GPS_Project

January 22, 2023

[313]:
```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[314]:
```
# Load the data file using pandas.
df = pd.read_csv('googleplaystore.csv')
```

1 ******** 1 Know your data ********

[315]:
```
df.head()
```

[315]:
```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite – FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7
3                              Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

  Reviews  Size     Installs  Type Price Content Rating  \
0     159   19M      10,000+  Free     0       Everyone
1     967   14M     500,000+  Free     0       Everyone
2   87510  8.7M   5,000,000+  Free     0       Everyone
3  215644   25M  50,000,000+  Free     0           Teen
4     967  2.8M     100,000+  Free     0       Everyone

                     Genres      Last Updated         Current Ver  \
0              Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
2              Art & Design    August 1, 2018               1.2.4
3              Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
```

```
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up
```

[316]:

```
df.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
```

[317]:

```
df.describe(include='all')
```

[317]:

```
           App Category       Rating Reviews                Size    Installs  \
count    10841    10841  9367.000000   10841               10841       10841
unique    9660       34          NaN    6002                 462          22
top     ROBLOX   FAMILY          NaN       0  Varies with device  1,000,000+
freq         9     1972          NaN     596                1695        1579
mean       NaN      NaN     4.193338     NaN                 NaN         NaN
std        NaN      NaN     0.537431     NaN                 NaN         NaN
min        NaN      NaN     1.000000     NaN                 NaN         NaN
25%        NaN      NaN     4.000000     NaN                 NaN         NaN
50%        NaN      NaN     4.300000     NaN                 NaN         NaN
75%        NaN      NaN     4.500000     NaN                 NaN         NaN
max        NaN      NaN    19.000000     NaN                 NaN         NaN

         Type  Price Content Rating Genres    Last Updated  \
count   10840  10841          10840  10841           10841
unique      3     93              6    120            1378
```

```
top      Free      0       Everyone  Tools  August 3, 2018
freq    10039  10040           8714    842             326
mean      NaN    NaN            NaN    NaN             NaN
std       NaN    NaN            NaN    NaN             NaN
min       NaN    NaN            NaN    NaN             NaN
25%       NaN    NaN            NaN    NaN             NaN
50%       NaN    NaN            NaN    NaN             NaN
75%       NaN    NaN            NaN    NaN             NaN
max       NaN    NaN            NaN    NaN             NaN

               Current Ver Android Ver
count                10833       10838
unique                2832          33
top     Varies with device  4.1 and up
freq                  1459        2451
mean                   NaN         NaN
std                    NaN         NaN
min                    NaN         NaN
25%                    NaN         NaN
50%                    NaN         NaN
75%                    NaN         NaN
max                    NaN         NaN
```

[318]:
```
# Check for null values in the data. Get the number of null values for each␣
 ↪column
```

[319]:
```
df.isnull().sum()
```

[319]:
```
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
```

[320]:
```
round((df.isnull().sum()/ df.shape[0]) * 100,2)
```

[320]:
```
App                0.00
Category           0.00
```

```
       Rating             13.60
       Reviews             0.00
       Size                0.00
       Installs            0.00
       Type                0.01
       Price               0.00
       Content Rating      0.01
       Genres              0.00
       Last Updated        0.00
       Current Ver         0.07
       Android Ver         0.03
       dtype: float64
```

```
[321]: df.shape
```

```
[321]: (10841, 13)
```

```
[322]: df.dropna(inplace=True)
       df.isnull().sum()
```

```
[322]: 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
```

```
[323]: df.shape
```

```
[323]: (9360, 13)
```

```
[324]: # 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
```

[325]:

```
df.head(2)
```

[325]:

```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                Coloring book moana  ART_AND_DESIGN     3.9

  Reviews Size  Installs  Type Price Content Rating  \
0     159  19M   10,000+  Free     0       Everyone
1     967  14M  500,000+  Free     0       Everyone

                     Genres      Last Updated Current Ver   Android Ver
0              Art & Design   January 7, 2018       1.0.0  4.0.3 and up
1 Art & Design;Pretend Play  January 15, 2018       2.0.0  4.0.3 and up
```

[326]:

```
 df['Size']
```

[326]:

```
0                       19M
1                       14M
2                      8.7M
3                       25M
4                      2.8M
                ...
10834                  2.6M
10836                   53M
10837                  3.6M
10839    Varies with device
10840                   19M
Name: Size, Length: 9360, dtype: object
```

[327]:

```
def size_convert(y):
    if 'M' in y:
        x = y[:-1] #19M
        x = float(x)*1000
        return x
    elif 'K' in y:
        x = y[:-1]
        return x
    else:
        return None
```

[328]:

```
df['Size'] = df['Size'].apply(size_convert)
```

[329]:

```
df.head()
```

[329]:
```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite – FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7
3                              Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

  Reviews     Size     Installs  Type Price Content Rating  \
0     159  19000.0      10,000+  Free     0       Everyone
1     967  14000.0     500,000+  Free     0       Everyone
2   87510   8700.0   5,000,000+  Free     0       Everyone
3  215644  25000.0  50,000,000+  Free     0           Teen
4     967   2800.0     100,000+  Free     0       Everyone

                       Genres      Last Updated         Current Ver  \
0                Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
2                Art & Design    August 1, 2018               1.2.4
3                Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up
```

[330]:
```python
# Reviews is a numeric field that is loaded as a string field. Convert it to␣
 ↪numeric (int/float).
df['Reviews'] = df['Reviews'].astype(float)
```

[331]:
```python
df['Size'].value_counts()
```

[331]:
```
14000.0    165
12000.0    161
11000.0    159
15000.0    159
13000.0    157
          …
89000.0      9
84000.0      9
86000.0      8
90000.0      5
1000.0       4
Name: Size, Length: 181, dtype: int64
```

[332]:
```
df['Size'].isnull().sum()
```

[332]:
```
1894
```

[333]:
```
df['Size']
```

[333]:
```
0        19000.0
1        14000.0
2         8700.0
3        25000.0
4         2800.0
          …
10834     2600.0
10836    53000.0
10837     3600.0
10839        NaN
10840    19000.0
Name: Size, Length: 9360, dtype: float64
```

[334]:
```
# df['Size'].fillna(method='ffill',inplace=True)
df['Size'] = df['Size'].fillna(0.0)
```

[335]:
```
df['Size']
```

[335]:
```
0        19000.0
1        14000.0
2         8700.0
3        25000.0
4         2800.0
          …
10834     2600.0
10836    53000.0
10837     3600.0
10839        0.0
10840    19000.0
Name: Size, Length: 9360, dtype: float64
```

[336]:
```
# 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
```

[337]:
```
df.head(2)
```

[337]:
```
                                              App        Category  Rating  \
0  Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
```

```
1                            Coloring book moana  ART_AND_DESIGN     3.9

   Reviews     Size  Installs  Type Price Content Rating  \
0    159.0  19000.0   10,000+  Free     0       Everyone
1    967.0  14000.0  500,000+  Free     0       Everyone

                     Genres      Last Updated Current Ver   Android Ver
0              Art & Design   January 7, 2018       1.0.0  4.0.3 and up
1  Art & Design;Pretend Play  January 15, 2018       2.0.0  4.0.3 and up
```

[338]:
```
df['Installs'] = df['Installs'].str.replace('+','')
df['Installs'] = df['Installs'].str.replace(',','')
```

[339]:
```
df.head(2)
```

[339]:
```
                                                App        Category  Rating  \
0  Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                            Coloring book moana  ART_AND_DESIGN     3.9

   Reviews     Size Installs  Type Price Content Rating  \
0    159.0  19000.0    10000  Free     0       Everyone
1    967.0  14000.0   500000  Free     0       Everyone

                     Genres      Last Updated Current Ver   Android Ver
0              Art & Design   January 7, 2018       1.0.0  4.0.3 and up
1  Art & Design;Pretend Play  January 15, 2018       2.0.0  4.0.3 and up
```

[340]:
```
df['Installs'] = df['Installs'].astype(int)
```

[341]:
```
# Reviews should not be more than installs as only those who installed can␣
 ↪review the app. If there are any such records, drop them.
```

[342]:
```
df.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   int64
 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), int64(1), object(9)
memory usage: 1023.8+ KB
```

2 Sanity checks:

Average rating should be between 1 and 5 as only these values are allowed on the play store. Di
range.
Reviews should not be more than installs as only those who installed can review the app. If the
For free apps (type = "Free"), the price should not be >0. Drop any such rows.

```
[343]: df['Reviews']= df['Reviews'].astype(float)
```

```
[344]: df[df['Reviews']>df['Installs']].index
```

```
[344]: Int64Index([2454, 4663, 5917, 6700, 7402, 8591, 10697], dtype='int64')
```

```
[345]: df.drop(df[df['Reviews']>df['Installs']].index,inplace=True)
```

```
[346]: df[df['Reviews']>df['Installs']].index
```

```
[346]: Int64Index([], dtype='int64')
```

```
[347]: df['Rating'].min(),df['Rating'].max() # No overvation found for more 5 and less␣
         ↪than 1 Ratings
```

```
[347]: (1.0, 5.0)
```

```
[348]: df['Rating'].mean()
```

```
[348]: 4.191254143055709
```

```
[349]: df[df['Reviews']>df['Installs']].index
```

```
[349]: Int64Index([], dtype='int64')
```

```
[350]: df[df['Reviews']>df['Installs']]
```

```
[350]: Empty DataFrame
        Columns: [App, Category, Rating, Reviews, Size, Installs, Type, Price, Content
        Rating, Genres, Last Updated, Current Ver, Android Ver]
        Index: []
```

[351]:
```
# Price field is a string and has $ symbol. Remove '$' sign, and convert it to␣
 ↪numeric.
df['Price'] = df['Price'].str.replace('$','')
df['Price'] = df['Price'].astype(float)
```

[352]:
```
df['Price'][df['Type']=='Free'].min()
```

[352]:
```
0.0
```

[353]:
```
df['Price'][df['Type']=='Free'].max()
```

[353]:
```
0.0
```

[354]:
```
df.dtypes
```

[354]:
```
App                  object
Category             object
Rating              float64
Reviews             float64
Size                float64
Installs              int64
Type                 object
Price               float64
Content Rating       object
Genres               object
Last Updated         object
Current Ver          object
Android Ver          object
dtype: object
```

[355]:
```
df[(df['Price']>0) & (df['Type']=='Free')]
```

[355]:
```
Empty DataFrame
Columns: [App, Category, Rating, Reviews, Size, Installs, Type, Price, Content
Rating, Genres, Last Updated, Current Ver, Android Ver]
Index: []
```

3 2 Univariate Analysis

```
5. Performing univariate analysis:
Boxplot for Price
Are there any outliers? Think about the price of usual apps on Play Store.
Boxplot for Reviews
Are there any apps with very high number of reviews? Do the values seem right?
Histogram for Rating
How are the ratings distributed? Is it more toward higher ratings?
```

Histogram for Size

Note down your observations for the plots made above. Which of these seem to have outliers?Are

[356]:

```
df['Price'].drop_duplicates()
```

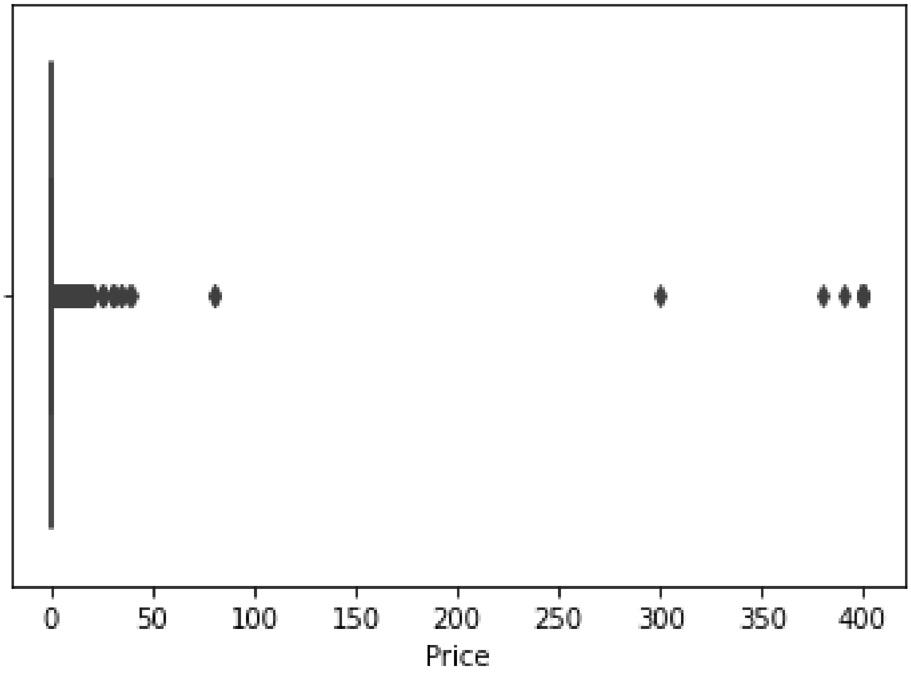
[356]:

```
0        0.00
234      4.99
427      3.99
477      6.99
481      7.99
         …
9465     2.95
9490     2.90
9566     1.97
9869     2.56
10785    1.20
Name: Price, Length: 73, dtype: float64
```

[357]:

```
#Boxplot for Price
sns.boxplot('Price',data=df)
plt.show()
```

```
/usr/local/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

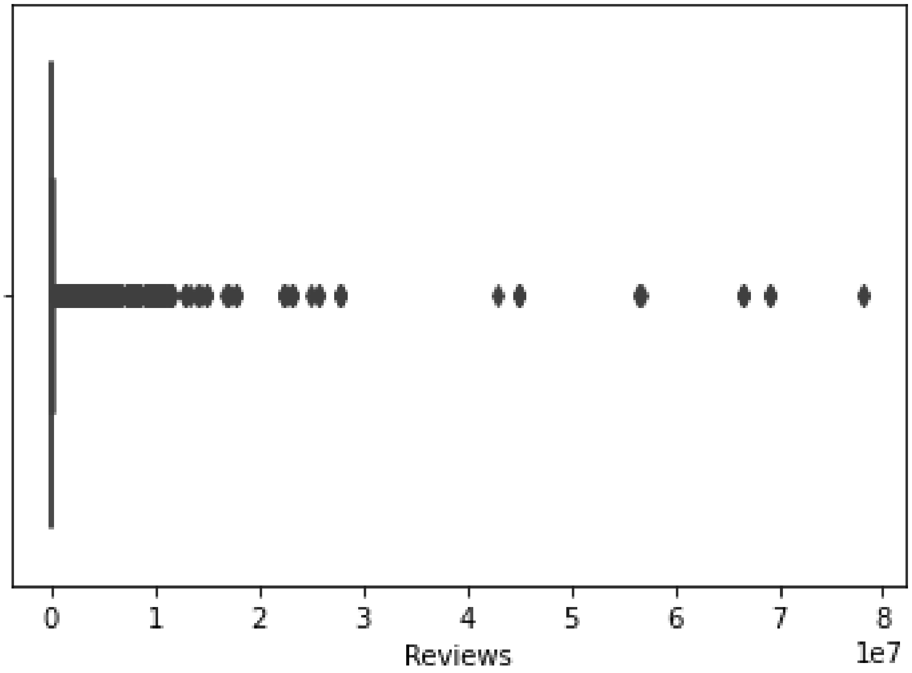


[358]:
```
# Are there any outliers? Think about the price of usual apps on Play Store.

#Ans : Yes, most of the apps are lies between 0-25-50$ price range and few apps␣
 ↪lies between 300-400$ which is more than usual apps on play store
```

[359]:
```
# Boxplot for Reviews
sns.boxplot('Reviews',data=df)
plt.show()
```

```
/usr/local/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



[360]:
```
# Are there any apps with very high number of reviews? Do the values seem right?
# Ans -  Certaintly there are some app which has high number of reviews
```

[361]:
```
df['Reviews'].max()
```

[361]:
```
78158306.0
```

[362]:

```
df['Reviews'].mean()
```

[362]:

```
514760.5758580135
```

[363]:

```
df['Reviews'].describe()
```

[363]:

```
count    9.353000e+03
mean     5.147606e+05
std      3.146169e+06
min      1.000000e+00
25%      1.870000e+02
50%      5.967000e+03
75%      8.174700e+04
max      7.815831e+07
Name: Reviews, dtype: float64
```

[364]:

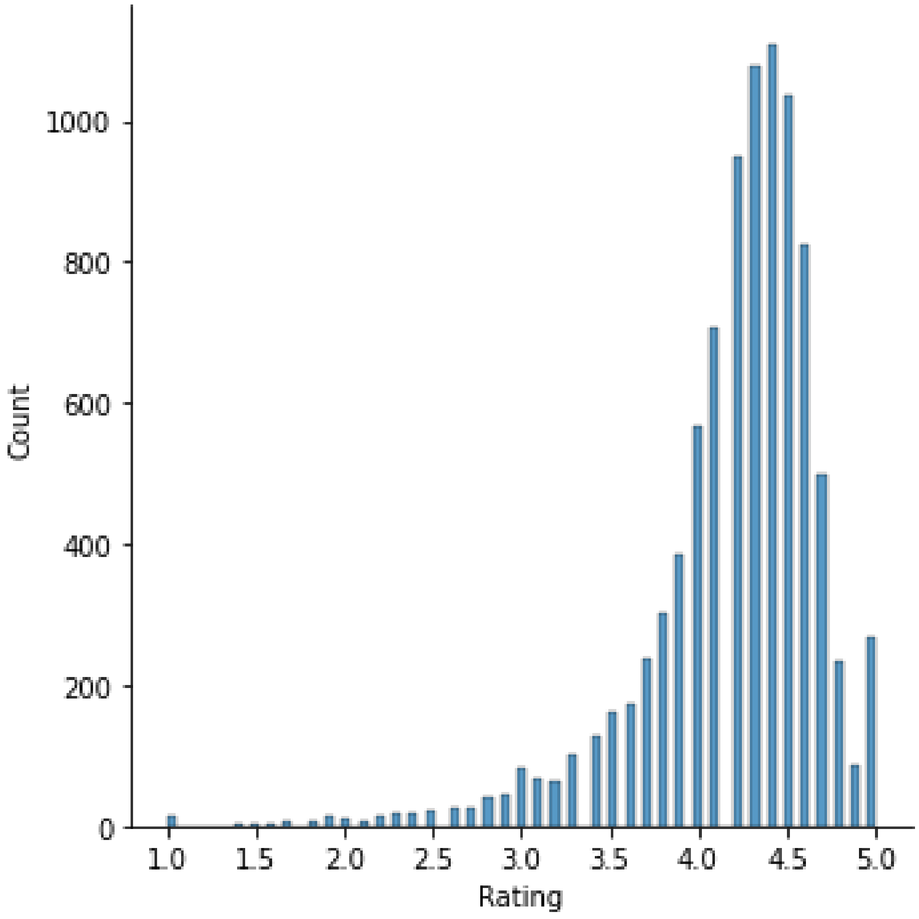
```
# Histogram for Ratingunique
```

[365]:

```
sns.displot(df['Rating'])
```

[365]:

```
<seaborn.axisgrid.FacetGrid at 0x7f309bfa4610>
```

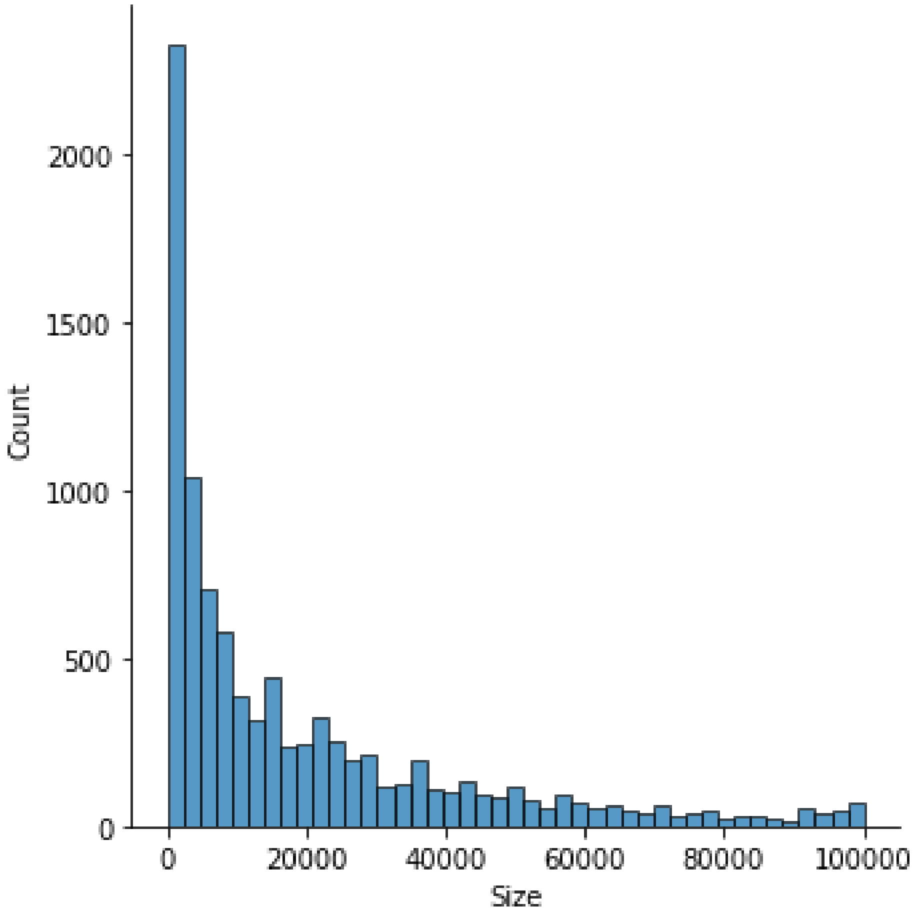


[366]:
```
# How are the ratings distributed? Is it more toward higher ratings?
# if majority of the data is towards left then it is right skewed distribution
# if the majority of the data is towards right then it is left skewed␣
 ↪distribution

# Ans -There is negative skewed data (left side) , some apps having higher␣
 ↪rating that usual
```

[367]:
```
# Histogram for Size
sns.displot(df['Size'])
```

[367]:
```
<seaborn.axisgrid.FacetGrid at 0x7f309bdcded0>
```



[368]:
```
# Graph indicates positovly skewed(Right side) data
```

4 3 Outliers

```
6. Outlier treatment:
Price: From the box plot, it seems like there are some apps with very high price. A price of $2
for an application on the Play Store is very high and suspicious!
Check out the records with very high price
Is 200 indeed a high price?
Drop these as most seem to be junk apps
Reviews: Very few apps have very high number of reviews. These are all star apps that don’t hel
with the analysis and, in fact, will skew it. Drop records having more than 2 million reviews.
Installs: There seems to be some outliers in this field too. Apps having very high number of in
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 than that
```

[369]:
```
df[df['Price']>200].size
```

[369]:
```
195
```

[370]:
```
# There are 195 obervation where apps has price greater that 200
```

[371]:
```
# Size of data before dropping
df.shape
```

[371]:
```
(9353, 13)
```

[372]:
```
# df.drop(df['Price']>200
df.drop(df[df['Price']>200].index,inplace=True)
```

[373]:
```
df[df['Price']>200].size
```

[373]:
```
0
```

[374]:
```
df.shape
```

[374]:
```
(9338, 13)
```

5 4 Review Drops

Reviews: Very few apps have very high number of reviews. These are all star apps that don't hel
with the analysis and, in fact, will skew it. Drop records having more than 2 million reviews.

[375]:
```
df['Reviews'].min()
```

[375]:
```
1.0
```

[376]:
```
df['Reviews'].max()
```

[376]:
```
78158306.0
```

[377]:
```
# drop records check shape before and after
```

[378]:
```
df.shape
```

[378]:
```
(9338, 13)
```

[379]:
```
df[df['Reviews']>20e5].count()
```

[379]:
```
App                 453
Category            453
Rating              453
```

```
Reviews           453
Size              453
Installs          453
Type              453
Price             453
Content Rating    453
Genres            453
Last Updated      453
Current Ver       453
Android Ver       453
dtype: int64
```

[380]:
```
#Drop rows from DB where reviews >2mn
df.drop(df[df['Reviews']>20e5].index,inplace=True)
```

[381]:
```
df.shape
# Size of data after dropping
```

[381]:
```
(8885, 13)
```

6 5 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 than that
```

[382]:
```
df.head(2)
```

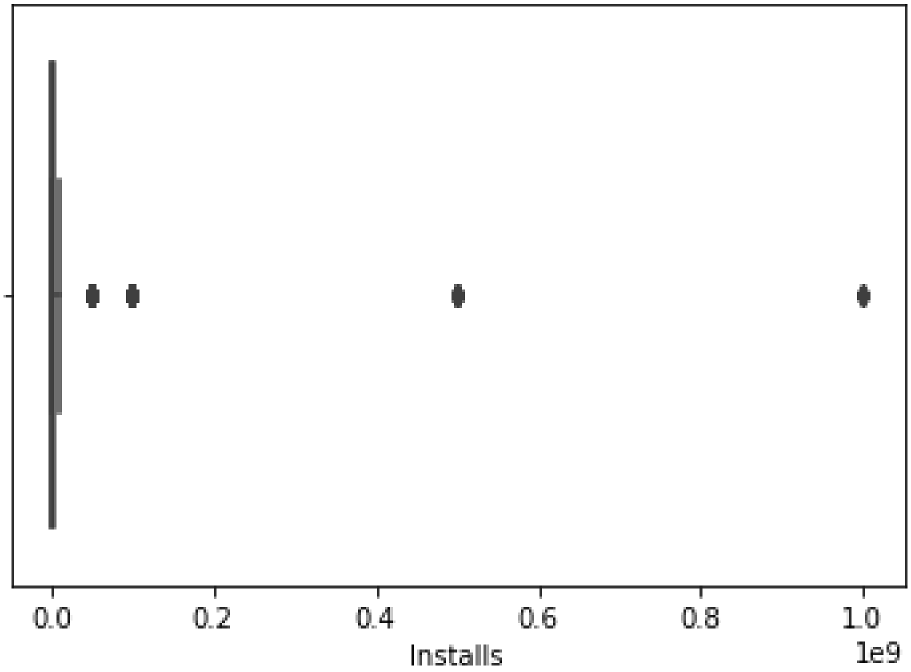
[382]:
```
                                             App        Category  Rating  \
0  Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                             Coloring book moana  ART_AND_DESIGN     3.9

   Reviews     Size  Installs  Type  Price Content Rating  \
0    159.0  19000.0     10000  Free    0.0       Everyone
1    967.0  14000.0    500000  Free    0.0       Everyone

                      Genres      Last Updated Current Ver   Android Ver
0               Art & Design   January 7, 2018       1.0.0  4.0.3 and up
1  Art & Design;Pretend Play  January 15, 2018       2.0.0  4.0.3 and up
```

[383]:
```
sns.boxplot('Installs',data=df)
plt.show()
```

```
/usr/local/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



[384]:
```
# divide the data in percentiles
df['Installs'].quantile([0.1,.25,.5,.7,.9,.95,.99])
```

[384]:
```
0.10         1000.0
0.25        10000.0
0.50       500000.0
0.70      1000000.0
0.90     10000000.0
0.95     10000000.0
0.99    100000000.0
Name: Installs, dtype: float64
```

[385]:
```
 # remove very high installation value
```

[386]:
```
# Installation with very high value to be removed
df.drop(df[df['Installs']>=100000000.0].index,inplace=True)
```

[387]:
```
df.shape
```

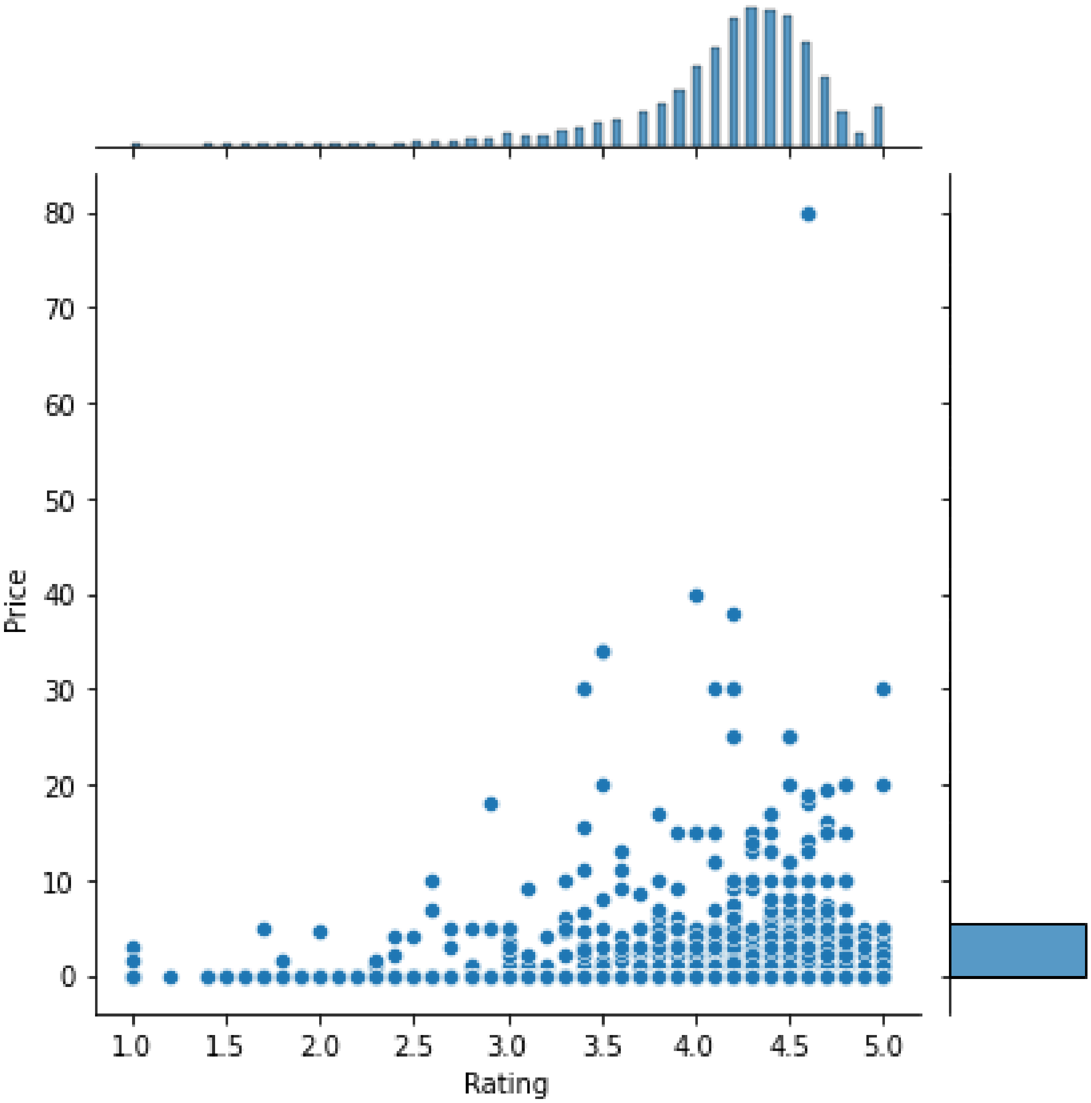
```
[387]: (8743, 13)
```

7 6 Bivariate Analysis

```
7. Bivariate analysis: Let's look at how the available predictors relate to the variable of in
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?
    Make scatter plot/joinplot for Rating vs. Size
    Are heavier apps rated better?
    Make scatter plot/joinplot for Rating vs. Reviews
    Does more review mean a better rating always?
    Make boxplot for Rating vs. Content Rating
    Is there any difference in the ratings? Are some types liked better?
    Make boxplot for Ratings vs. Category
    Which genre has the best ratings?
```

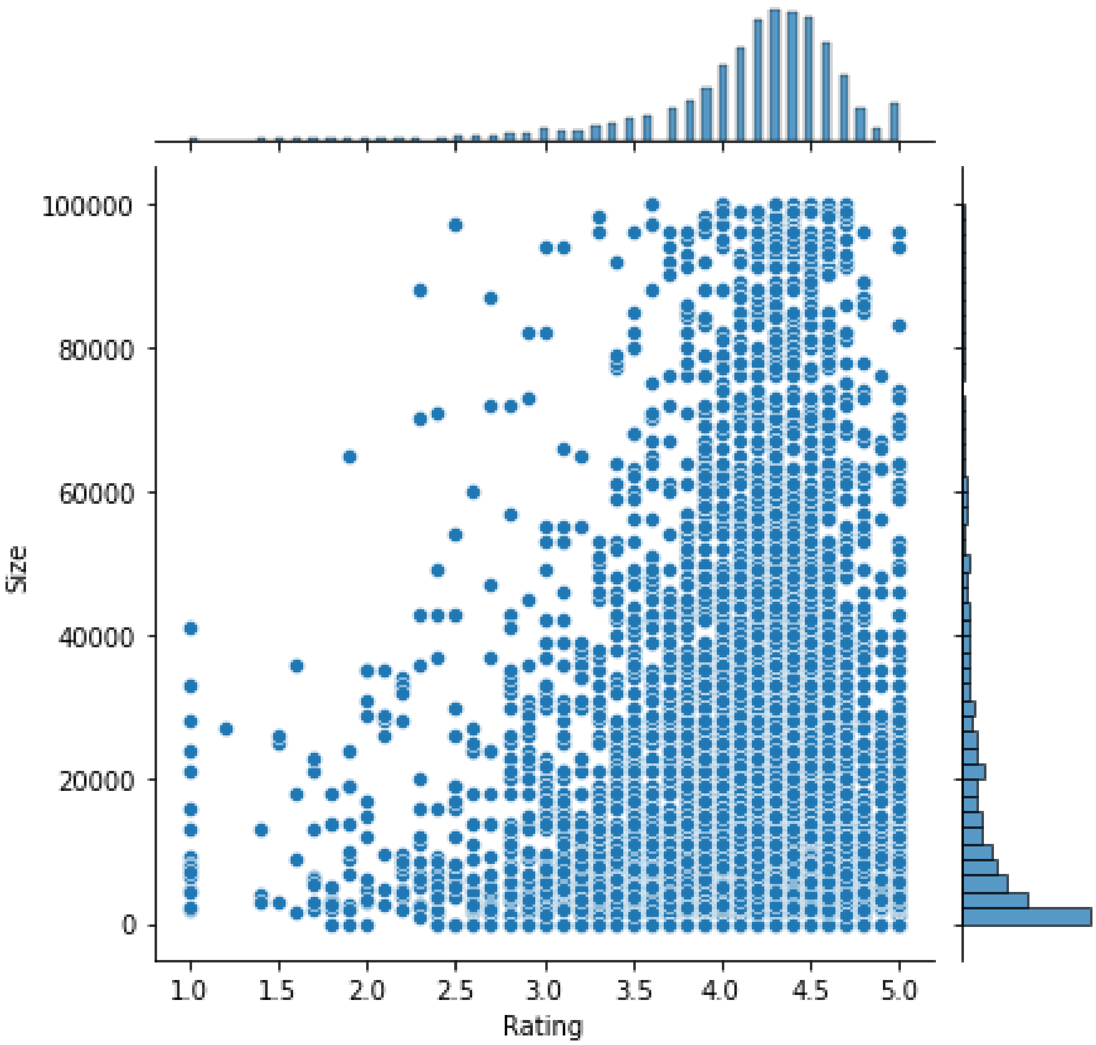
```
[388]: sns.jointplot(x='Rating',y='Price',data=df)
```

```
[388]: <seaborn.axisgrid.JointGrid at 0x7f30a023f450>
```



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

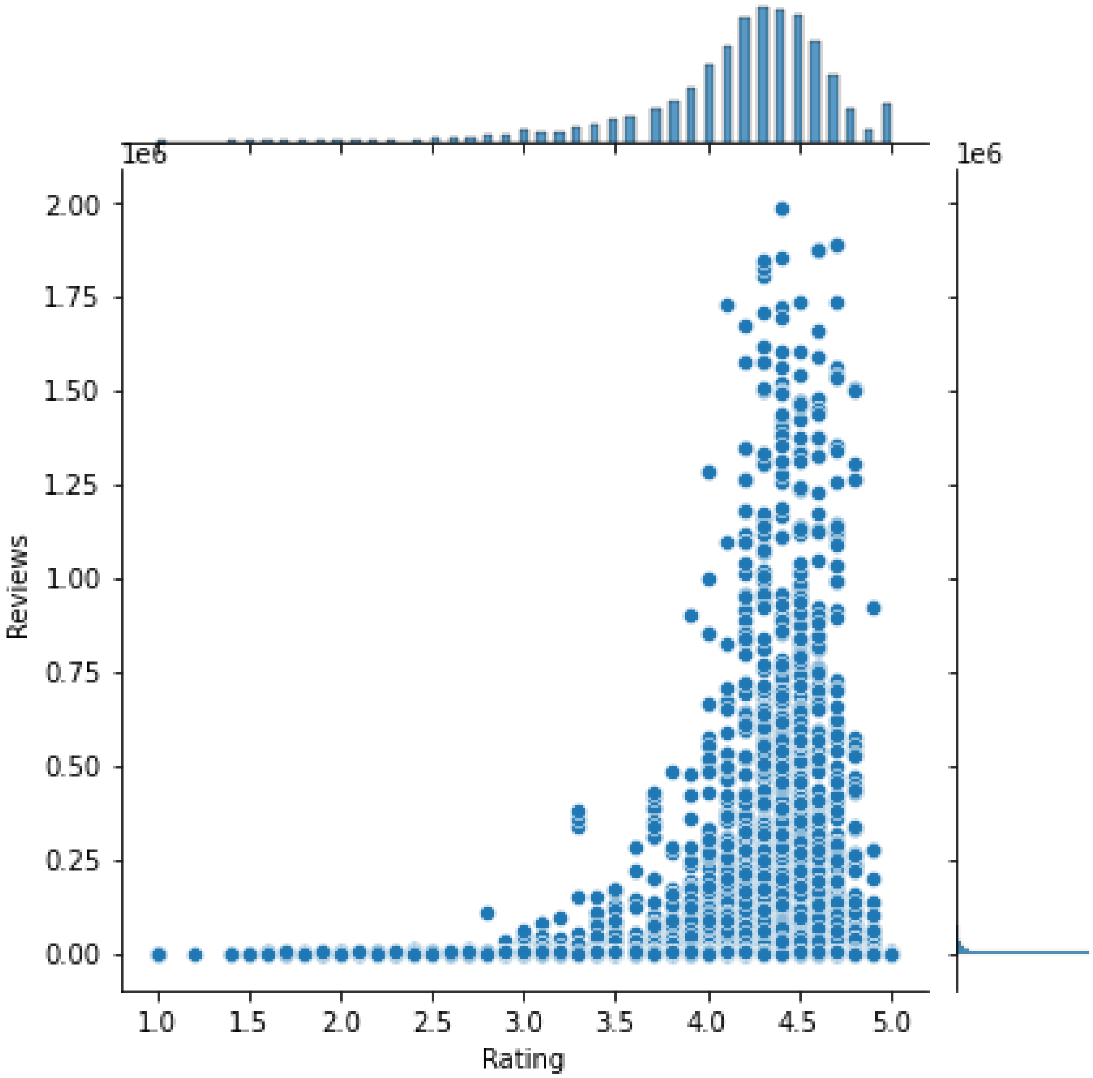
```
[389]: <seaborn.axisgrid.JointGrid at 0x7f309bc1d6d0>
```



```
[390]: # Based on above 2 graphs paid apps have the highest of Ratings
```

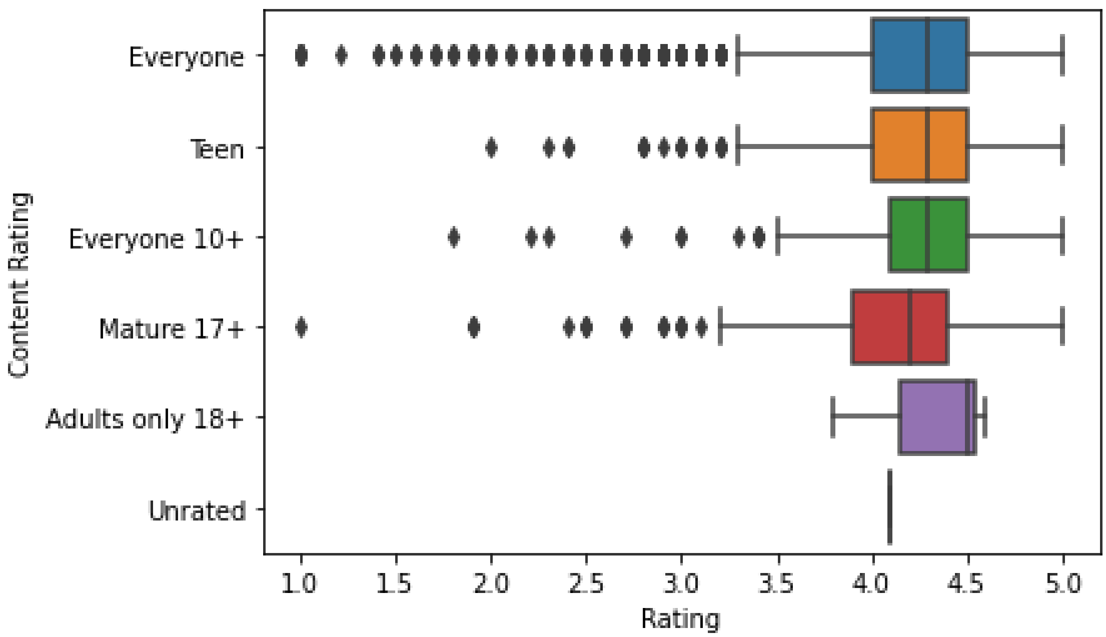
```
[391]: sns.jointplot(x='Rating',y='Reviews',data=df)
```

```
[391]: <seaborn.axisgrid.JointGrid at 0x7f30a88dbd90>
```



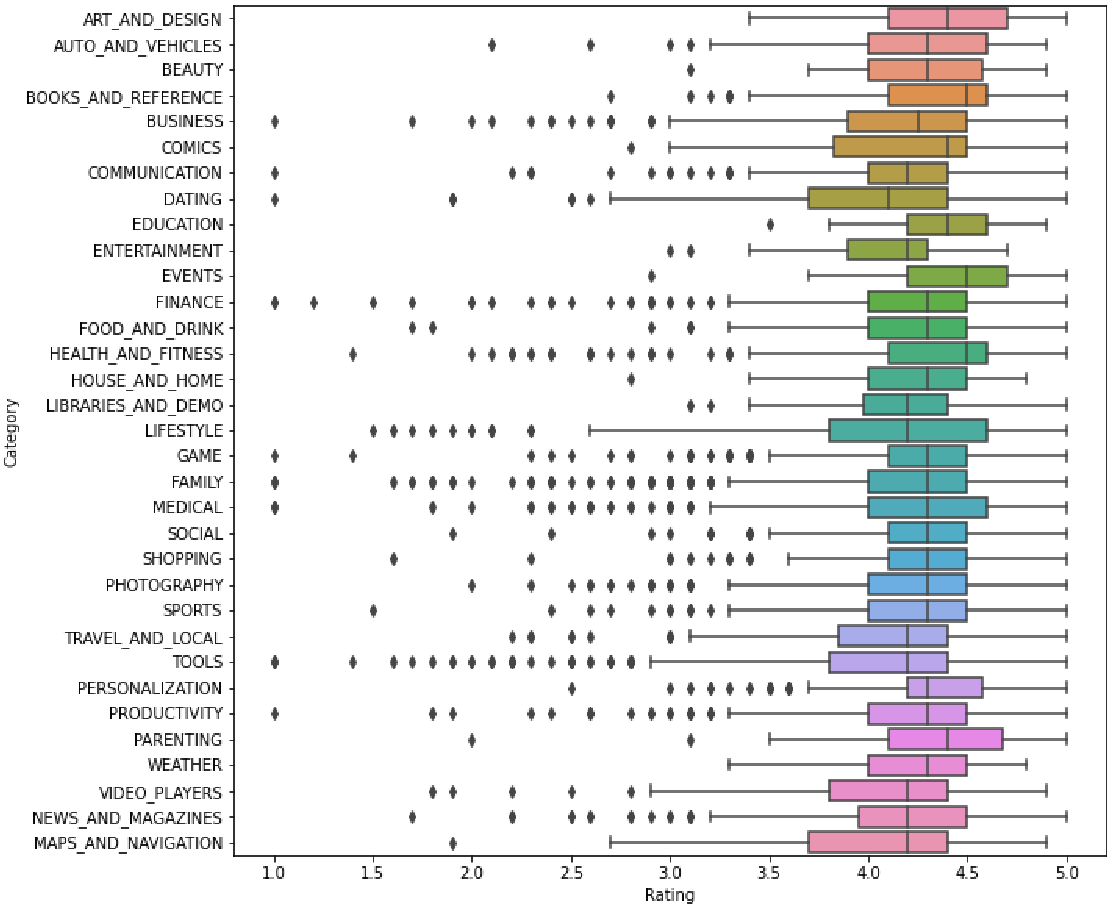
[392]:
```
# The plots show a positive linear relationship; as the Size increases the␣
↪Ratings increases. This stats the heavier apps are rated better
```

[393]:
```
sns.boxplot(x='Rating',y='Content Rating',data=df)
plt.show()
```



[394]:
```
#The above plot shows the apps for Everyone is worst rated as it contain the␣
↪highest number of outliers followed by apps for Mature 17+ and Everyone 10+␣
↪along with Teen. The catergory Adults only 18+ is rated better and falls␣
↪under most liked type
```

[395]:
```
plt.figure(figsize=(10,10))
sns.boxplot(x='Rating',y='Category',data=df)
plt.show()
```



[396]:
```
# Game and Family category are the most appearances for application in google play store
```

8 7 Machine Learning

8. Data preprocessing
For the steps below, create a copy of the dataframe to make all the edits. Name it inp1.
Reviews and Install have some values that are still relatively very high. Before building a li
regression model, you need to reduce the skew. Apply log transformation (np.log1p) to Reviews
and Installs.
Drop columns App, Last Updated, Current Ver, and Android Ver. These variables are not useful
for our task.
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.
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.

```
11 . Model building
Use linear regression as the technique
Report the R2 on the train set
12. Make predictions on test set and report R2.
```

[397]:

```
# extract the features
# data encoding
# transformation
# Train test
```

[398]:

```
# For the steps below, create a copy of the dataframe to make all the edits.
 ↪Name it inp1.
inp1 = df.copy()
```

[399]:

```
inp1.skew()
```

[399]:

```
Rating      -1.777070
Reviews      4.149314
Size         1.643761
Installs     4.401787
Price       16.495052
dtype: float64
```

[400]:

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

[401]:

```
reviewskew.skew()
```

[401]:

```
-0.19114430925837925
```

[402]:

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

[402]:

```
0           10000
1          500000
2         5000000
3        50000000
4          100000
           …
10834         500
10836        5000
10837         100
10839        1000
10840    10000000
Name: Installs, Length: 8743, dtype: int64
```

[403]:

```
installsskew.skew()
```

[403]:
```
-0.46306064681638154
```

[404]:
```
inp1.head()
```

[404]:
```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite – FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7
3                              Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

     Reviews     Size  Installs  Type  Price Content Rating  \
0   5.075174  19000.0     10000  Free    0.0       Everyone
1   6.875232  14000.0    500000  Free    0.0       Everyone
2  11.379520   8700.0   5000000  Free    0.0       Everyone
3  12.281389  25000.0  50000000  Free    0.0           Teen
4   6.875232   2800.0    100000  Free    0.0       Everyone

                      Genres      Last Updated         Current Ver  \
0               Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
2               Art & Design    August 1, 2018               1.2.4
3               Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up
```

[405]:
```
inp1.drop(["Last Updated","Current Ver","Android␣
↪Ver","App","Type"],axis=1,inplace=True)
```

[406]:
```
inp1.head()
```

[406]:
```
         Category  Rating    Reviews     Size  Installs  Price Content Rating  \
0  ART_AND_DESIGN     4.1   5.075174  19000.0     10000    0.0       Everyone
1  ART_AND_DESIGN     3.9   6.875232  14000.0    500000    0.0       Everyone
2  ART_AND_DESIGN     4.7  11.379520   8700.0   5000000    0.0       Everyone
3  ART_AND_DESIGN     4.5  12.281389  25000.0  50000000    0.0           Teen
4  ART_AND_DESIGN     4.3   6.875232   2800.0    100000    0.0       Everyone

                      Genres
0               Art & Design
1  Art & Design;Pretend Play
```

```
2                Art & Design
3                Art & Design
4    Art & Design;Creativity
```

[407]:

```
inp1.shape
```

[407]:

```
(8743, 8)
```

[408]:

```
inp2 = inp1
```

[409]:

```
inp2.head()
```

[409]:

```
         Category  Rating    Reviews     Size  Installs  Price Content Rating  \
0  ART_AND_DESIGN     4.1   5.075174  19000.0     10000    0.0       Everyone
1  ART_AND_DESIGN     3.9   6.875232  14000.0    500000    0.0       Everyone
2  ART_AND_DESIGN     4.7  11.379520   8700.0   5000000    0.0       Everyone
3  ART_AND_DESIGN     4.5  12.281389  25000.0  50000000    0.0           Teen
4  ART_AND_DESIGN     4.3   6.875232   2800.0    100000    0.0       Everyone

                     Genres
0              Art & Design
1  Art & Design;Pretend Play
2              Art & Design
3              Art & Design
4    Art & Design;Creativity
```

9 Reviews and Install have some values that are still relatively very high. Before building a linear regression model, you need to reduce the skew. hence column need log transformation

[410]:

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

[410]:

```
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)
```

[411]:

```
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()
```

[411]:

```
   Rating    Reviews     Size  Installs  Price Content Rating  \
0     4.1   5.075174  19000.0     10000    0.0       Everyone
1     3.9   6.875232  14000.0    500000    0.0       Everyone
2     4.7  11.379520   8700.0   5000000    0.0       Everyone
3     4.5  12.281389  25000.0  50000000    0.0           Teen
4     4.3   6.875232   2800.0    100000    0.0       Everyone

                      Genres  Category_ART_AND_DESIGN  \
0               Art & Design                        1
1  Art & Design;Pretend Play                        1
2               Art & Design                        1
3               Art & Design                        1
4    Art & Design;Creativity                        1

   Category_AUTO_AND_VEHICLES  Category_BEAUTY  …  Category_PERSONALIZATION  \
0                           0                0  …                         0
1                           0                0  …                         0
2                           0                0  …                         0
3                           0                0  …                         0
4                           0                0  …                         0

   Category_PHOTOGRAPHY  Category_PRODUCTIVITY  Category_SHOPPING  \
0                     0                      0                  0
1                     0                      0                  0
2                     0                      0                  0
3                     0                      0                  0
4                     0                      0                  0

   Category_SOCIAL  Category_SPORTS  Category_TOOLS  \
0                0                0               0
1                0                0               0
2                0                0               0
3                0                0               0
4                0                0               0

   Category_TRAVEL_AND_LOCAL  Category_VIDEO_PLAYERS  Category_WEATHER
0                          0                       0                 0
1                          0                       0                 0
```

```
2                            0                        0                  0
3                            0                        0                  0
4                            0                        0                  0

[5 rows x 40 columns]
```

[412]:
```
#get unique values in Column "Genres"
inp2["Genres"].unique()
```

[412]:
```
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;Music & Video', 'Entertainment;Brain Games',
       'Entertainment;Creativity', 'Events', 'Finance', 'Food & Drink',
       'Health & Fitness', 'House & Home', 'Libraries & Demo',
       'Lifestyle', 'Lifestyle;Pretend Play', 'Card', 'Casual',
       'Casual;Pretend Play', 'Puzzle', 'Action', 'Arcade', 'Music',
       'Word', 'Racing', 'Casual;Creativity', 'Sports', 'Simulation',
       'Board', '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', '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',
       '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)
```

```
[413]: 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]
```

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

```
[414]: 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', 'Casual;Pretend Play', 'Puzzle',
              'Action', 'Arcade', 'Music', 'Word', 'Racing', 'Sports',
              'Simulation', 'Board', 'Role Playing', 'Adventure', 'Strategy',
              'Trivia', 'Educational;Education', 'Racing;Action & Adventure',
              'Medical', 'Social', 'Shopping', 'Photography', 'Travel & Local',
              'Tools', 'Personalization', 'Productivity', 'Parenting', 'Weather',
              'Video Players & Editors', 'News & Magazines', 'Maps & Navigation',
              'Educational', 'Casino'], dtype=object)
```

```
[415]: 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)
```

```
[416]: inp2.head()
```

```
[416]:    Rating    Reviews     Size  Installs  Price Content Rating  \
       0     4.1   5.075174  19000.0     10000    0.0       Everyone
       1     3.9   6.875232  14000.0    500000    0.0       Everyone
       2     4.7  11.379520   8700.0   5000000    0.0       Everyone
       3     4.5  12.281389  25000.0  50000000    0.0           Teen
       4     4.3   6.875232   2800.0    100000    0.0       Everyone

          Category_ART_AND_DESIGN  Category_AUTO_AND_VEHICLES  Category_BEAUTY  \
       0                        1                           0                0
       1                        1                           0                0
       2                        1                           0                0
       3                        1                           0                0
       4                        1                           0                0

          Category_BOOKS_AND_REFERENCE  …  Genres_Simulation  Genres_Social  \
       0                             0  …                  0              0
       1                             0  …                  0              0
       2                             0  …                  0              0
       3                             0  …                  0              0
       4                             0  …                  0              0

          Genres_Sports  Genres_Strategy  Genres_Tools  Genres_Travel & Local  \
       0              0                0             0                      0
       1              0                0             0                      0
       2              0                0             0                      0
       3              0                0             0                      0
       4              0                0             0                      0

          Genres_Trivia  Genres_Video Players & Editors  Genres_Weather  Genres_Word
       0              0                               0               0            0
       1              0                               0               0            0
       2              0                               0               0            0
       3              0                               0               0            0
       4              0                               0               0            0

       [5 rows x 93 columns]
```

[417]:

```
inp2.shape
```

[417]:

```
(8743, 93)
```

[418]:

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

[418]:

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

[419]:

```
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()
```

[419]:

```
   Rating    Reviews     Size  Installs  Price  Category_ART_AND_DESIGN  \
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
3     4.5  12.281389  25000.0  50000000    0.0                        1
4     4.3   6.875232   2800.0    100000    0.0                        1

   Category_AUTO_AND_VEHICLES  Category_BEAUTY  Category_BOOKS_AND_REFERENCE  \
0                           0                0                             0
1                           0                0                             0
2                           0                0                             0
3                           0                0                             0
4                           0                0                             0

   Category_BUSINESS  …  Genres_Trivia  Genres_Video Players & Editors  \
0                  0  …              0                               0
1                  0  …              0                               0
2                  0  …              0                               0
3                  0  …              0                               0
4                  0  …              0                               0

   Genres_Weather  Genres_Word  Content Rating_Adults only 18+  \
0               0            0                               0
1               0            0                               0
2               0            0                               0
3               0            0                               0
4               0            0                               0

   Content Rating_Everyone  Content Rating_Everyone 10+  \
0                        1                            0
1                        1                            0
2                        1                            0
3                        0                            0
4                        1                            0

   Content Rating_Mature 17+  Content Rating_Teen  Content Rating_Unrated
0                          0                    0                       0
1                          0                    0                       0
2                          0                    0                       0
3                          0                    1                       0
4                          0                    0                       0

[5 rows x 98 columns]
```

[420]:

```
inp2.shape
```

[420]:

```
(8743, 98)
```

10 Model Building

[421]:

```
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
```

[422]:

```
# Train test split and apply 70-30 split. Name the new dataframes df_train and␣
 ↪df_test.

# Separate the dataframes into X_train, y_train, X_test, and y_test
```

[423]:

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

x_train, x_test, y_train, y_test = tts(X,y, test_size=0.3, random_state=5)
```

[424]:

```
reg_all = LR()
reg_all.fit(x_train,y_train)
```

[424]:

```
LinearRegression()
```

[425]:

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

```
The R2 value of the Training Set is : 0.083
```

[426]:

```
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 : 0.05
```