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

In [2]:
```
#Upload data set
data=pd.read_csv('googleplaystore.csv')
```

In [3]:
```
data.shape
```

Out[3]: (10841, 13)

In [4]:
```
#information of data set
data.info
```

Out[4]:
```
<bound method DataFrame.info of
App             Category  \
0         Photo Editor & Candy Camera & Grid & ScrapBook       ART_AND_D
ESIGN
1                                    Coloring book moana       ART_AND_D
ESIGN
2      U Launcher Lite – FREE Live Cool Themes, Hide ...       ART_AND_D
ESIGN
3                                    Sketch - Draw & Paint       ART_AND_D
ESIGN
4                  Pixel Draw - Number Art Coloring Book       ART_AND_D
ESIGN
...                                                  ...
...
10836                                  Sya9a Maroc - FR               F
AMILY
10837                    Fr. Mike Schmitz Audio Teachings               F
AMILY
10838                           Parkinson Exercices FR              ME
DICAL
10839                      The SCP Foundation DB fr nn5n  BOOKS_AND_REFE
RENCE
10840      iHoroscope - 2018 Daily Horoscope & Astrology            LIFE
STYLE

       Rating Reviews                Size     Installs  Type Price  \
0         4.1     159                19M      10,000+  Free     0
1         3.9     967                14M     500,000+  Free     0
2         4.7   87510                8.7M   5,000,000+  Free     0
3         4.5  215644                25M  50,000,000+  Free     0
4         4.3     967                2.8M     100,000+  Free     0
...       ...     ...                 ...          ...   ...    ...
10836     4.5      38                53M       5,000+  Free     0
10837     5.0       4                3.6M        100+  Free     0
10838     NaN       3                9.5M       1,000+  Free     0
10839     4.5     114  Varies with device       1,000+  Free     0
10840     4.5  398307                19M  10,000,000+  Free     0

      Content Rating                     Genres      Last Updated  \
0           Everyone               Art & Design   January 7, 2018
1           Everyone  Art & Design;Pretend Play  January 15, 2018
2           Everyone               Art & Design    August 1, 2018
3               Teen               Art & Design      June 8, 2018
4           Everyone    Art & Design;Creativity     June 20, 2018
...              ...                        ...               ...
10836       Everyone                  Education     July 25, 2017
10837       Everyone                  Education      July 6, 2018
10838       Everyone                    Medical  January 20, 2017
10839     Mature 17+          Books & Reference  January 19, 2015
10840       Everyone                  Lifestyle     July 25, 2018

             Current Ver         Android Ver
0                  1.0.0        4.0.3 and up
```

```
1                  2.0.0        4.0.3 and up
2                  1.2.4        4.0.3 and up
3      Varies with device          4.2 and up
4                    1.1          4.4 and up
...                  ...                 ...
10836               1.48          4.1 and up
10837                1.0          4.1 and up
10838                1.0          2.2 and up
10839  Varies with device  Varies with device
10840  Varies with device  Varies with device

[10841 rows x 13 columns]>
```

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

Out[5]:

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

In [6]:
```
# to check null values in cell
data.isnull().sum()
```

Out[6]:
```
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
```

In [7]:
```
# drop null value
data.dropna(inplace=True)
```

In [8]:
```
data.isnull().sum()
```

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

Convert size column to nummaric and to KB

In [9]:
```
data['Size'].value_counts()
```

Out[9]:
```
Varies with device    1637
14M                    165
12M                    161
15M                    159
11M                    159
                      ...
383k                     1
454k                     1
812k                     1
442k                     1
619k                     1
Name: Size, Length: 413, dtype: int64
```

In [10]:
```
def change(Size):
    if 'M' in Size:
        x=Size[:-1]
        x=float(x)*1000
        return x
    elif 'k' in Size:
        x=Size[:-1]
        x=float(x)
        return x
    else:
        return None
```

In [11]:
```
data['Size']=data['Size'].map(change) # apply and map are same
```

In [12]:
```
data.head()
```

Out[12]:

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

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

In [13]:
```
data['Reviews']=data['Reviews'].astype('int')   #astype convert the data type
```

In [14]:
```
data.dtypes
```

Out[14]:
```
App                object
Category           object
Rating            float64
Reviews             int32
Size              float64
Installs           object
Type               object
Price              object
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

Installs field is currently stored as string and has values like 1,000,000+

Treat 1,000,000+ as 1,000,000

Remove '+', ',' from the field, convert it to integer

In [15]:
```
data['Installs']=data['Installs'].str.replace('[+,]','')
# this will replace both [+and,]in installs column with nothing so we put'' t
# if we need to replace only + we will not put [] only +
```

```
C:\Users\Anjan\AppData\Local\Temp\ipykernel_24240\1336437775.py:1: FutureWa
rning: The default value of regex will change from True to False in a futur
e version.
  data['Installs']=data['Installs'].str.replace('[+,]','')
```

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

Out[16]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0	Everyone
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100000	Free	0	Everyone

In [17]:
```
# Price field is a string and has $ symbol. Remove "$" sign, and convert it t
```

In [18]:
```
data['Price']=data['Price'].str.replace('$','')
```

```
C:\Users\Anjan\AppData\Local\Temp\ipykernel_24240\1843015269.py:1: FutureWa
rning: The default value of regex will change from True to False in a futur
e version. In addition, single character regular expressions will *not* be
treated as literal strings when regex=True.
  data['Price']=data['Price'].str.replace('$','')
```

In [19]:
```
data.head()
```

Out[19]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0	Everyone
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100000	Free	0	Everyone

In [20]:
```
data.dtypes
```

Out[20]:
```
App               object
Category          object
Rating           float64
Reviews            int32
Size             float64
Installs          object
Type              object
Price             object
Content Rating    object
Genres            object
Last Updated      object
Current Ver       object
Android Ver       object
dtype: object
```

In [21]:
```
data['Price']=data['Price'].astype('float')
```

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

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

Sanity checks

In [23]:
```
#Average rating should be between 1 and 5 as only these values are allowed on
#Drop the rows that have a value outside this range.
```

In [24]:
```
# we need to check if there is any data greater than 5
data[data['Rating']>5]
```

Out[24]:

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Cu

In [25]:
```
# we need to check if there is any data less than 1
data[data['Rating']<1]
```

Out[25]:

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Cu

#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 [26]:
```
#to convert installs to int
data['Installs']=data['Installs'].astype('int')
```

In [27]:
```
data.dtypes
```

Out[27]:
```
App                object
Category           object
Rating            float64
Reviews             int32
Size              float64
Installs            int32
Type               object
Price             float64
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

In [28]:
```
#to check reviews are greater than installs, if yes drop
data.drop(data[data['Installs']<data['Reviews']].index, inplace=True)
```

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

In [29]:
```
data.drop(data[(data['Type']=='Free') & (data['Price']>0)].index, inplace= Tr
```

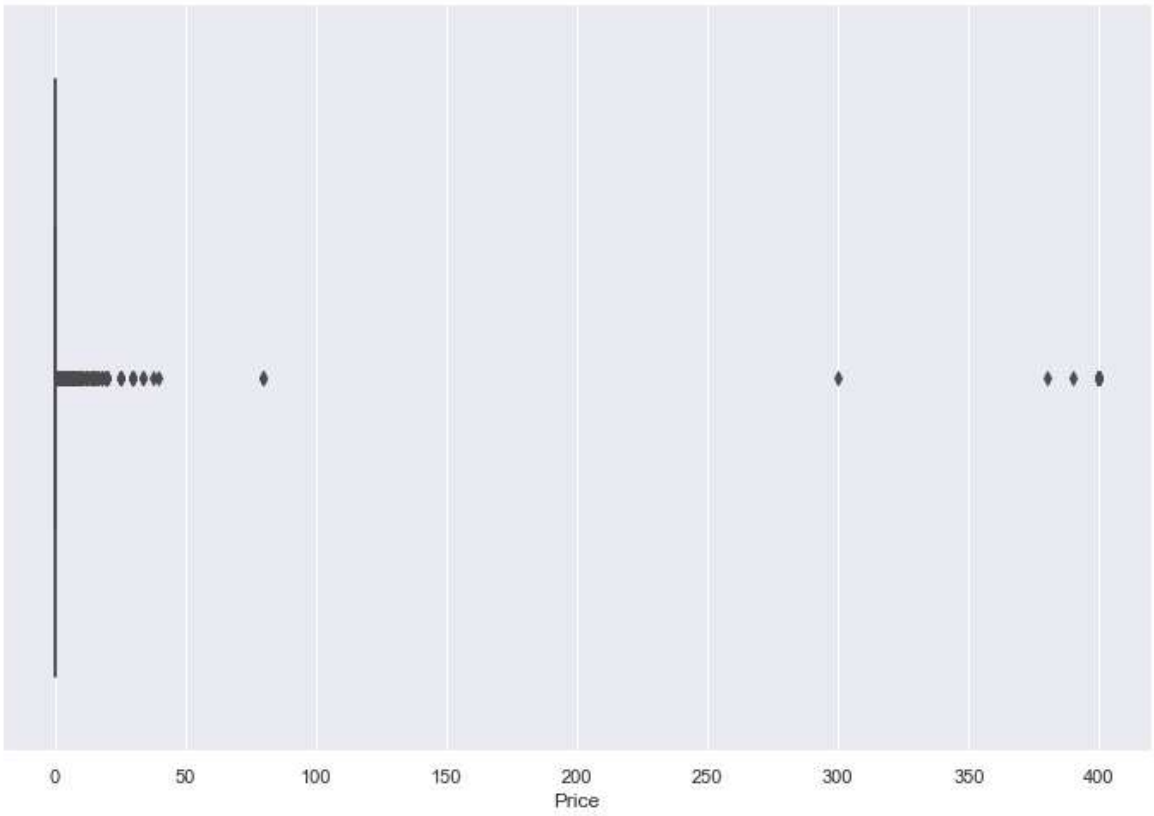
Performing univariate analysis

In [30]:
```
# Boxplot for Price
# Are there any outliers? Think about the price of usual apps on Play Store
```

In [31]:
```
sns.set(rc={'figure.figsize':(12,8)})#for graph size
```

In [32]:
```
sns.boxplot(x='Price',data=data)
```

Out[32]: <AxesSubplot:xlabel='Price'>



In [ ]:
```

```

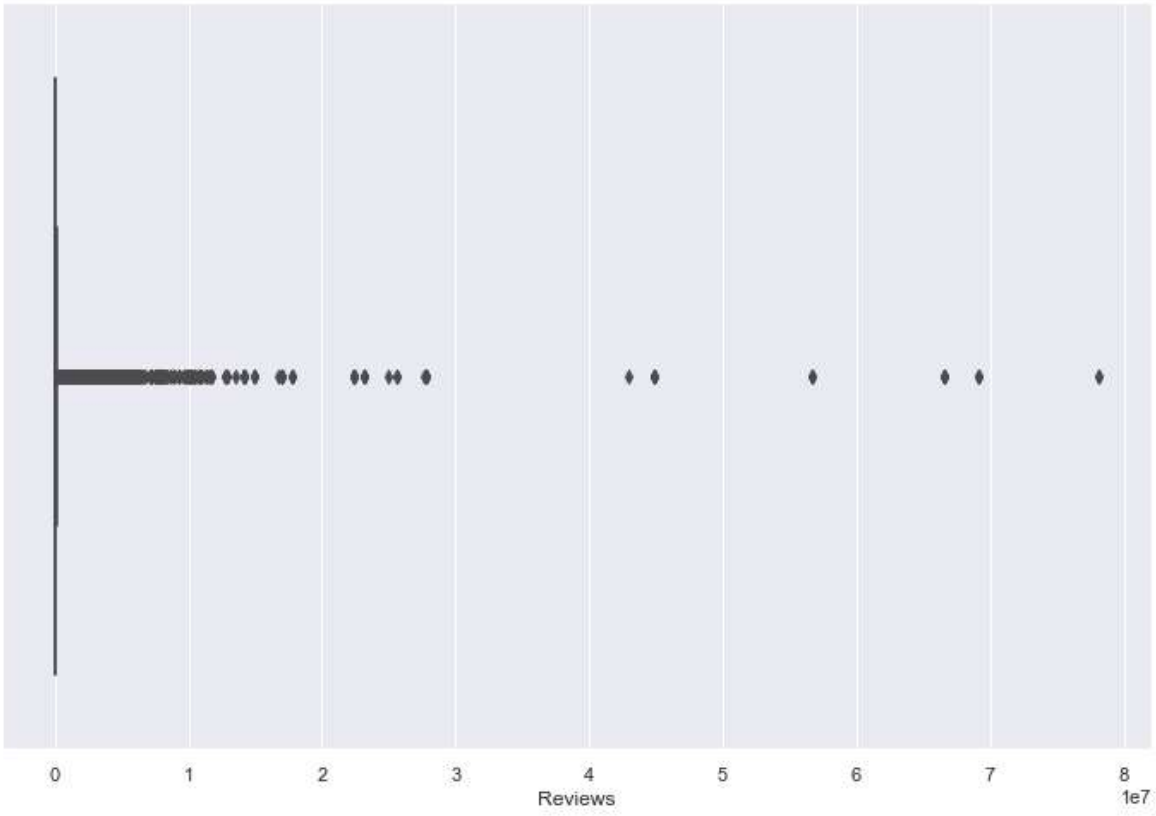
In [33]:
```
#Boxplot for Reviews

#Are there any apps with very high number of reviews? Do the values seem righ
```

In [34]:
```
sns.boxplot(data['Reviews'])
```

```
C:\Users\Anjan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futur
eWarning: Pass the following variable as a keyword arg: x. From version 0.1
2, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpreta
tion.
  warnings.warn(
```

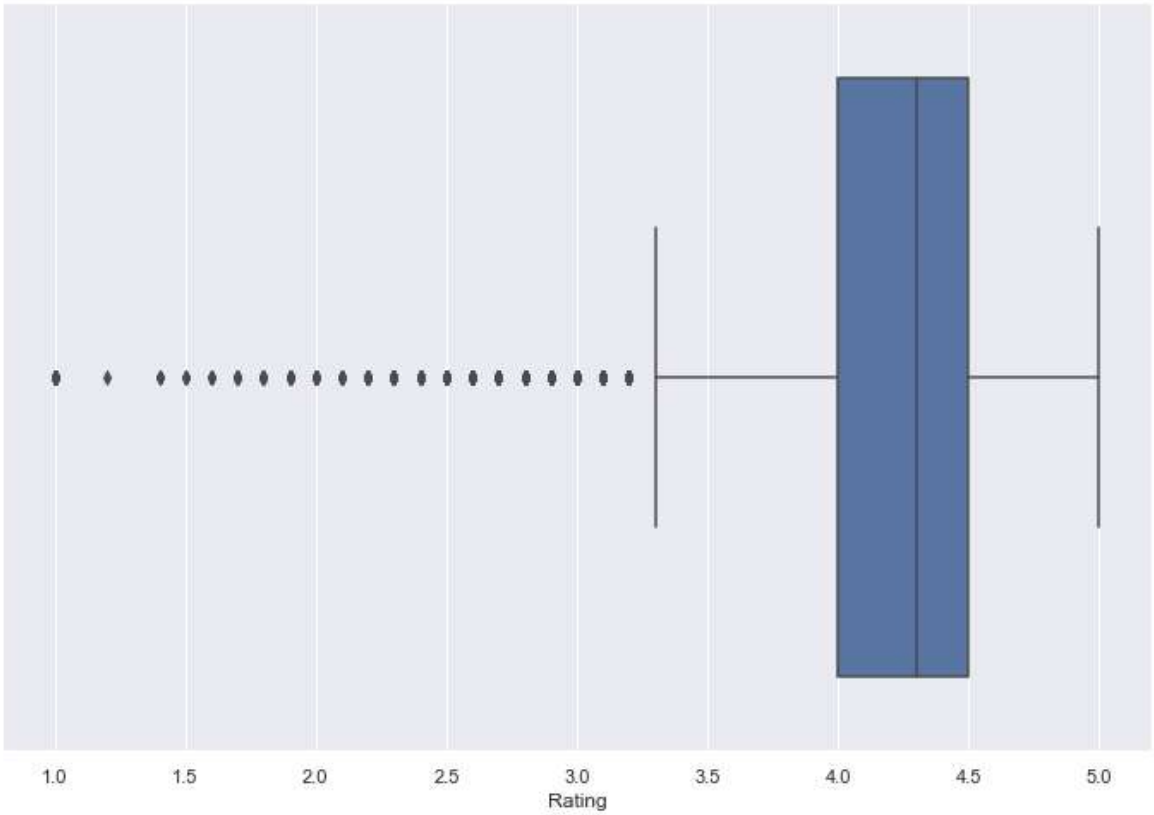
Out[34]: <AxesSubplot:xlabel='Reviews'>



In [35]:
```
sns.boxplot(data['Rating'])
```

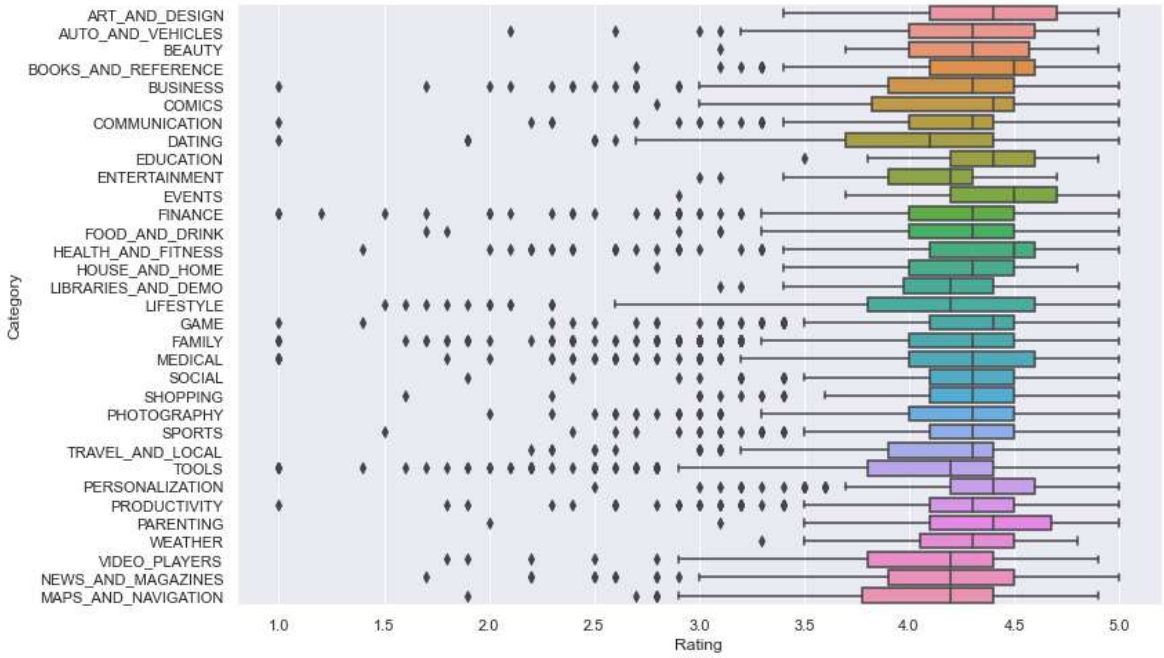
```
C:\Users\Anjan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futur
eWarning: Pass the following variable as a keyword arg: x. From version 0.1
2, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpreta
tion.
  warnings.warn(
```

Out[35]: <AxesSubplot:xlabel='Rating'>



In [36]:
```
sns.boxplot(x="Rating", y="Category", data=data)
```

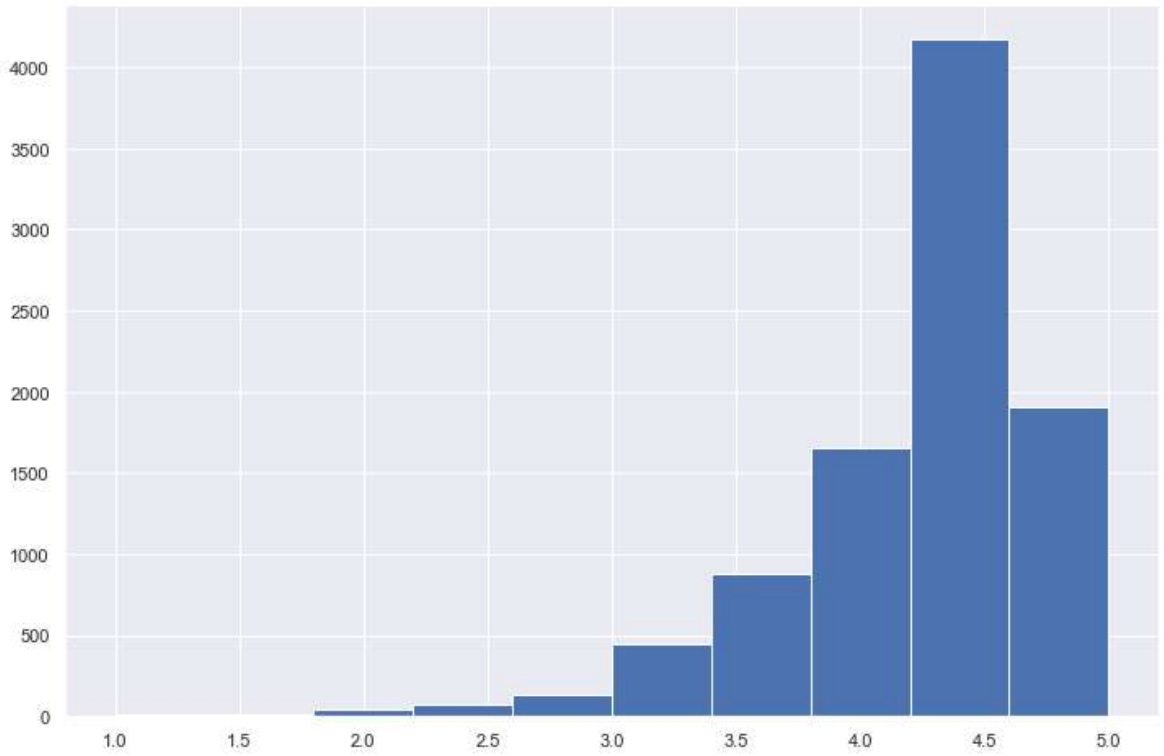
Out[36]: <AxesSubplot:xlabel='Rating', ylabel='Category'>



In [37]:
```
# Histogram for Rating
```

In [38]:
```
plt.hist(data['Rating'])
```

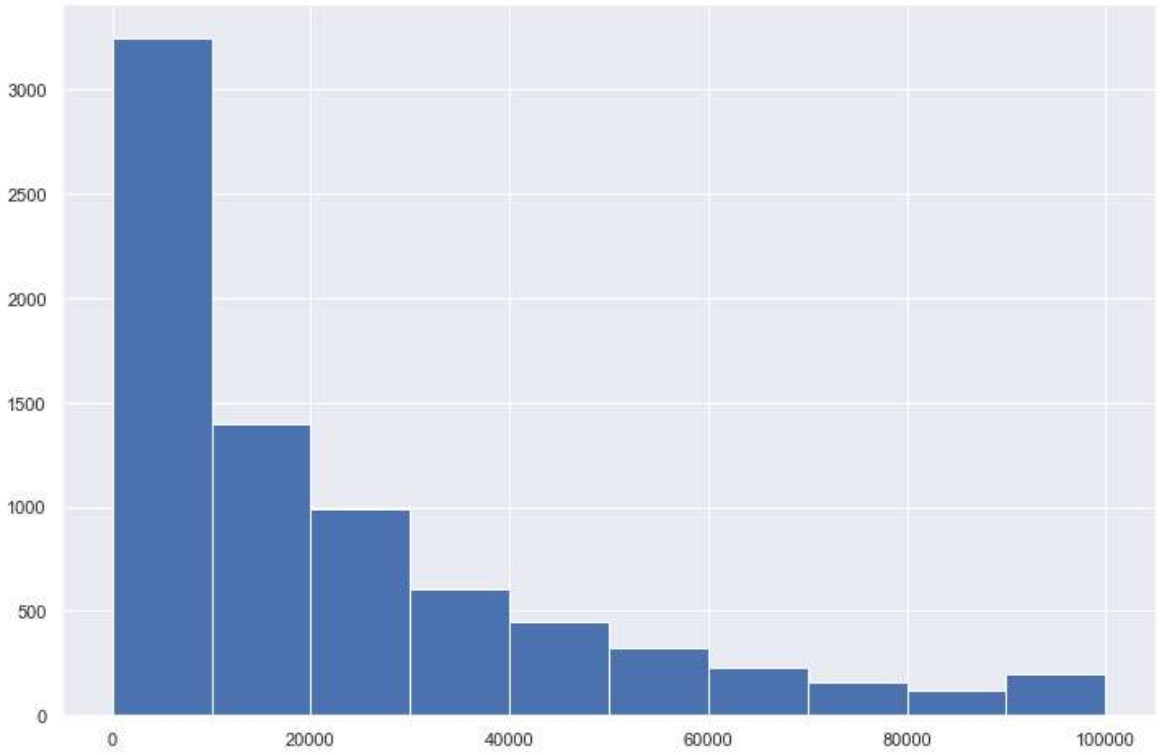
Out[38]:
```
(array([  17.,   18.,   41.,   74.,  137.,  445.,  879., 1660., 4172.,
        1910.]),
 array([1. , 1.4, 1.8, 2.2, 2.6, 3. , 3.4, 3.8, 4.2, 4.6, 5. ]),
 <BarContainer object of 10 artists>)
```



In [39]:
```
# Histogram for Size
```

In [40]:
```
plt.hist(data['Size'])
```

Out[40]:
```
(array([3245., 1398.,  991.,  606.,  449.,  325.,  226.,  161.,  117.,
         199.]),
 array([8.500000e+00, 1.000765e+04, 2.000680e+04, 3.000595e+04,
        4.000510e+04, 5.000425e+04, 6.000340e+04, 7.000255e+04,
        8.000170e+04, 9.000085e+04, 1.000000e+05]),
 <BarContainer object of 10 artists>)
```



Outlier treatment

In [41]:
```
#Price: From the box plot, it seems like there are some apps with very high p
#A price of $200 for an application on the Play Store is very high and suspic

#Check out the records with very high price

#Is 200 indeed a high price?

#Drop these as most seem to be junk apps
```

In [42]:
```
more = data.apply(lambda x : True
            if x['Price'] > 200 else False, axis = 1)
```

In [43]:
```
more_count = len(more[more == True].index)
```

In [44]:
```
data.shape
```

Out[44]: (9353, 13)

In [45]:
```
data.drop(data[data['Price']>200].index, inplace=True)
```

In [46]:
```
data.shape
```

Out[46]: (9338, 13)

In [47]:
```
#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
#Drop records having more than 2 million reviews.
```

In [48]:
```
data.drop(data[data['Reviews']>2000000].index, inplace=True)
```

In [49]:
```
data.shape #reviews more than 2 million are droped.
```

Out[49]: (8885, 13)

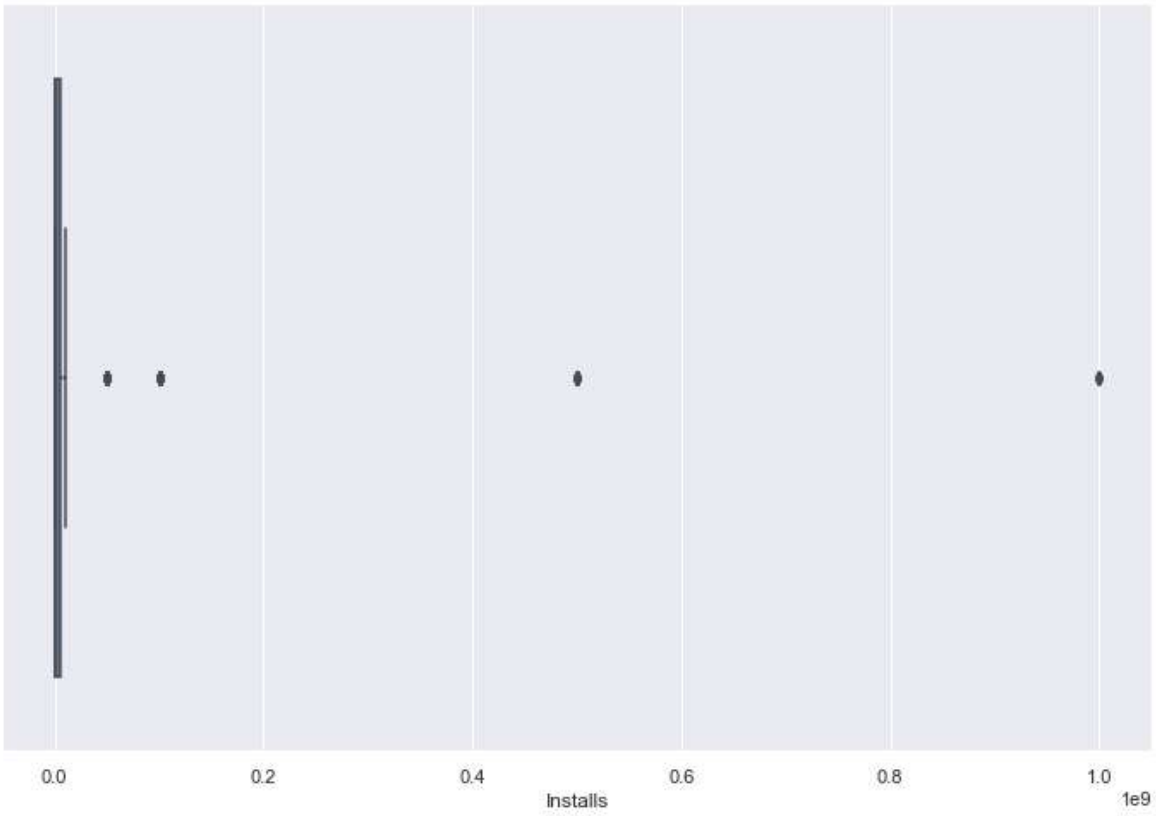
In [50]:
```
#Installs:  There seems to be some outliers in this field too.
#Apps having very high number of installs should be dropped from the analysis

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

#Decide a threshold as cutoff for outlier and drop records having values more
```

In [51]:
```
sns.boxplot(x='Installs',data=data) #boxplot for outliers fo installs.
```

Out[51]: <AxesSubplot:xlabel='Installs'>



In [52]:
```
#a quantile is where a sample is divided into equal-sized, adjacent, subgroup
data.quantile([.1, .25, .5, .70, .90, .95, .99], axis = 0)
```

Out[52]:

	Rating	Reviews	Size	Installs	Price
0.10	3.5	18.00	2500.0	1000.0	0.00
0.25	4.0	159.00	5100.0	10000.0	0.00
0.50	4.3	4290.00	14000.0	500000.0	0.00
0.70	4.5	35930.40	26000.0	1000000.0	0.00
0.90	4.7	296771.00	56000.0	10000000.0	0.00
0.95	4.8	637298.00	72900.0	10000000.0	1.99
0.99	5.0	1462800.88	96000.0	100000000.0	7.49

In [53]:
```
# dropping more than 10000000 Installs value
data.drop(data[data['Installs']>10000000].index, inplace=True)
```

In [54]:
```
data.shape
```

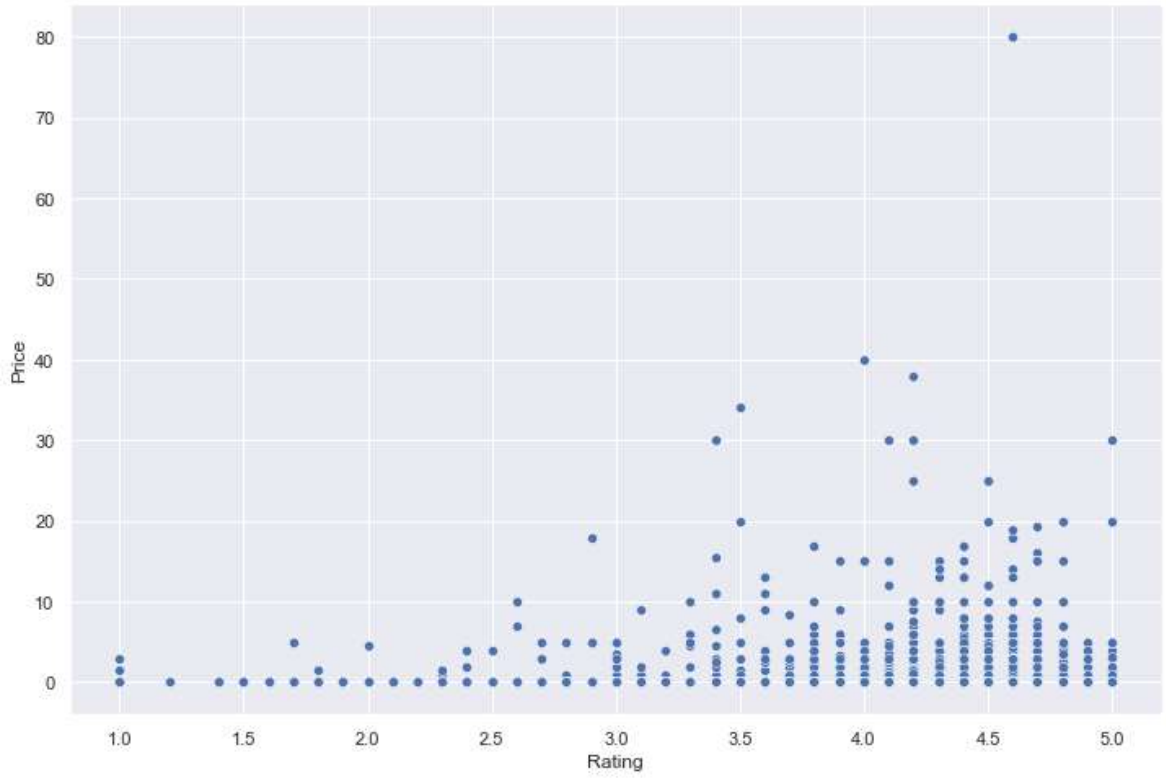
Out[54]: (8496, 13)

Bivariate analysis

In [55]:
```
#Make scatter plot/joinplot for Rating vs. Price
```

In [56]:
```
sns.scatterplot(x='Rating',y='Price',data=data)
```

Out[56]: <AxesSubplot:xlabel='Rating', ylabel='Price'>

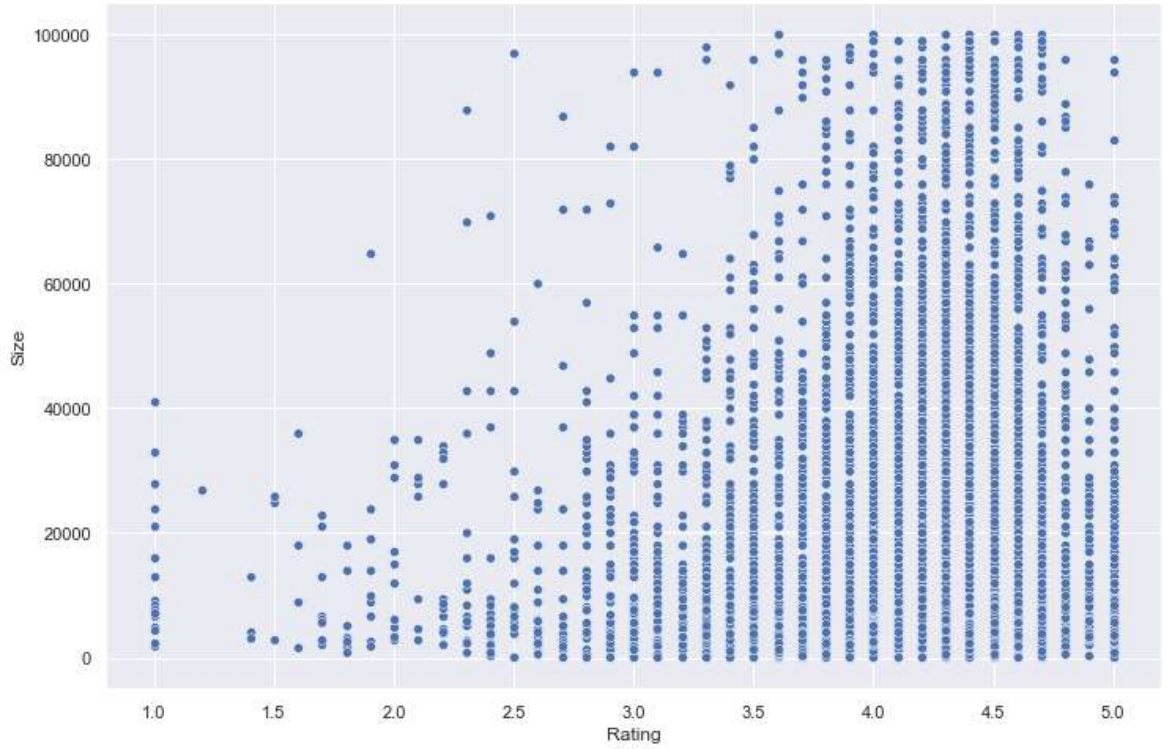


#from above plot we find that rating is more for paid apps.

In [57]:
```
# Make scatter plot/joinplot for Rating vs. Size
```

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

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

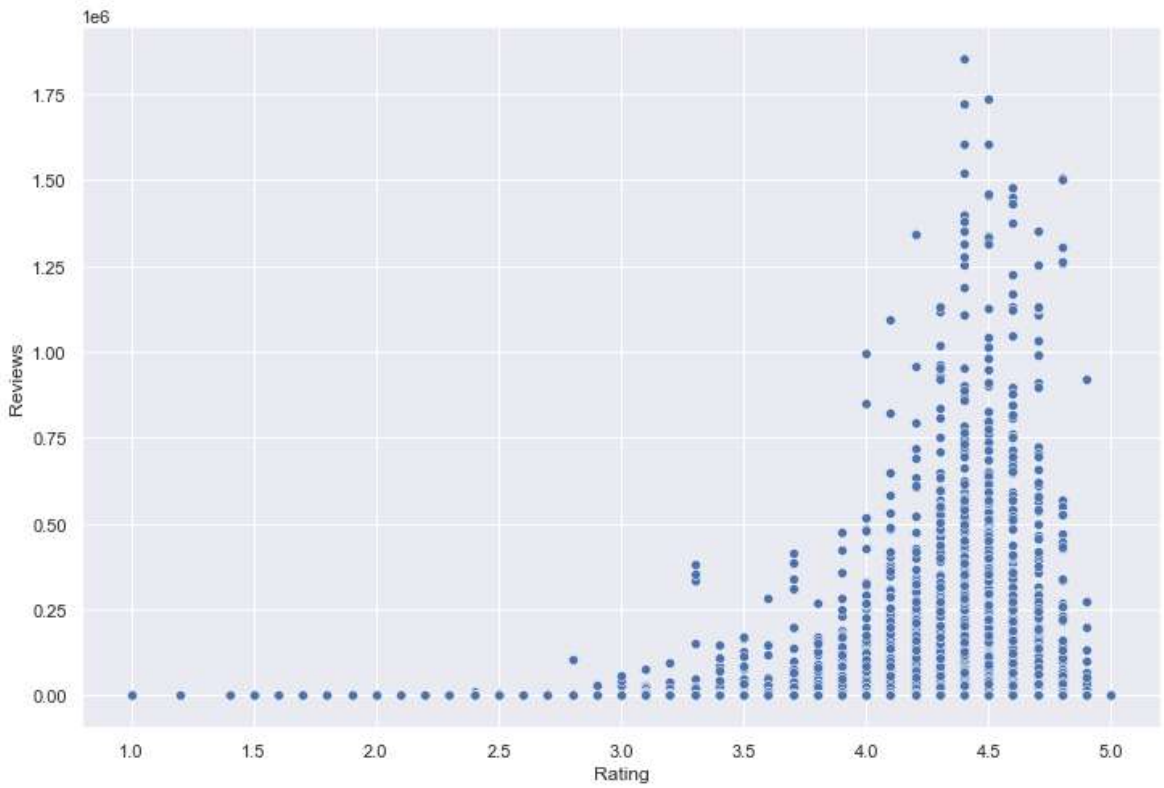


higher the app size better is rating

In [59]:
```
#Make scatter plot/joinplot for Rating vs. Reviews
```

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

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

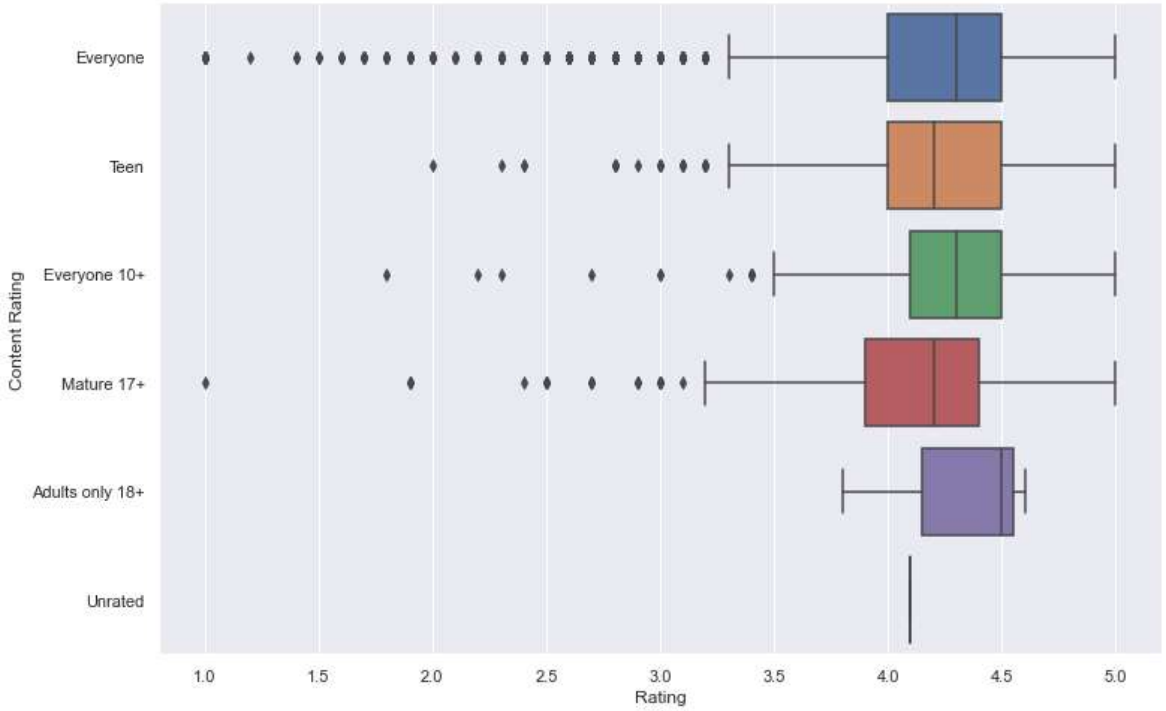


More reviews have higher ratting

In [61]:
```
#Make boxplot for Rating vs. Content Rating
```

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

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

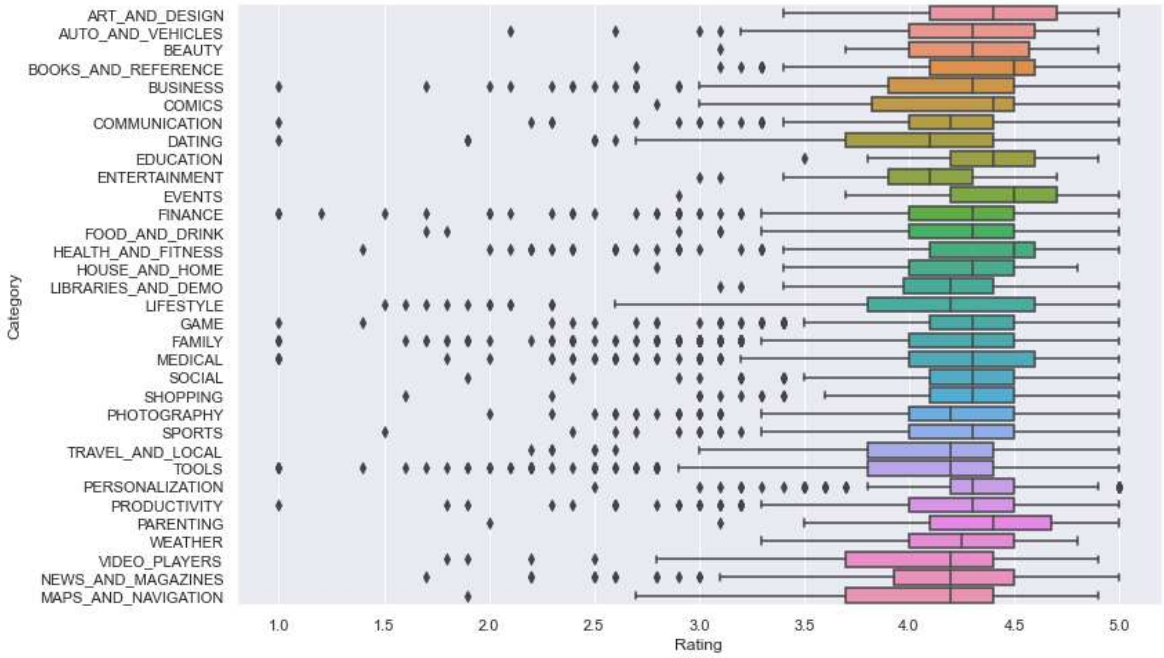


App for Everyone has more bad raing and app for Adult only 18+ has good ratting.

In [63]:
```
#Make boxplot for Rating vs. Category
```

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

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



All most all has good raitting

Data preprocessing

In [65]:
```
# create a copy of the dataframe to make all the edits. Name it inp1
```

In [66]:
```
inp1=data
```

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

Out[67]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0.0	Everyone
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100000	Free	0.0	Everyone
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167	5600.0	50000	Free	0.0	Everyone

In [68]:
```
inp1.skew()
```

```
C:\Users\Anjan\AppData\Local\Temp\ipykernel_24240\3545313420.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_
only=None') is deprecated; in a future version this will raise TypeError.
Select only valid columns before calling the reduction.
  inp1.skew()
```

Out[68]:
```
Rating      -1.749753
Reviews      4.576494
Size         1.533259
Installs     1.543697
Price       16.264811
dtype: float64
```

In [69]:
```
reviewskew = np.log1p(inp1['Reviews']) #Apply log transformation to reduce sk
inp1['Reviews'] = reviewskew
```

In [70]:
```
reviewskew.skew()
```

Out[70]:
```
-0.20039949659264134
```

In [71]:
```
installsskew = np.log1p(inp1['Installs'])
inp1['Installs']
```

Out[71]:
```
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 [72]:
```
installsskew.skew()
```

Out[72]:
```
-0.5097286542754812
```

In [ ]:
```

```

In [73]:
```
inp1.head()
```

Out[73]:

	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	0.0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	Free	0.0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	Free	0.0	Everyone
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	Free	0.0	Everyone
5	Paper flowers instructions	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	Free	0.0	Everyone

In [74]:
```
#Drop columns App, Last Updated, Current Ver, and Android Ver. These variable
```

In [75]:
```
inp1.drop(['App','Last Updated','Current Ver','Android Ver','Type'],axis=1,in
```

In [76]:
```
inp1.head()
```

Out[76]:

	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres
0	ART_AND_DESIGN	4.1	5.075174	19000.0	10000	0.0	Everyone	Art & Design
1	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	0.0	Everyone	Art & Design;Pretend Play
2	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	0.0	Everyone	Art & Design
4	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	0.0	Everyone	Art & Design;Creativity
5	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	0.0	Everyone	Art & Design

In [77]:
```
#Get dummy columns for Category, Genres, and Content Rating.
#This needs to be done as the models do not understand categorical data, and
#Dummy encoding is one way to convert character fields to numeric.
#Name of dataframe should be inp2.
```

In [78]:
```
inp2 = inp1
```

In [79]:
```
inp2.head()
```

Out[79]:

	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres
0	ART_AND_DESIGN	4.1	5.075174	19000.0	10000	0.0	Everyone	Art & Design
1	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	0.0	Everyone	Art & Design;Pretend Play
2	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	0.0	Everyone	Art & Design
4	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	0.0	Everyone	Art & Design;Creativity
5	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	0.0	Everyone	Art & Design

In [80]:
```
# Dummy encoding is one way to convert character fields to numeric for
```

In [81]:
```
# Dummy encoding Category
```

In [82]:
```
#get unique values in Column "Category"
inp2.Category.unique()
```

Out[82]:
```
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 [ ]:
```
```

In [83]:
```
#this is second way to Dummy encoding
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[83]:

	Rating	Reviews	Size	Installs	Price	Content Rating	Genres	Category_ART_AND_I
0	4.1	5.075174	19000.0	10000	0.0	Everyone	Art & Design	
1	3.9	6.875232	14000.0	500000	0.0	Everyone	Art & Design;Pretend Play	
2	4.7	11.379520	8700.0	5000000	0.0	Everyone	Art & Design	
4	4.3	6.875232	2800.0	100000	0.0	Everyone	Art & Design;Creativity	
5	4.4	5.123964	5600.0	50000	0.0	Everyone	Art & Design	

5 rows × 40 columns

In [84]:
```
# Dummy encoding Genres
```

In [85]:
```
#get unique values in Column "Genres"
inp2["Genres"].unique()
```

Out[85]:
```
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)
```

=> Since, There are too many categories under Genres. Hence, we will try to reduce some categories which have very few samples under them and put them under one new common category i.e. "Other"

In [ ]:
```

```

In [86]:
```
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 [87]:
```
inp2["Genres"].unique()
```

Out[87]:
```
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 [88]:
```
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 [89]:
```
inp2.head()
```

Out[89]:

	Rating	Reviews	Size	Installs	Price	Content Rating	Category_ART_AND_DESIGN	Categor
0	4.1	5.075174	19000.0	10000	0.0	Everyone	1	
1	3.9	6.875232	14000.0	500000	0.0	Everyone	1	
2	4.7	11.379520	8700.0	5000000	0.0	Everyone	1	
4	4.3	6.875232	2800.0	100000	0.0	Everyone	1	
5	4.4	5.123964	5600.0	50000	0.0	Everyone	1	

5 rows × 91 columns

In [90]:
```
# Dummy encoding "Content Rating"
```

In [91]:
```
#get unique values in Column "Content Rating"
inp2["Content Rating"].unique()
```

Out[91]:
```
array(['Everyone', 'Teen', 'Everyone 10+', 'Mature 17+',
       'Adults only 18+', 'Unrated'], dtype=object)
```

In [92]:
```
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[92]:

	Rating	Reviews	Size	Installs	Price	Category_ART_AND_DESIGN	Category_AUTO_A
0	4.1	5.075174	19000.0	10000	0.0	1	
1	3.9	6.875232	14000.0	500000	0.0	1	
2	4.7	11.379520	8700.0	5000000	0.0	1	
4	4.3	6.875232	2800.0	100000	0.0	1	
5	4.4	5.123964	5600.0	50000	0.0	1	

5 rows × 96 columns

In [93]:
```
inp2.shape
```

Out[93]:
```
(8496, 96)
```

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

In [94]:
```
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 [ ]:
```

```

In [95]:
```
inp2.isnull().sum()
```

Out[95]:
```
Rating                              0
Reviews                             0
Size                             1189
Installs                            0
Price                               0
                                  ...
Content Rating_Everyone             0
Content Rating_Everyone 10+         0
Content Rating_Mature 17+           0
Content Rating_Teen                 0
Content Rating_Unrated              0
Length: 96, dtype: int64
```

In [96]:
```
inp2.dropna(inplace=True)
```

In [97]:
```
inp2.isnull().sum()
```

Out[97]:
```
Rating                            0
Reviews                           0
Size                              0
Installs                          0
Price                             0
                                 ..
Content Rating_Everyone           0
Content Rating_Everyone 10+       0
Content Rating_Mature 17+         0
Content Rating_Teen               0
Content Rating_Unrated            0
Length: 96, dtype: int64
```

In [99]:
```
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)
```

Model building

In [100]:
```
reg_all = LR()
reg_all.fit(Xtrain,ytrain)
```

Out[100]:
```
LinearRegression()
```

In [101]:
```
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 : 0.068
```

In [102]:
```
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 : 0.058
```

In [ ]:
```

```