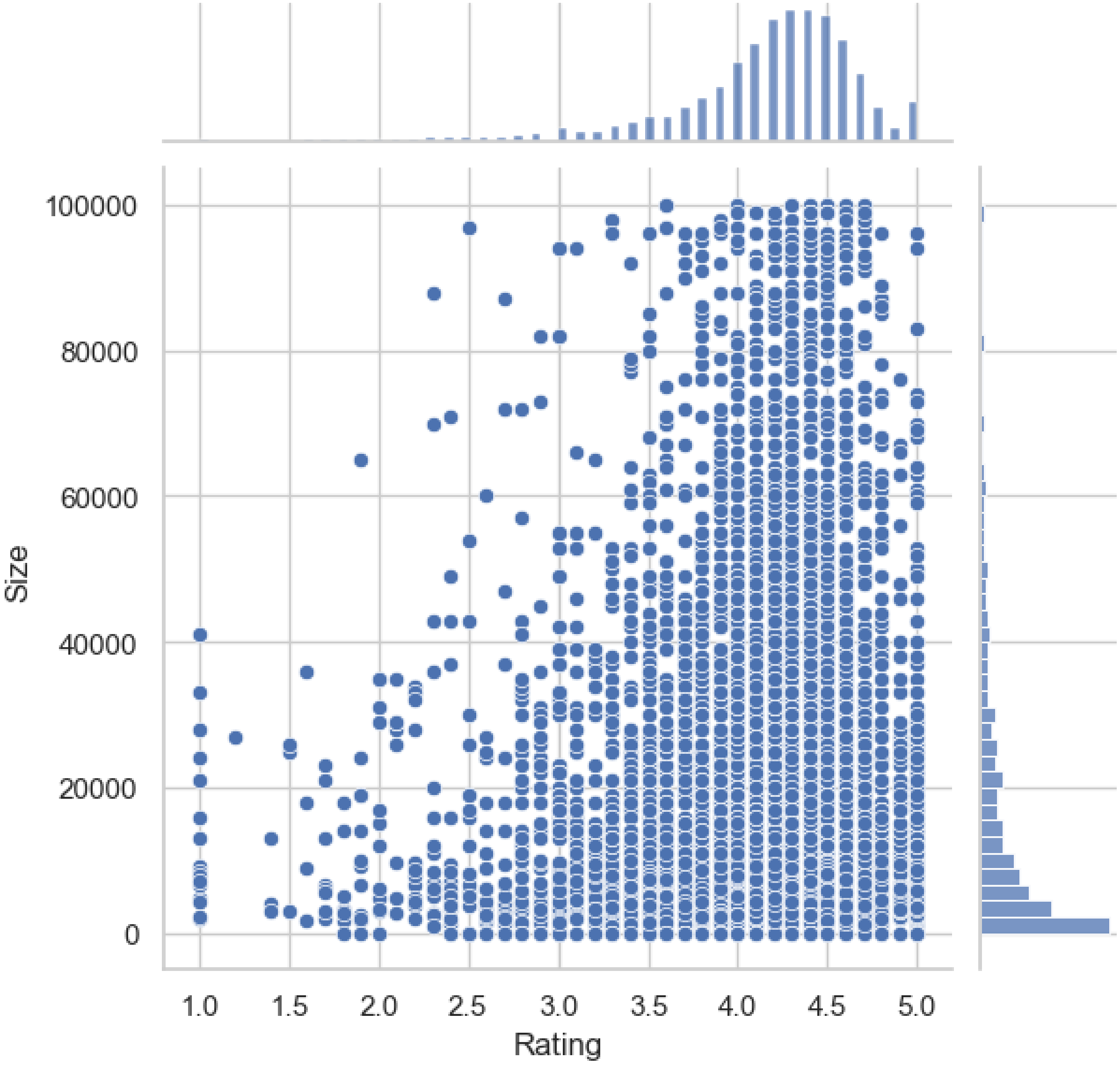


Make scatter plot/joinplot for Rating vs. Size

Are heavier apps rated better?

In [61]:
```
sns.jointplot(x= 'Rating', y= 'Size', data = data)
```

Out[61]:
```
<seaborn.axisgrid.JointGrid at 0x2063f679d60>
```

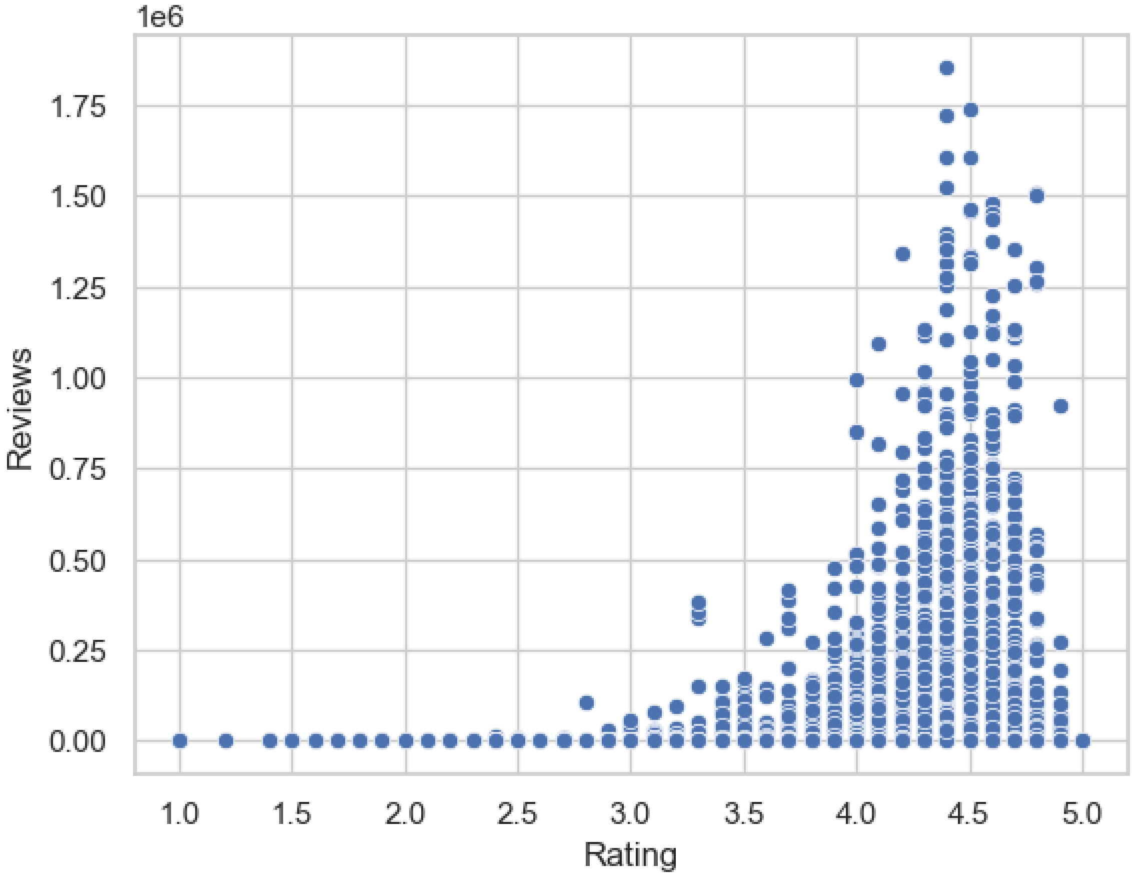


Make scatter plot/joinplot for Rating vs. Reviews

Does more review mean a better rating always?

In [62]:
```
sns.scatterplot(x= 'Rating',y= 'Reviews', data = data)
```

Out[62]:
```
<AxesSubplot:xlabel='Rating', ylabel='Reviews'>
```

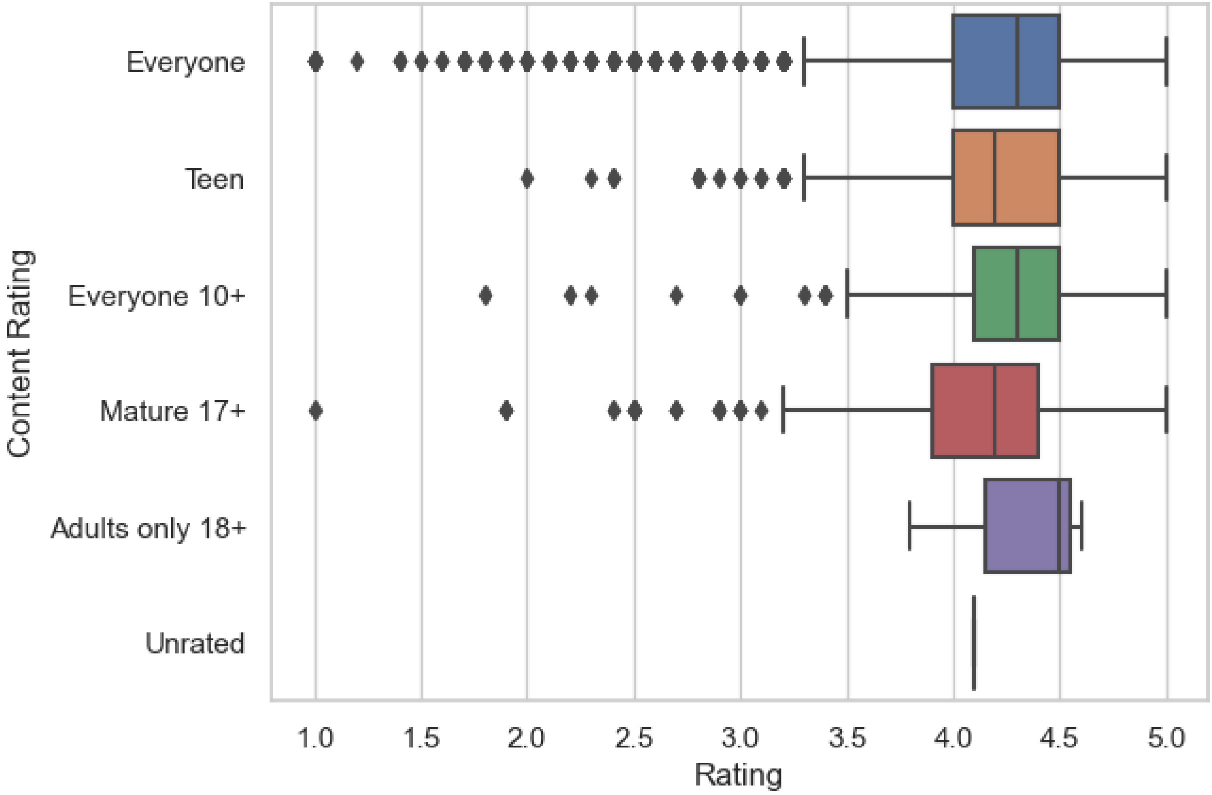


Make boxplot for Rating vs. Content Rating

Is there any difference in the ratings? Are some types liked better?

```
In [63]: sns.boxplot(x= 'Rating', y= 'Content Rating', data = data)
```

```
Out[63]: <AxesSubplot:xlabel='Rating', ylabel='Content Rating'>
```

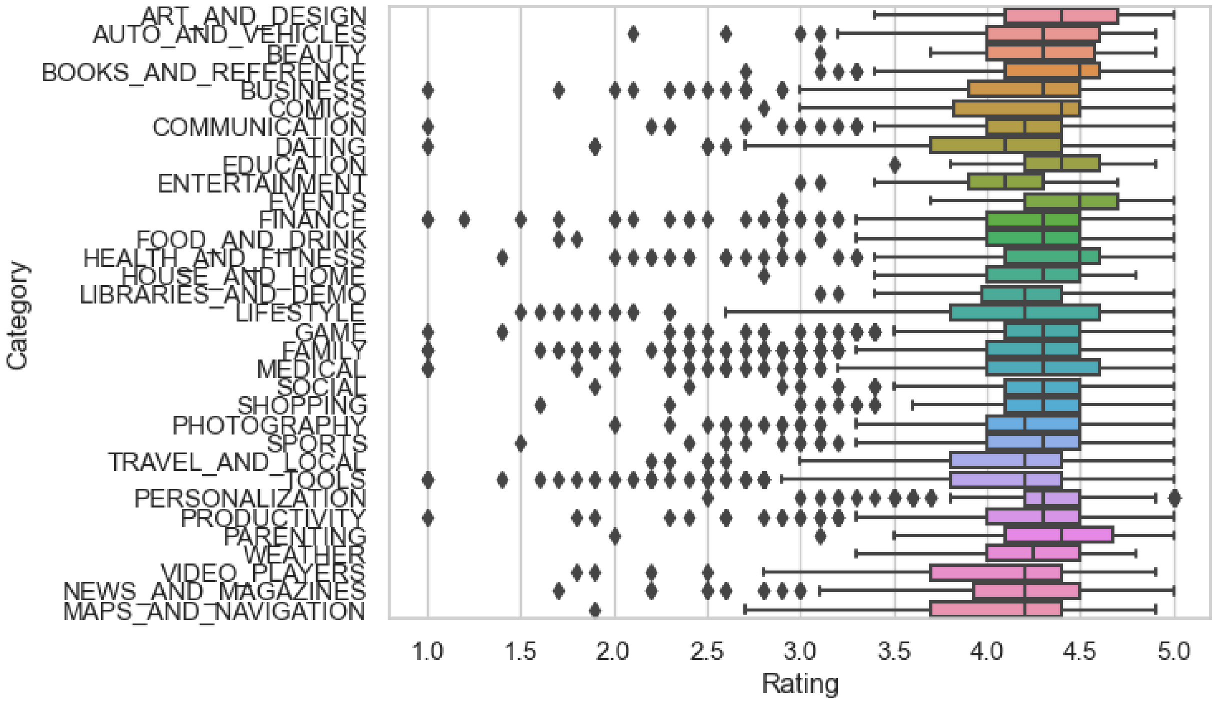


Make boxplot for Ratings vs. Category

Which genre has the best ratings?

In [64]:
```
sns.boxplot(x= 'Rating', y= 'Category', data = data)
```

Out[64]:
```
<AxesSubplot:xlabel='Rating', ylabel='Category'>
```



8. Data preprocessing

For the steps below, create a copy of the dataframe to make all the edits. Name it inp1.

(1) Reviews and Install have some values that are still relatively very high.

Before building a linear regression model, you need to reduce the skew.

Apply log transformation (np.log1p) to Reviews and Installs.

```
In [65]: inp1 = data
```

```
In [66]: inp1.head()
```

Out[66]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	G
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19000.0	10000	Free	0	Everyone	Art & D
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000	Free	0	Everyone	Design;P
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000	Free	0	Everyone	Art & D
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000	Free	0	Everyone	Design;Cre
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167.0	5600.0	50000	Free	0	Everyone	Art & D

```
In [67]: inp1.skew()
```

```
C:\Users\Rakesh\AppData\Local\Temp\ipykernel_1348\3545313420.py:1: FutureWarning: Dro
pping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is depre
cated; in a future version this will raise TypeError.  Select only valid columns befo
re calling the reduction.
  inp1.skew()
```

Out[67]:

```
Rating      -1.749753
Reviews      4.576494
Size         1.655917
Installs     1.543697
Price       18.074542
dtype: float64
```

In [68]:

```
reviewskew = np.log1p(inp1['Reviews'])
inp1['Reviews'] = reviewskew
```

In [69]:

```
reviewskew.skew()
```

Out[69]:

```
-0.20039949659264134
```

In [70]:

```
installsskew = np.log1p(inp1['Installs'])
inp1['Installs']
```

Out[70]:

```
0           10000
1          500000
2         5000000
4          100000
5           50000
           ...   
10834         500
10836        5000
10837         100
10839        1000
10840    10000000
Name: Installs, Length: 8496, dtype: int32
```

In [71]:

```
installsskew.skew()
```

Out[71]:

```
-0.5097286542754812
```

In [72]:

```
inp1.head()
```

Out[72]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	5.075174	19000.0	10000	Free	0	Everyone	Art &
1	Coloring book moana	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	Free	0	Everyone	Design;
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	Free	0	Everyone	Art &
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	Free	0	Everyone	Design;C
5	Paper flowers instructions	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	Free	0	Everyone	Art &

8(2) Drop columns App, Last Updated, Current Ver, and Android Ver. These variables are not useful for our task.

In [73]:

```
inp1.drop(["Last Updated","Current Ver","Android Ver","App","Type"],axis=1,inplace=Tru
```

In [74]:

```
inp1.head()
```

Out[74]:

	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres
0	ART_AND_DESIGN	4.1	5.075174	19000.0	10000	0	Everyone	Art & Design
1	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	0	Everyone	Art & Design;Pretend Play
2	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	0	Everyone	Art & Design
4	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	0	Everyone	Art & Design;Creativity
5	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	0	Everyone	Art & Design

In [75]:

```
inp1.shape
```

Out[75]:

```
(8496, 8)
```

8(3) Get dummy columns for Category, Genres, and Content Rating.

This needs to be done as the models do not understand categorical data, and all data should be numeric.

Dummy encoding is one way to convert character fields to numeric.

Name of dataframe should be inp2.

In [76]:

```
inp2 = inp1
```

In [77]:

```
inp2.head()
```

Out[77]:

	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres
0	ART_AND_DESIGN	4.1	5.075174	19000.0	10000	0	Everyone	Art & Design
1	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	0	Everyone	Art & Design;Pretend Play
2	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	0	Everyone	Art & Design
4	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	0	Everyone	Art & Design;Creativity
5	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	0	Everyone	Art & Design

In [78]:

```
###  Dummy EnCoding on Column "Category"
#get unique values in Column "Category"
inp2.Category.unique()
```

Out[78]:

```
array(['ART_AND_DESIGN', 'AUTO_AND_VEHICLES', 'BEAUTY',
       'BOOKS_AND_REFERENCE', 'BUSINESS', 'COMICS', 'COMMUNICATION',
       'DATING', 'EDUCATION', 'ENTERTAINMENT', 'EVENTS', 'FINANCE',
       'FOOD_AND_DRINK', 'HEALTH_AND_FITNESS', 'HOUSE_AND_HOME',
       'LIBRARIES_AND_DEMO', 'LIFESTYLE', 'GAME', 'FAMILY', 'MEDICAL',
       'SOCIAL', 'SHOPPING', 'PHOTOGRAPHY', 'SPORTS', 'TRAVEL_AND_LOCAL',
       'TOOLS', 'PERSONALIZATION', 'PRODUCTIVITY', 'PARENTING', 'WEATHER',
       'VIDEO_PLAYERS', 'NEWS_AND_MAGAZINES', 'MAPS_AND_NAVIGATION'],
      dtype=object)
```

In [79]:

```
inp2.Category = pd.Categorical(inp2.Category)

x = inp2[['Category']]
del inp2['Category']

dummies = pd.get_dummies(x, prefix = 'Category')
inp2 = pd.concat([inp2,dummies], axis=1)
inp2.head()
```

Out[79]:

	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Category_ART_AND_DESIGN
0	4.1	5.075174	19000.0	10000	0	Everyone	Art & Design	1
1	3.9	6.875232	14000.0	500000	0	Everyone	Art & Design;Pretend Play	1
2	4.7	11.379520	8700.0	5000000	0	Everyone	Art & Design	1
4	4.3	6.875232	2800.0	100000	0	Everyone	Art & Design;Creativity	1
5	4.4	5.123964	5600.0	50000	0	Everyone	Art & Design	1

5 rows × 40 columns

In [80]:

```
inp2.shape
```

Out[80]:

```
(8496, 40)
```

In [81]:

```
### Dummy EnCoding on Column "Genres"
#get unique values in Column "Genres"
inp2["Genres"].unique()
```

Out[81]:

```
array(['Art & Design', 'Art & Design;Pretend Play',
       'Art & Design;Creativity', 'Auto & Vehicles', 'Beauty',
       'Books & Reference', 'Business', 'Comics', 'Comics;Creativity',
       'Communication', 'Dating', 'Education', 'Education;Creativity',
       'Education;Education', 'Education;Music & Video',
       'Education;Action & Adventure', 'Education;Pretend Play',
       'Education;Brain Games', 'Entertainment',
       'Entertainment;Brain Games', 'Entertainment;Creativity',
       'Entertainment;Music & Video', 'Events', 'Finance', 'Food & Drink',
       'Health & Fitness', 'House & Home', 'Libraries & Demo',
       'Lifestyle', 'Lifestyle;Pretend Play', 'Card', 'Casual', 'Puzzle',
       'Action', 'Arcade', 'Word', 'Racing', 'Casual;Creativity',
       'Sports', 'Board', 'Simulation', 'Role Playing', 'Adventure',
       'Strategy', 'Simulation;Education', 'Action;Action & Adventure',
       'Trivia', 'Casual;Brain Games', 'Simulation;Action & Adventure',
       'Educational;Creativity', 'Puzzle;Brain Games',
       'Educational;Education', 'Card;Brain Games',
       'Educational;Brain Games', 'Educational;Pretend Play',
       'Casual;Action & Adventure', 'Entertainment;Education',
       'Casual;Education', 'Casual;Pretend Play', 'Music;Music & Video',
       'Racing;Action & Adventure', 'Arcade;Pretend Play',
       'Adventure;Action & Adventure', 'Role Playing;Action & Adventure',
       'Simulation;Pretend Play', 'Puzzle;Creativity',
       'Sports;Action & Adventure', 'Educational;Action & Adventure',
       'Arcade;Action & Adventure', 'Entertainment;Action & Adventure',
       'Puzzle;Action & Adventure', 'Strategy;Action & Adventure',
       'Music & Audio;Music & Video', 'Health & Fitness;Education',
       'Adventure;Education', 'Board;Brain Games',
       'Board;Action & Adventure', 'Board;Pretend Play',
       'Casual;Music & Video', 'Role Playing;Pretend Play',
       'Entertainment;Pretend Play', 'Video Players & Editors;Creativity',
       'Card;Action & Adventure', 'Medical', 'Social', 'Shopping',
       'Photography', 'Travel & Local',
       'Travel & Local;Action & Adventure', 'Tools', 'Tools;Education',
       'Personalization', 'Productivity', 'Parenting',
       'Parenting;Music & Video', 'Parenting;Brain Games',
       'Parenting;Education', 'Weather', 'Video Players & Editors',
       'Video Players & Editors;Music & Video', 'News & Magazines',
       'Maps & Navigation', 'Health & Fitness;Action & Adventure',
       'Music', 'Educational', 'Casino', 'Adventure;Brain Games',
       'Lifestyle;Education', 'Books & Reference;Education',
       'Puzzle;Education', 'Role Playing;Brain Games',
       'Strategy;Education', 'Racing;Pretend Play',
       'Communication;Creativity', 'Strategy;Creativity'], dtype=object)
```

In [82]:

```
lists = []
for i in inp2.Genres.value_counts().index:
    if inp2.Genres.value_counts()[i]<20:
        lists.append(i)
inp2.Genres = ['Other' if i in lists else i for i in inp2.Genres]
```

In [83]:

```
inp2["Genres"].unique()
```

Out[83]:
```
array(['Art & Design', 'Other', 'Auto & Vehicles', 'Beauty',
       'Books & Reference', 'Business', 'Comics', 'Communication',
       'Dating', 'Education', 'Education;Education',
       'Education;Pretend Play', 'Entertainment',
       'Entertainment;Music & Video', 'Events', 'Finance', 'Food & Drink',
       'Health & Fitness', 'House & Home', 'Libraries & Demo',
       'Lifestyle', 'Card', 'Casual', 'Puzzle', 'Action', 'Arcade',
       'Word', 'Racing', 'Sports', 'Board', 'Simulation', 'Role Playing',
       'Adventure', 'Strategy', 'Trivia', 'Educational;Education',
       'Casual;Pretend Play', 'Medical', 'Social', 'Shopping',
       'Photography', 'Travel & Local', 'Tools', 'Personalization',
       'Productivity', 'Parenting', 'Weather', 'Video Players & Editors',
       'News & Magazines', 'Maps & Navigation', 'Educational', 'Casino'],
      dtype=object)
```

In [84]:
```python
inp2.Genres = pd.Categorical(inp2['Genres'])
x = inp2[["Genres"]]
del inp2['Genres']
dummies = pd.get_dummies(x, prefix = 'Genres')
inp2 = pd.concat([inp2,dummies], axis=1)
```

In [85]:
```python
inp2.head()
```

Out[85]:

	Rating	Reviews	Size	Installs	Price	Content Rating	Category_ART_AND_DESIGN	Category_AUTO_
0	4.1	5.075174	19000.0	10000	0	Everyone	1	
1	3.9	6.875232	14000.0	500000	0	Everyone	1	
2	4.7	11.379520	8700.0	5000000	0	Everyone	1	
4	4.3	6.875232	2800.0	100000	0	Everyone	1	
5	4.4	5.123964	5600.0	50000	0	Everyone	1	

5 rows × 91 columns

In [86]:
```python
inp2.shape
```

Out[86]:
```
(8496, 91)
```

In [87]:
```python
### Dummy EnCoding on Column "Content Rating
#get unique values in Column "Content Rating"
inp2["Content Rating"].unique()
```

Out[87]:
```
array(['Everyone', 'Teen', 'Everyone 10+', 'Mature 17+',
       'Adults only 18+', 'Unrated'], dtype=object)
```

In [88]:
```python
inp2['Content Rating'] = pd.Categorical(inp2['Content Rating'])

x = inp2[['Content Rating']]
del inp2['Content Rating']

dummies = pd.get_dummies(x, prefix = 'Content Rating')
inp2 = pd.concat([inp2,dummies], axis=1)
inp2.head()
```

Out[88]:

	Rating	Reviews	Size	Installs	Price	Category_ART_AND_DESIGN	Category_AUTO_AND_VEHI
0	4.1	5.075174	19000.0	10000	0	1	
1	3.9	6.875232	14000.0	500000	0	1	
2	4.7	11.379520	8700.0	5000000	0	1	
4	4.3	6.875232	2800.0	100000	0	1	
5	4.4	5.123964	5600.0	50000	0	1	

5 rows × 96 columns

9. Train test split and apply 70-30 split. Name the new dataframes df_train and df_test.

10. Separate the dataframes into X_train, y_train, X_test, and y_test.

In [89]:

```
from sklearn.model_selection import train_test_split as tts
from sklearn.linear_model import LinearRegression as LR
from sklearn.metrics import mean_squared_error as mse
```

In [90]:

```
d1 = inp2
X = d1.drop('Rating',axis=1)
y = d1['Rating']

Xtrain, Xtest, ytrain, ytest = tts(X,y, test_size=0.3, random_state=5)
```

11 . Model building

Use linear regression as the technique

Report the R2 on the train set

In [91]:

```
reg_all = LR()
reg_all.fit(Xtrain,ytrain)
```

Out[91]:

```
LinearRegression()
```

In [92]:

```
R2_train = round(reg_all.score(Xtrain,ytrain),3)
print("The R2 value of the Training Set is : {}".format(R2_train))
```

```
The R2 value of the Training Set is : 0.074
```

12. Make predictions on test set and report R2.

In [93]:

```
R2_test = round(reg_all.score(Xtest,ytest),3)
print("The R2 value of the Testing Set is : {}".format(R2_test))
```

```
The R2 value of the Testing Set is : 0.063
```

```
In [ ]:
```