Google Play Store App Rating Prediction - *Lwandile Ngesi*

Objective: Make a model to predict the app rating, with other information about the app provided.

Fields in the data –

- App: Application name
- Category: Category to which the app belongs
- Rating: Overall user rating of the app
- Reviews: Number of user reviews for the app
- Size: Size of the app
- Installs: Number of user downloads/installs for the app
- Type: Paid or Free
- Price: Price of the app
- Content Rating: Age group the app is targeted at - Children / Mature 21+ / Adult
- Genres: An app can belong to multiple genres (apart from its main category). For example, a musical family game will belong to Music, Game, Family genres.
- Last Updated: Date when the app was last updated on Play Store
- Current Ver: Current version of the app available on Play Store
- Android Ver: Minimum required Android version

1. Load the data file using pandas.

In [93]:

```python
#Required Imports
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt, seaborn as sns
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')
```

In [2]:

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

In [3]:

```
data.head()
```

Out[3]:

	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 â€“ FREE Live Cool Themes, Hid...	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 [4]:

```
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 [5]:

```
data.shape
```

Out[5]:

```
(10841, 13)
```

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

In [6]:

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

Out[6]:

```
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 [7]:

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

Out[7]:

```
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 [8]:

```
data = data.dropna()
```

In [9]:

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

Out[9]:

```
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 [10]:

```
data.shape
```

Out[10]:

```
(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.

Extract the numeric value from the column

Multiply the value by 1,000, if size is mentioned in Mb

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

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 [11]:

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

In [12]:

```
data.head()
```

Out[12]:

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

In [13]:

```
data["Size"] = 1000 * data["Size"]
```

In [14]:

```
data
```

Out[14]:

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

9360 rows × 13 columns

In [15]:

```
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 [16]:

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

In [17]:

```
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
```

In [18]:

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

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	159.0	19000.0	10000.0	Free	0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000.0	Free	0	Everyone
2	U Launcher Lite â€“ FREE Live Cool Themes, Hid...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000.0	Free	0	Everyone
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25000.0	50000000.0	Free	0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000.0	Free	0	Everyone

In [20]:

```
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 [21]:

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

In [22]:

```
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
```

In [23]:

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

In [24]:

```
data.head()
```

Out[24]:

	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
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000	Free	0.0	Everyone
2	U Launcher Lite â€“ FREE Live Cool Themes, Hid...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000	Free	0.0	Everyone
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25000.0	50000000	Free	0.0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000	Free	0.0	Everyone

In [25]:

```
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 [26]:

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

In [27]:

```
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
```

5.1. Sanity Checks:

- Average rating should be between 1 and 5.

In [28]:

```
data.shape
```

Out[28]:

```
(9360, 13)
```

In [29]:

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

```

In [30]:

```
data.shape
```

Out[30]:

```
(9360, 13)
```

- Reviews should not be more than installs

In [31]:

```
data.shape
```

Out[31]:

```
(9360, 13)
```

In [32]:

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

In [33]:

```
data.shape
```

Out[33]:

```
(9353, 13)
```

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

In [34]:

```
data.shape
```

Out[34]:

```
(9353, 13)
```

In [35]:

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

In [36]:

```
data.shape
```

Out[36]:

```
(9353, 13)
```

5.1. Boxplot for Price

In [37]:

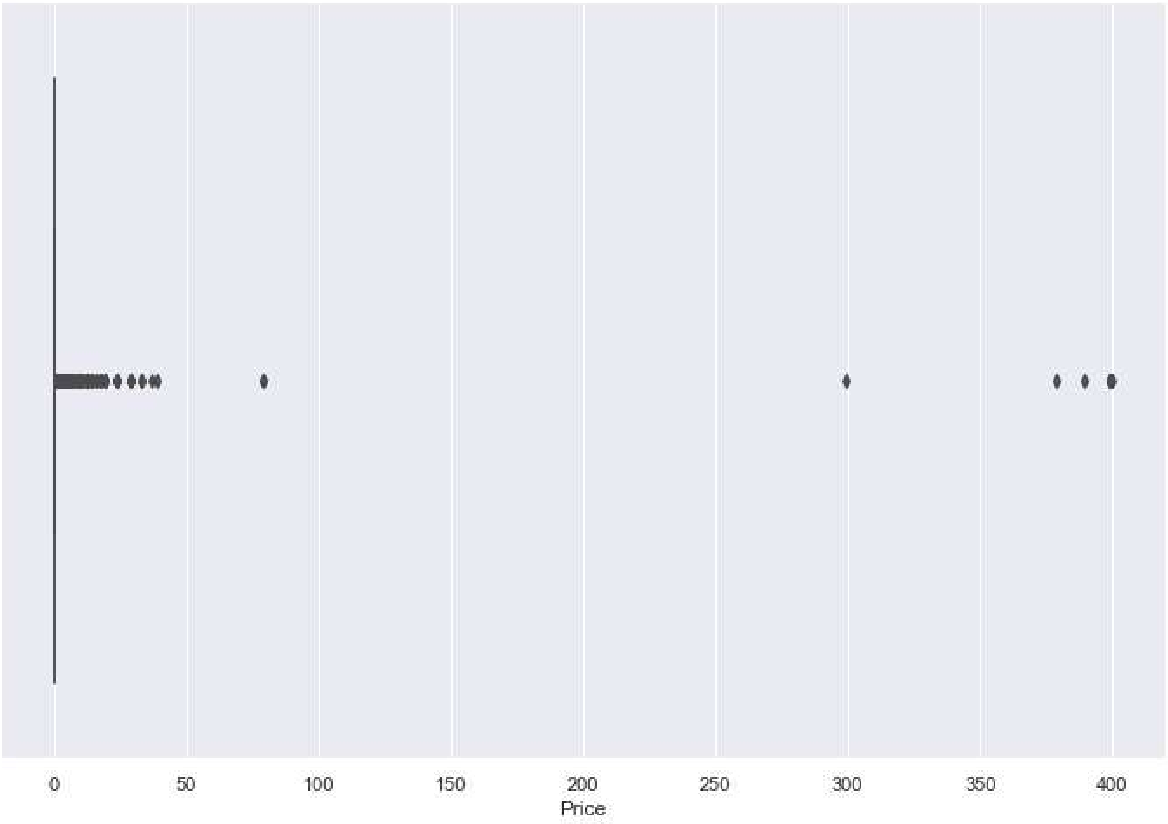
```
sns.set(rc={'figure.figsize':(12,8)})
```

In [38]:

```
sns.boxplot(data['Price'])
```

Out[38]:

```
<AxesSubplot:xlabel='Price'>
```



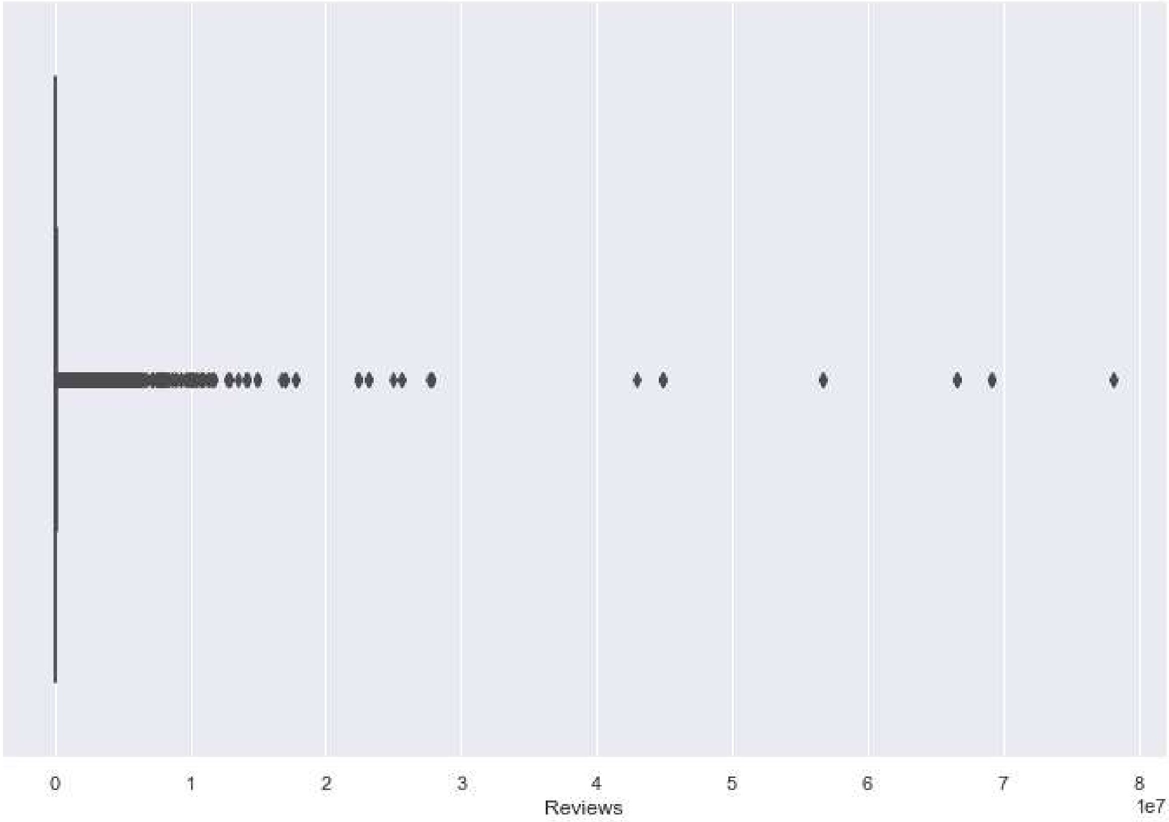
5.2. Boxplot for Reviews

In [39]:

```
sns.boxplot(data['Reviews'])
```

Out[39]:

```
<AxesSubplot:xlabel='Reviews'>
```



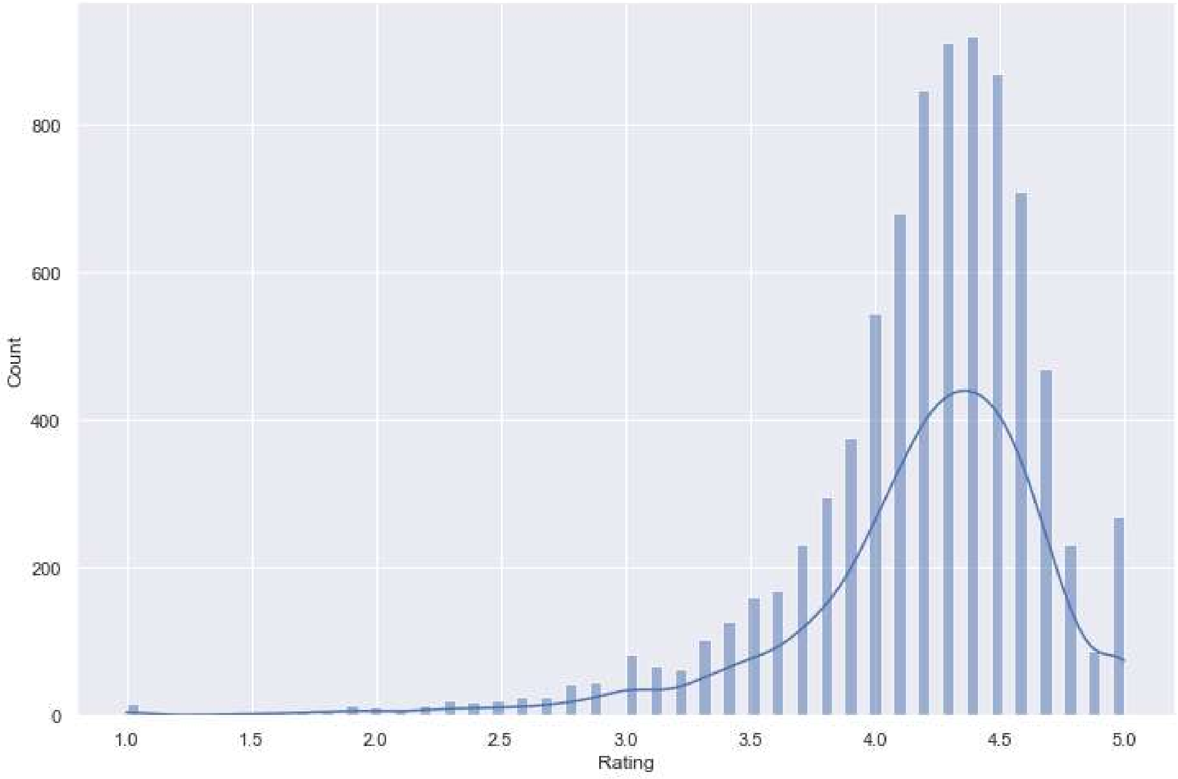
5.3. Histogram for Rating

In [91]:

```
sns.histplot(data['Rating'], kde=True)
```

Out[91]:

```
<AxesSubplot:xlabel='Rating', ylabel='Count'>
```



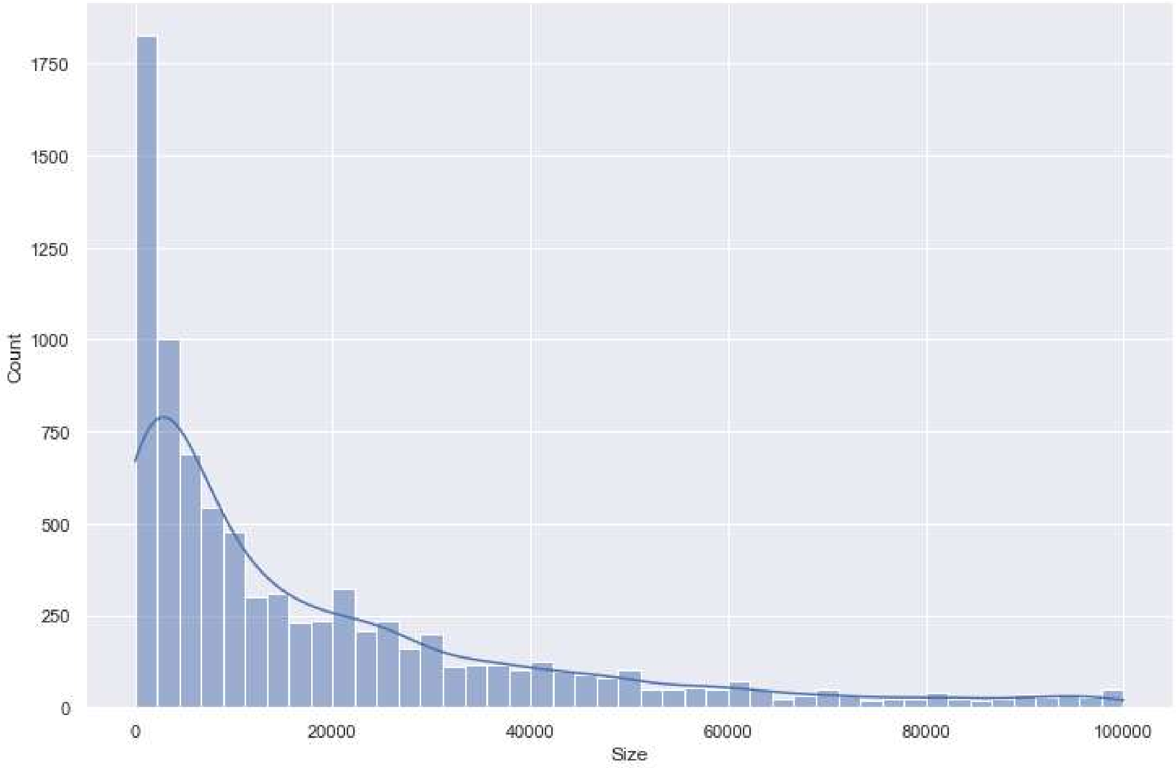
5.4. Histogram for Size

In [92]:

```
sns.histplot(data['Size'], kde=True)
```

Out[92]:

```
<AxesSubplot:xlabel='Size', ylabel='Count'>
```



6. Drop all prices greater than $200

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)
```

6.2. Drop records having more than 2 million reviews.

In [47]:

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

In [48]:

```
data.shape
```

Out[48]:

```
(8885, 13)
```

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

In [88]:

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

Out[88]:

	Rating	Reviews	Size	Installs	Price
0.10	3.5	2.833213	0.0	1000.0	0.0
0.25	4.0	4.905275	2900.0	10000.0	0.0
0.50	4.3	8.109676	9800.0	100000.0	0.0
0.70	4.5	10.224157	23000.0	1000000.0	0.0
0.90	4.7	12.168641	50000.0	10000000.0	0.0
0.95	4.8	12.782290	68250.0	10000000.0	1.0
0.99	5.0	13.709092	95000.0	10000000.0	7.0

In [50]:

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

In [51]:

```python
data.shape
```

Out[51]:

```
(8496, 13)
```

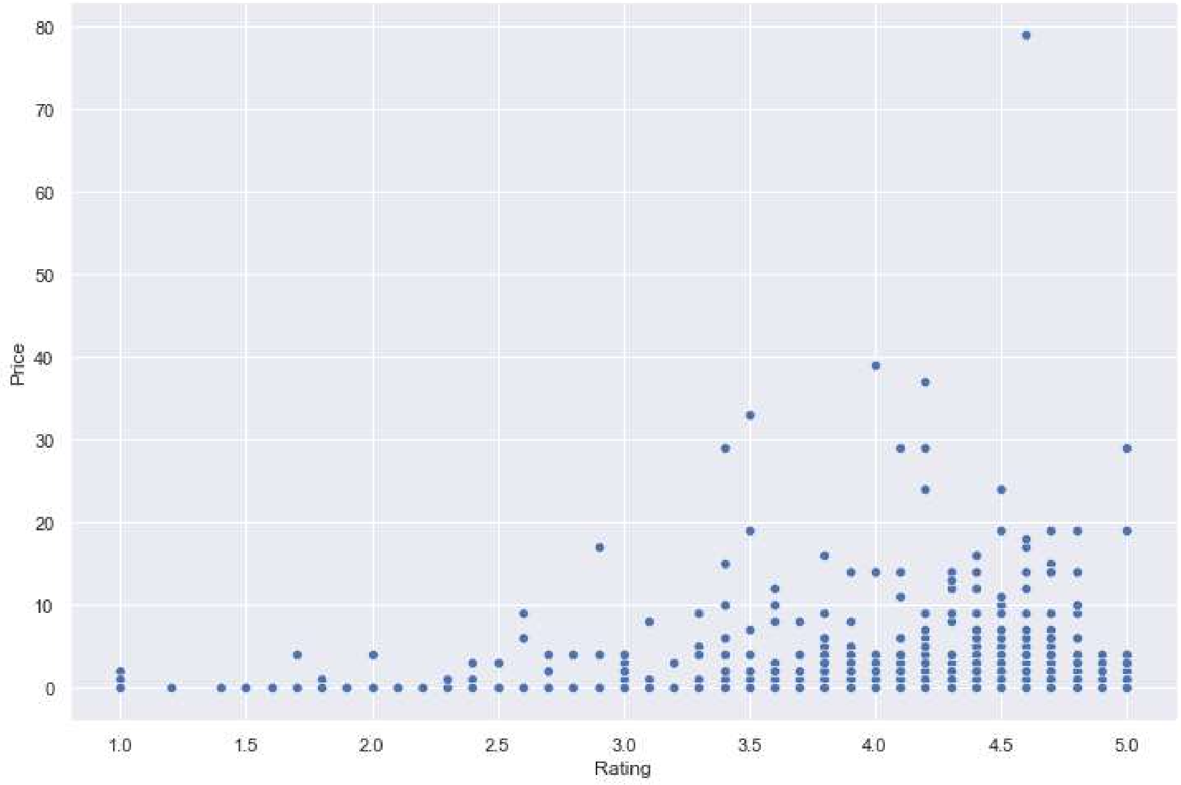
7.1. scatter plot/joinplot for Rating vs. Price

In [52]:

```
sns.scatterplot(x='Rating',y='Price',data=data)
```

Out[52]:

```
<AxesSubplot:xlabel='Rating', ylabel='Price'>
```



Yes, Paid apps have higher ratings as compared to Free apps.

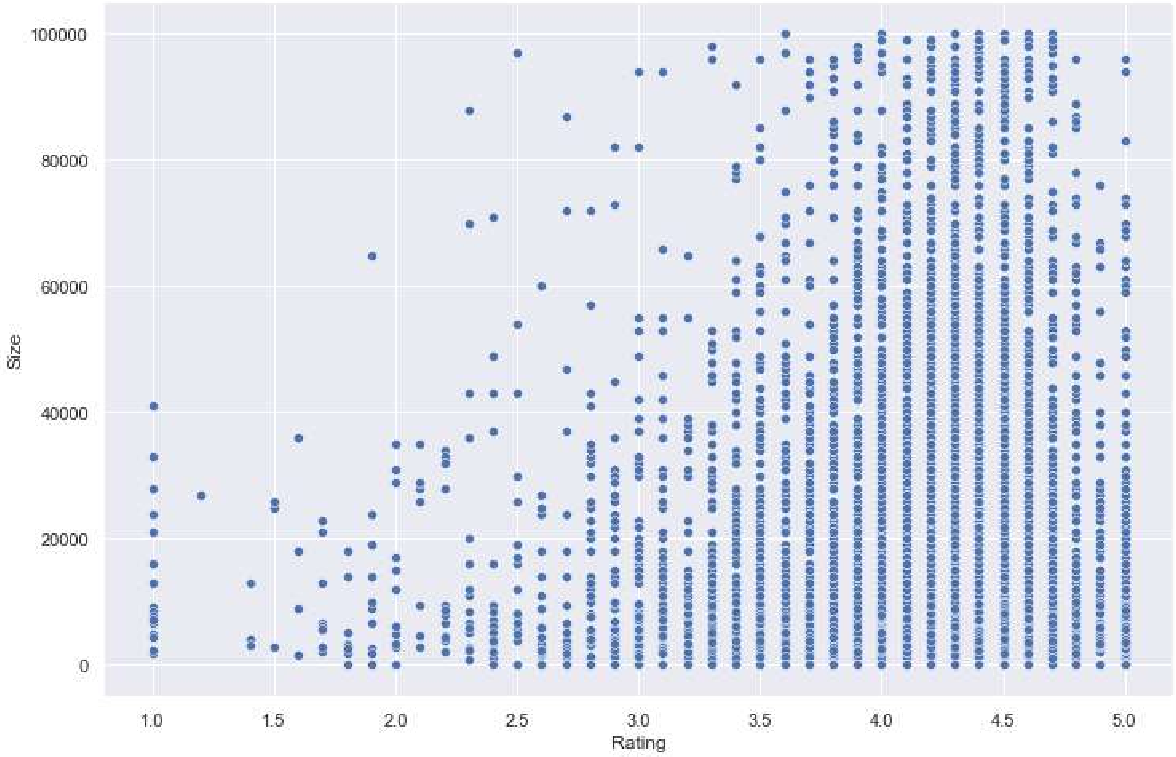
7.2. Scatter plot/joinplot for Rating vs. Size

In [53]:

```
sns.scatterplot(x='Rating',y='Size',data=data)
```

Out[53]:

```
<AxesSubplot:xlabel='Rating', ylabel='Size'>
```



- Yes it is clear that heavier apps are rated better on play store

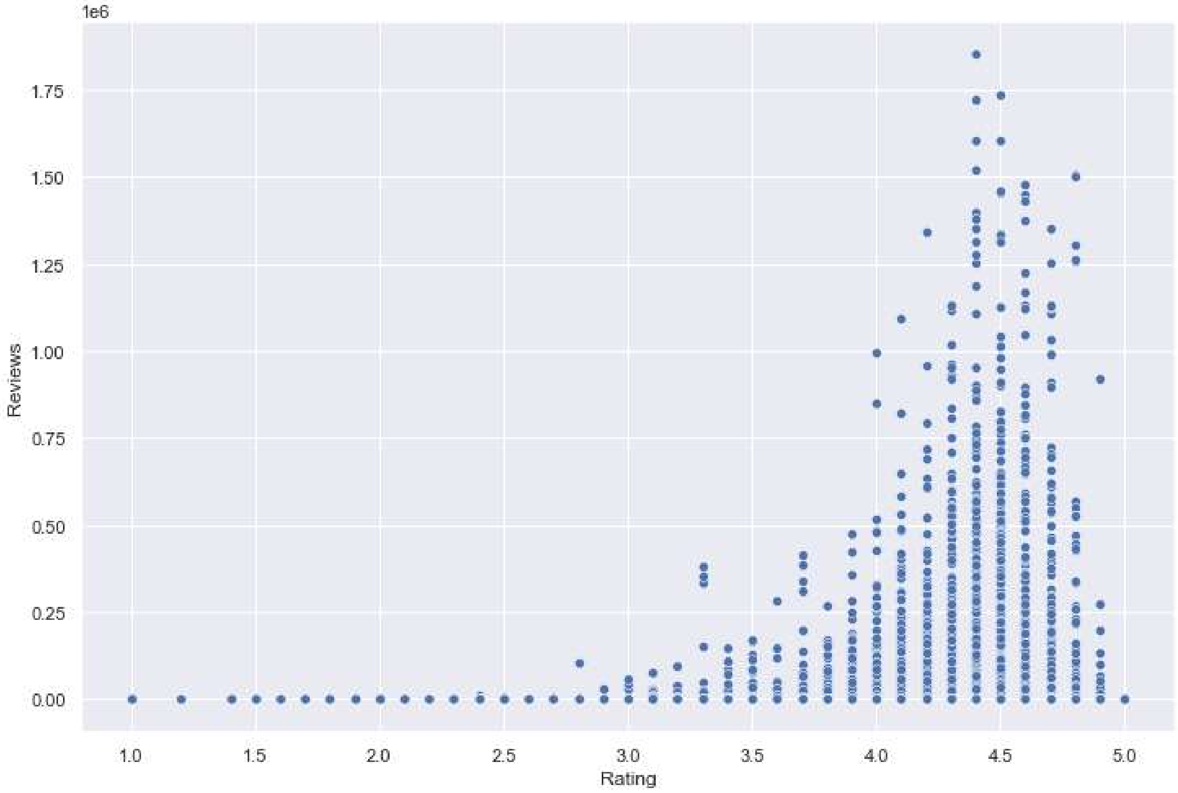
7.3. Scatter plot/joinplot for Rating vs. Reviews

In [54]:

```
sns.scatterplot(x='Rating',y='Reviews',data=data)
```

Out[54]:

```
<AxesSubplot:xlabel='Rating', ylabel='Reviews'>
```



- The more the reviews, the better the app ratting.

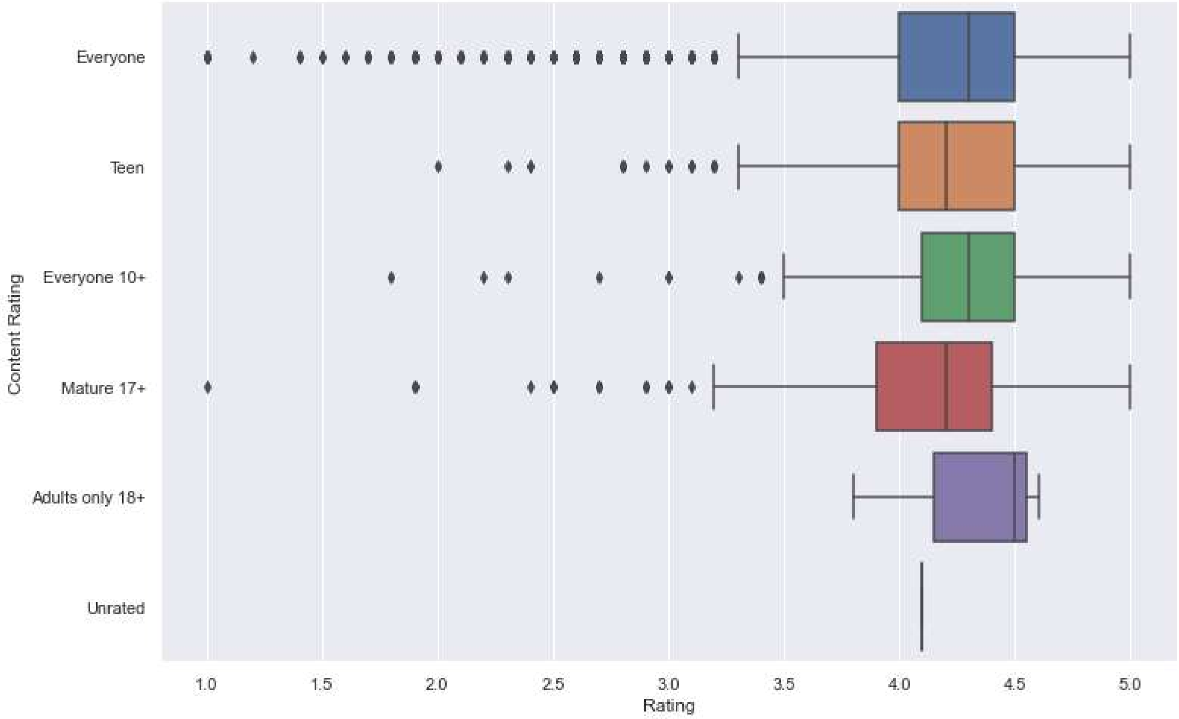
7.4. Boxplot for Rating vs. Content Rating

In [55]:

```
sns.boxplot(x="Rating", y="Content Rating", data=data)
```

Out[55]:

```
<AxesSubplot:xlabel='Rating', ylabel='Content Rating'>
```



- Apps which are for everyone have more negetive ratings as compared to other sections as it has so much outlier value, while 18+ apps have better ratings.

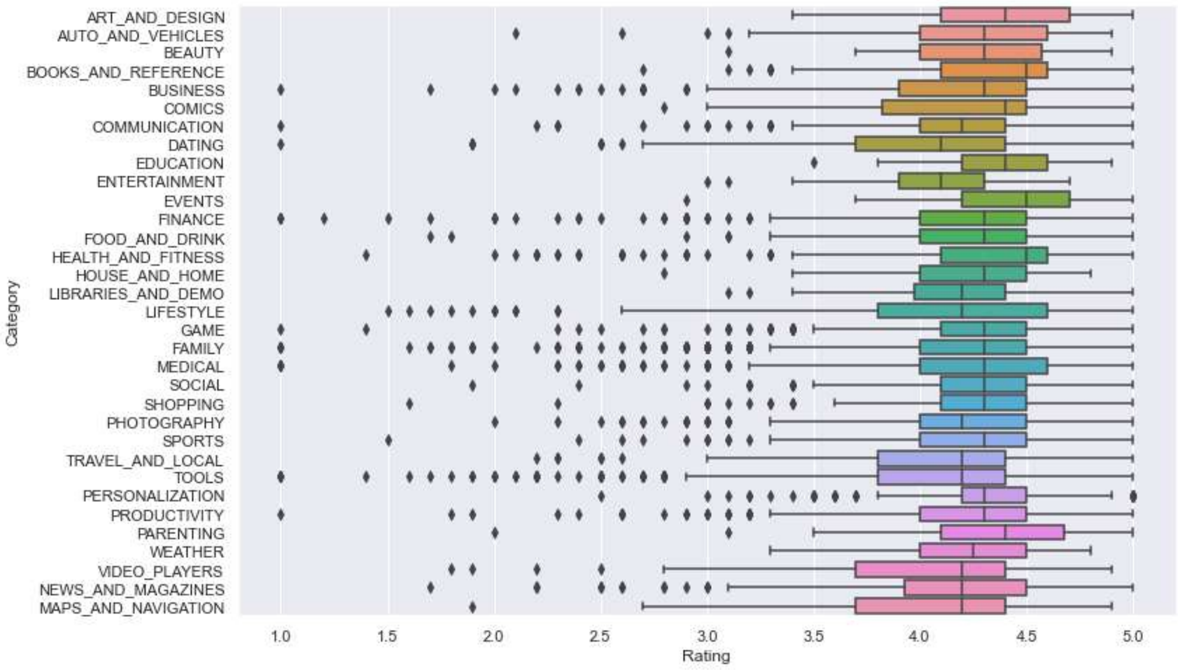
7.5. Boxplot for Ratings vs. Category

In [56]:

```
sns.boxplot(x="Rating", y="Category", data=data)
```

Out[56]:

```
<AxesSubplot:xlabel='Rating', ylabel='Category'>
```



- The Events category has better ratings compared to others.

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

In [57]:

```
inp1 = data
```

In [58]:

```
inp1.head()
```

Out[58]:

	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	
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14000.0	500000	Free	0	Everyone	
2	U Launcher Lite â€“ FREE Live Cool Themes, Hid...	ART_AND_DESIGN	4.7	87510.0	8700.0	5000000	Free	0	Everyone	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2800.0	100000	Free	0	Everyone	D
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167.0	5600.0	50000	Free	0	Everyone	

In [59]:

```
inp1.skew()
```

Out[59]:

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

In [60]:

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

In [61]:

```
reviewskew.skew()
```

Out[61]:

```
-0.20039949659264134
```

In [62]:

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

Out[62]:

```
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 [63]:

```
installsskew.skew()
```

Out[63]:

```
-0.5097286542754812
```

In [64]:

```
inp1.head()
```

Out[64]:

	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
1	Coloring book moana	ART_AND_DESIGN	3.9	6.875232	14000.0	500000	Free	0	Everyone
2	U Launcher Lite â€“ FREE Live Cool Themes, Hid...	ART_AND_DESIGN	4.7	11.379520	8700.0	5000000	Free	0	Everyone
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	6.875232	2800.0	100000	Free	0	Everyone
5	Paper flowers instructions	ART_AND_DESIGN	4.4	5.123964	5600.0	50000	Free	0	Everyone

8.2. Drop columns App, Last Updated, Current Ver, and Android Ver.

In [65]:

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

In [66]:

```
inp1.head()
```

Out[66]:

	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 [67]:

```
inp1.shape
```

Out[67]:

```
(8496, 8)
```

8.3. Get dummy columns for Category, Genres, and Content Rating.

In [68]:

```
inp2 = inp1
```

In [69]:

```
inp2.head()
```

Out[69]:

	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

- Apply Dummy Encoding on Column "Category"

In [70]:

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

Out[70]:

```
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 [71]:

```
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[71]:

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

5 rows × 40 columns

In [72]:

```
inp2.shape
```

Out[72]:

```
(8496, 40)
```

- Apply Dummy Encoding on Column "Genres"

In [73]:

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

Out[73]:

```
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 [74]:

```
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 [75]:

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

Out[75]:

```
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 [76]:

```
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 [77]:

```
inp2.head()
```

Out[77]:

	Rating	Reviews	Size	Installs	Price	Content Rating	Category_ART_AND_DESIGN	Category_
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 [78]:

```
inp2.shape
```

Out[78]:

```
(8496, 91)
```

- Apply Dummy Encoding on Column "Content Rating"

In [79]:

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

Out[79]:

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

In [80]:

```
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[80]:

	Rating	Reviews	Size	Installs	Price	Category_ART_AND_DESIGN	Category_AUTO_AN
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

In [81]:

```
inp2.shape
```

Out[81]:

```
(8496, 96)
```

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 [82]:

```
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 [83]:

```
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 [84]:

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

Out[84]:

```
LinearRegression()
```

In [85]:

```
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 [86]:

```
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 [ ]:

```

```

..........END OF FILE..........