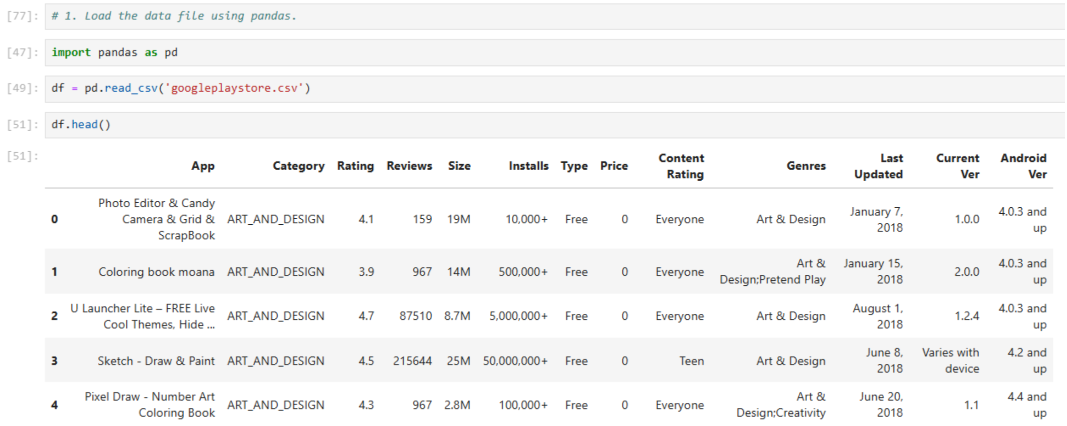
Google App Store Prediction Procedures

1. Load the data file using pandas.

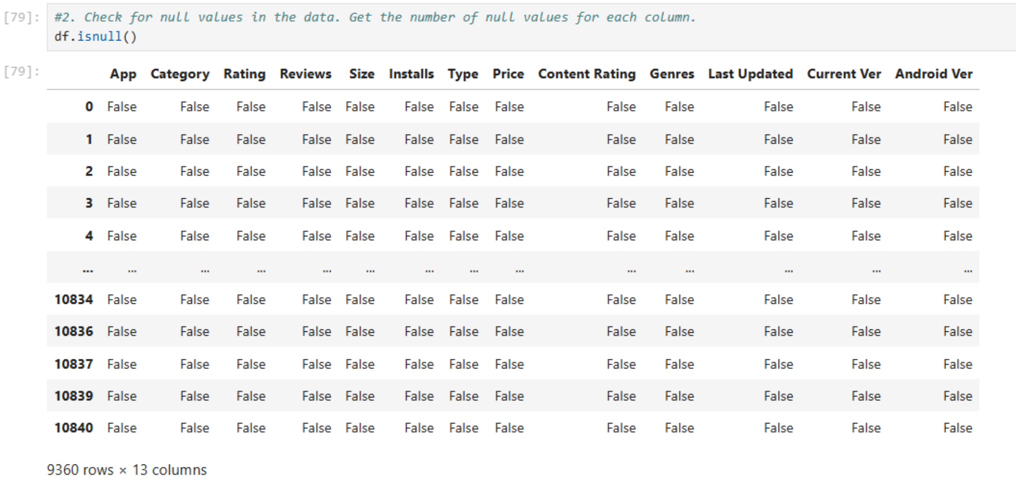


```
[77]: # 1. Load the data file using pandas.

[47]: import pandas as pd

[49]: df = pd.read_csv('googleplaystore.csv')

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

[51]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

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



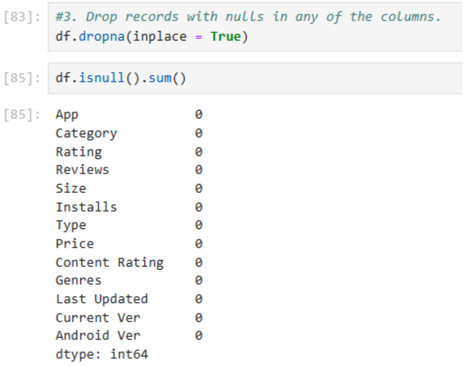
```
[79]: #2. Check for null values in the data. Get the number of null values for each column.
      df.isnull()
```

[79]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
10834	False	False	False	False	False	False	False	False	False	False	False	False	False
10836	False	False	False	False	False	False	False	False	False	False	False	False	False
10837	False	False	False	False	False	False	False	False	False	False	False	False	False
10839	False	False	False	False	False	False	False	False	False	False	False	False	False
10840	False	False	False	False	False	False	False	False	False	False	False	False	False

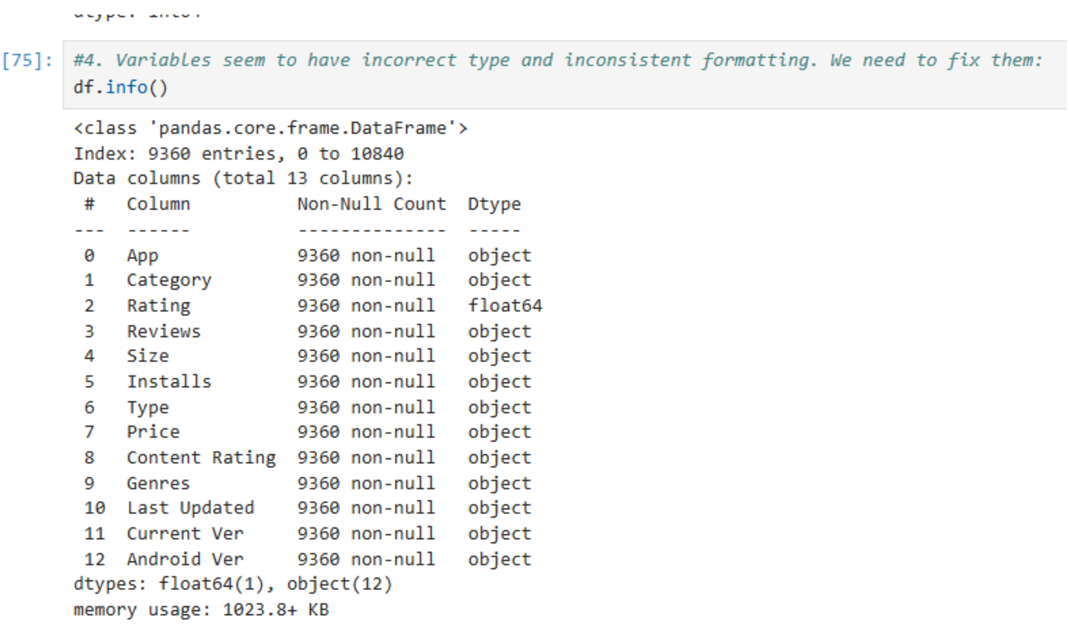
9360 rows × 13 columns

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



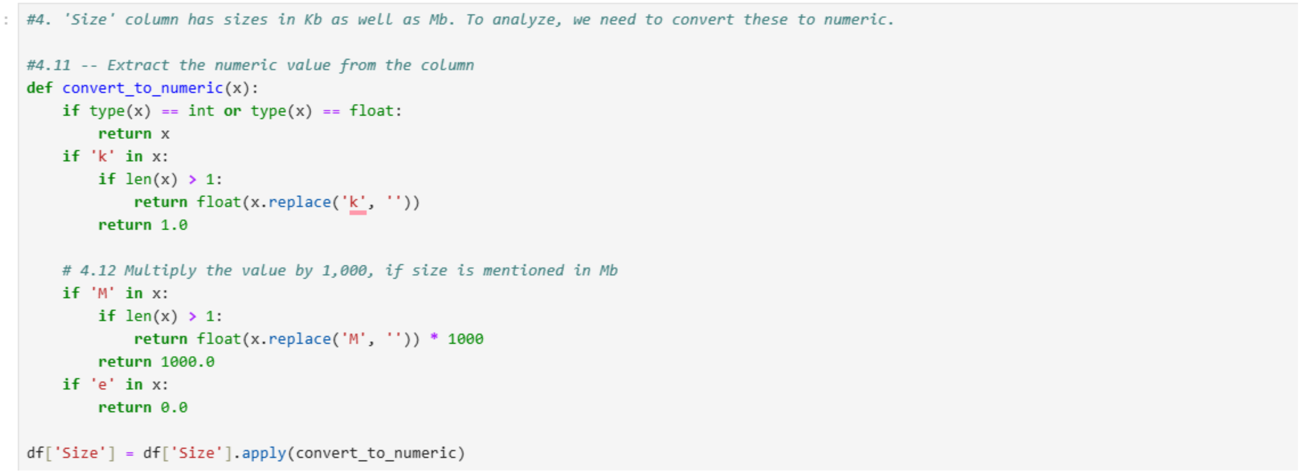
```
[83]: #3. Drop records with nulls in any of the columns.
      df.dropna(inplace = True)

[85]: df.isnull().sum()

[85]: 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
```

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



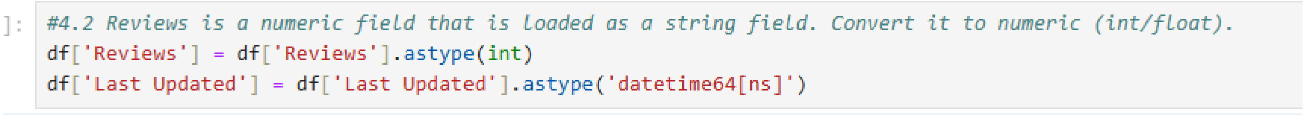
```
dtype: int64

[75]: #4. Variables seem to have incorrect type and inconsistent formatting. We need to fix them:
      df.info()

      <class 'pandas.core.frame.DataFrame'>
      Index: 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   object 
       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(1), object(12)
      memory usage: 1023.8+ KB
```

1. Size column has sizes in Kb as well as Mb. To analyze, you'll need to convert these to numeric.
    1. Extract the numeric value from the column
    2. Multiply the value by 1,000, if size is mentioned in Mb



```
: #4. 'Size' column has sizes in Kb as well as Mb. To analyze, we need to convert these to numeric.

  #4.11 -- Extract the numeric value from the column
  def convert_to_numeric(x):
      if type(x) == int or type(x) == float:
          return x
      if 'k' in x:
          if len(x) > 1:
              return float(x.replace('k', ''))
          return 1.0

      # 4.12 Multiply the value by 1,000, if size is mentioned in Mb
      if 'M' in x:
          if len(x) > 1:
              return float(x.replace('M', '')) * 1000
          return 1000.0
      if 'e' in x:
          return 0.0

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

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

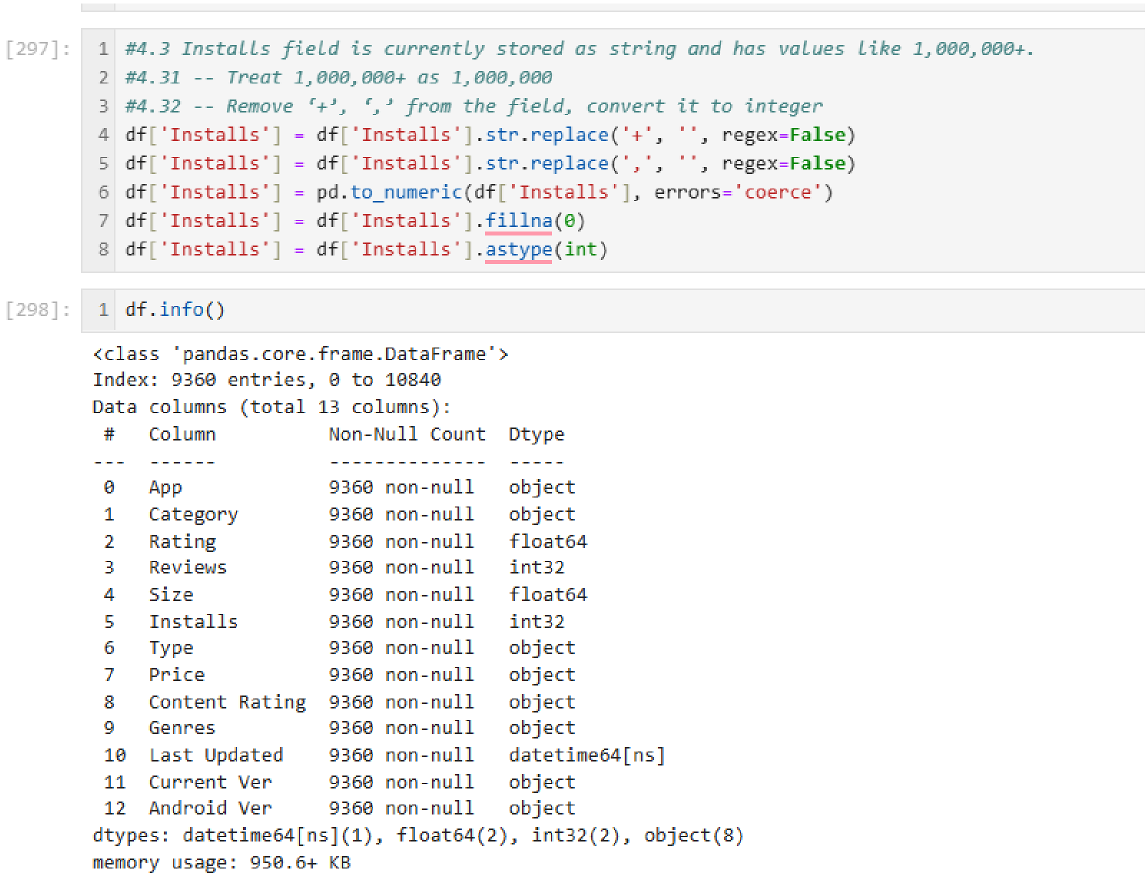


```
]: #4.2 Reviews is a numeric field that is loaded as a string field. Convert it to numeric (int/float).
   df['Reviews'] = df['Reviews'].astype(int)
   df['Last Updated'] = df['Last Updated'].astype('datetime64[ns]')
```

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

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

    2. remove ‘+’, ‘,’ from the field, convert it to integer



[297]:

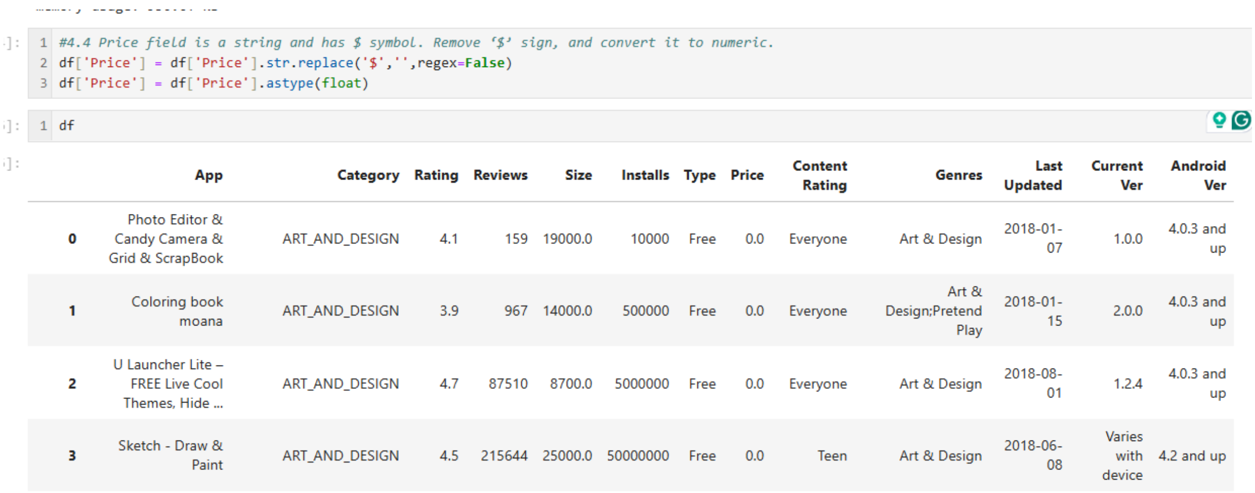
```
#4.3 Installs field is currently stored as string and has values like 1,000,000+.
#4.31 -- Treat 1,000,000+ as 1,000,000
#4.32 -- Remove ‘+’, ‘,’ from the field, convert it to integer
df['Installs'] = df['Installs'].str.replace('+', '', regex=False)
df['Installs'] = df['Installs'].str.replace(',', '', regex=False)
df['Installs'] = pd.to_numeric(df['Installs'], errors='coerce')
df['Installs'] = df['Installs'].fillna(0)
df['Installs'] = df['Installs'].astype(int)
```

[298]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 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   int32
 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   datetime64[ns]
 11  Current Ver     9360 non-null   object
 12  Android Ver     9360 non-null   object
dtypes: datetime64[ns](1), float64(2), int32(2), object(8)
memory usage: 950.6+ KB
```

4. Price field is a string and has $ symbol. Remove ‘$’ sign, and convert it to numeric.



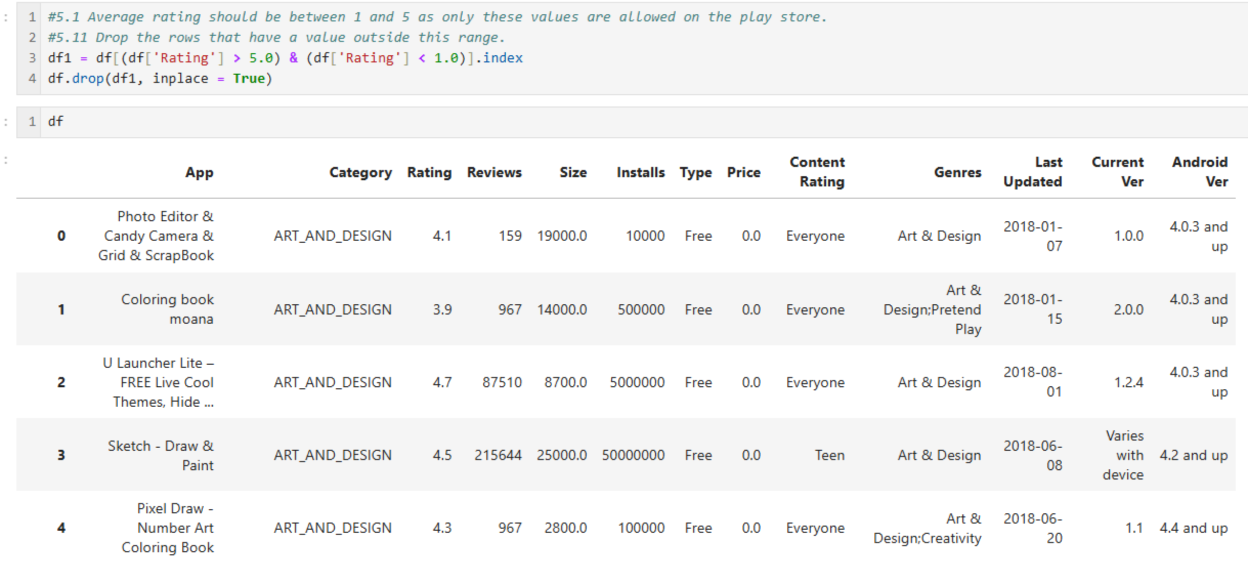
```
#4.4 Price field is a string and has $ symbol. Remove ‘$’ sign, and convert it to numeric.
df['Price'] = df['Price'].str.replace('$','',regex=False)
df['Price'] = df['Price'].astype(float)
```

```
df
```

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	2018-06-08	Varies with device	4.2 and up

5.1 Sanity checks:

1. Average rating should be between 1 and 5 as only these values are allowed on the play store. Drop the rows that have a value outside this range.

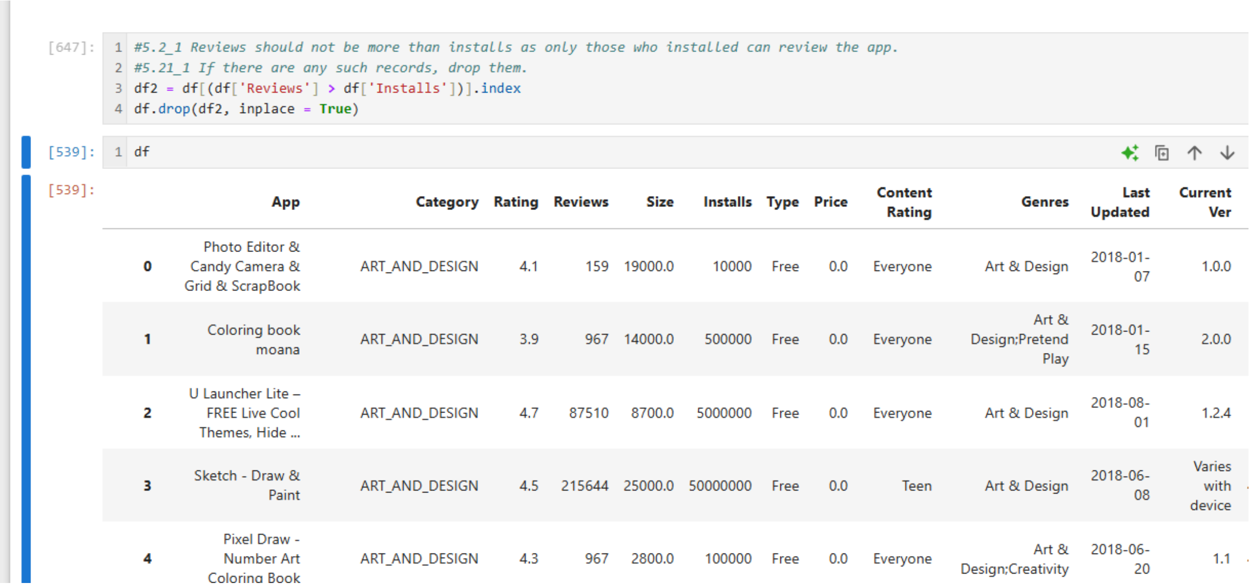


```
#5.1 Average rating should be between 1 and 5 as only these values are allowed on the play store.
#5.11 Drop the rows that have a value outside this range.
df1 = df[(df['Rating'] > 5.0) & (df['Rating'] < 1.0)].index
df.drop(df1, inplace = True)
```

```
df
```

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	2018-06-08	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100000	Free	0.0	Everyone	Art & Design;Creativity	2018-06-20	1.1	4.4 and up

2. Reviews should not be more than installs as only those who installed can review the app. If there are any such records, drop them.



[647]:

```
#5.2_1 Reviews should not be more than installs as only those who installed can review the app.
#5.21_1 If there are any such records, drop them.
df2 = df[(df['Reviews'] > df['Installs'])].index
df.drop(df2, inplace = True)
```

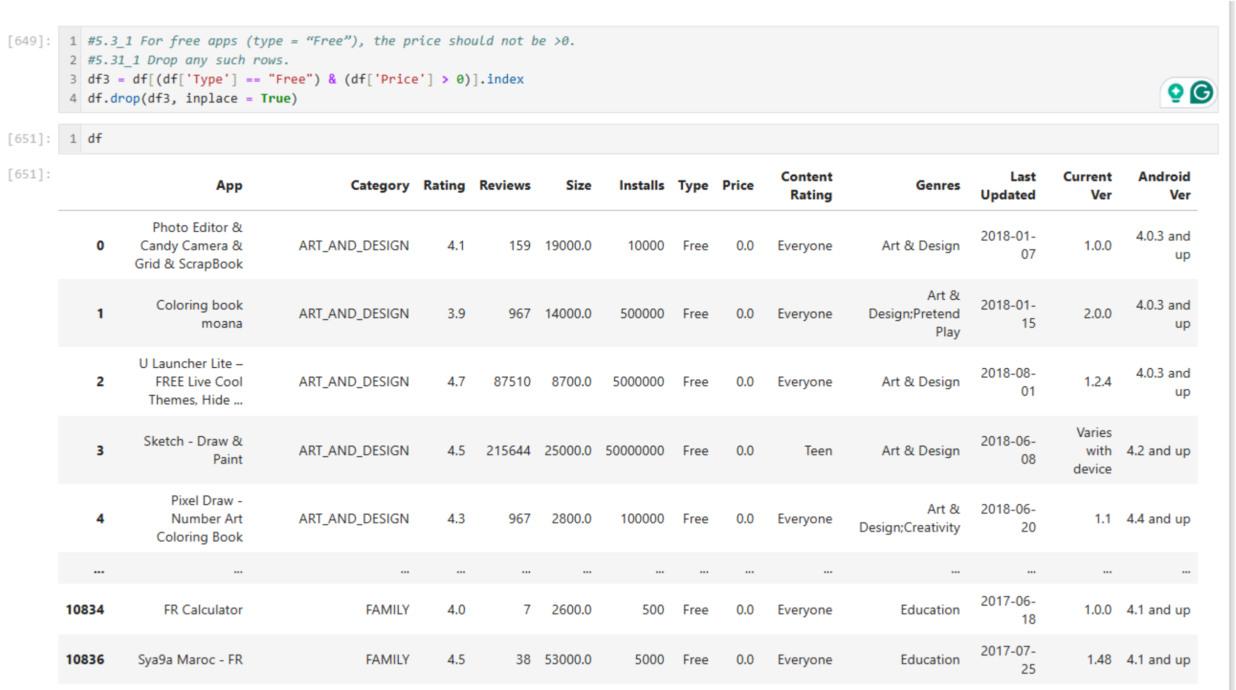
[539]:

```
df
```

[539]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	2018-06-08	Varies with device
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100000	Free	0.0	Everyone	Art & Design;Creativity	2018-06-20	1.1

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



```
[649]:
#5.3_1 For free apps (type = “Free”), the price should not be >0.
#5.31_1 Drop any such rows.
df3 = df[(df['Type'] == "Free") & (df['Price'] > 0)].index
df.drop(df3, inplace = True)
```

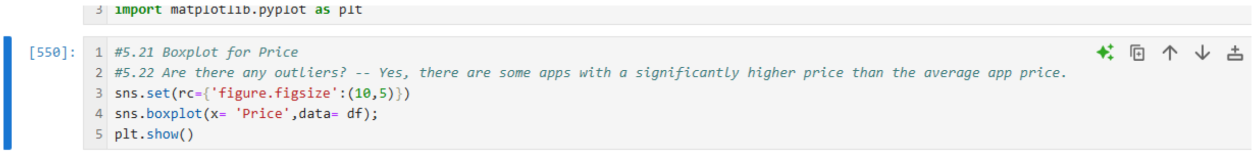
```
[651]:
df
```

[651]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	2018-06-08	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2800.0	100000	Free	0.0	Everyone	Art & Design;Creativity	2018-06-20	1.1	4.4 and up
...	...	...	...	...	...	...	...	...	...	...	...	...	...
10834	FR Calculator	FAMILY	4.0	7	2600.0	500	Free	0.0	Everyone	Education	2017-06-18	1.0.0	4.1 and up
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53000.0	5000	Free	0.0	Everyone	Education	2017-07-25	1.48	4.1 and up

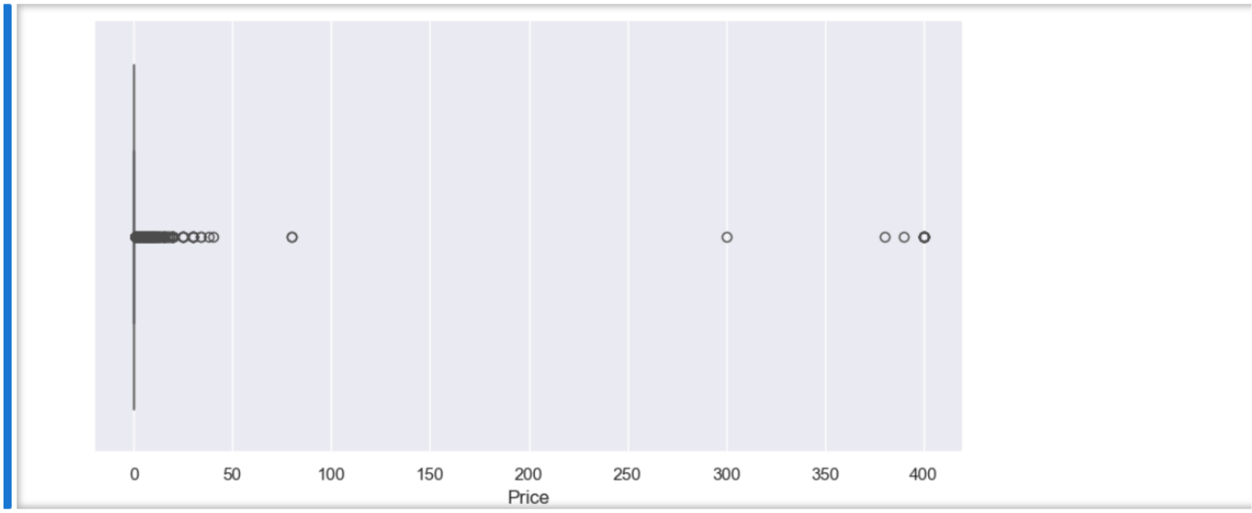
5.2 Performing univariate analysis:

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

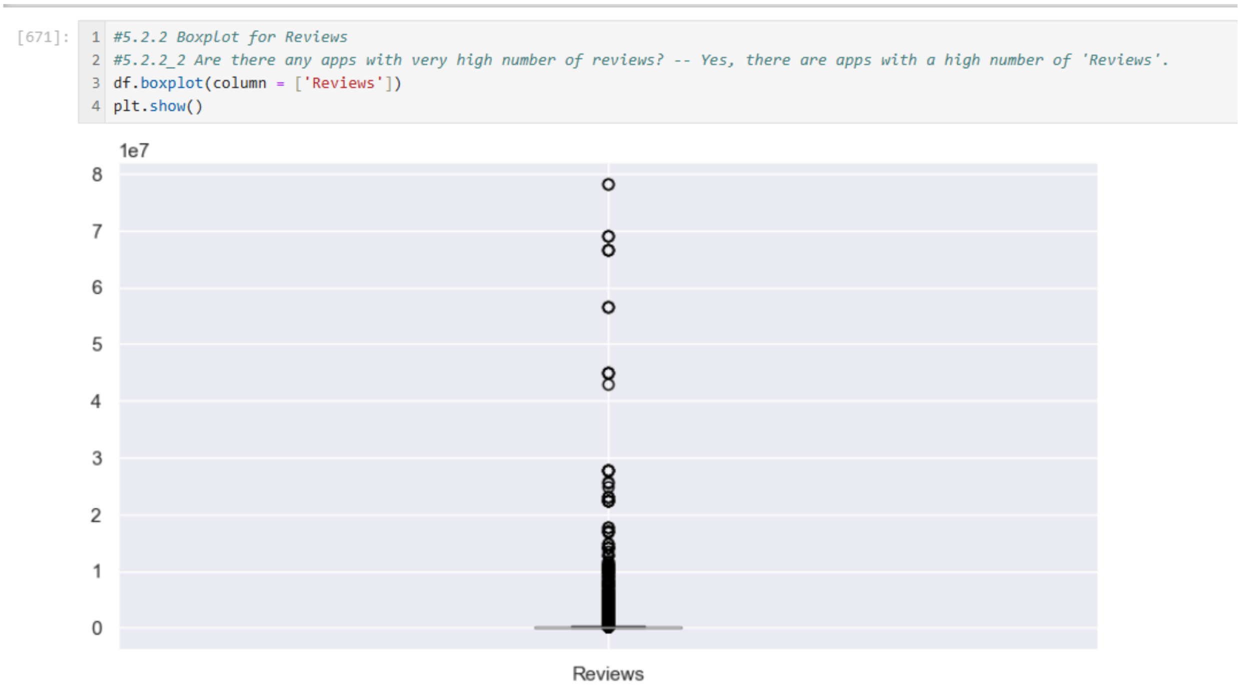


```
import matplotlib.pyplot as plt
```

```
[550]:
#5.21 Boxplot for Price
#5.22 Are there any outliers? -- Yes, there are some apps with a significantly higher price than the average app price.
sns.set(rc={'figure.figsize':(10,5)})
sns.boxplot(x= 'Price',data= df);
plt.show()
```

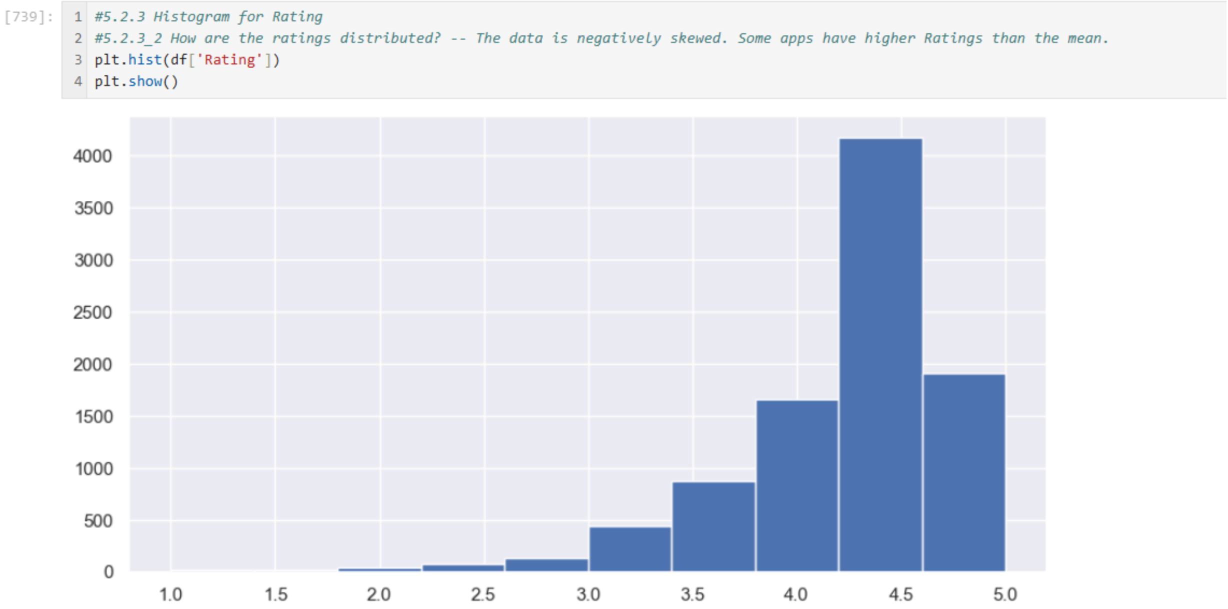


- Boxplot for Reviews
    - Are there any apps with very high number of reviews? Do the values seem right?



```
[671]:
#5.2.2 Boxplot for Reviews
#5.2.2_2 Are there any apps with very high number of reviews? -- Yes, there are apps with a high number of 'Reviews'.
df.boxplot(column = ['Reviews'])
plt.show()
```

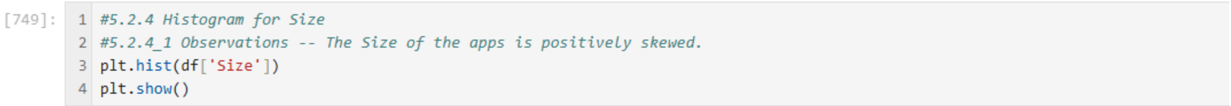
- Histogram for Rating
  - How are the ratings distributed? Is it more toward higher ratings?



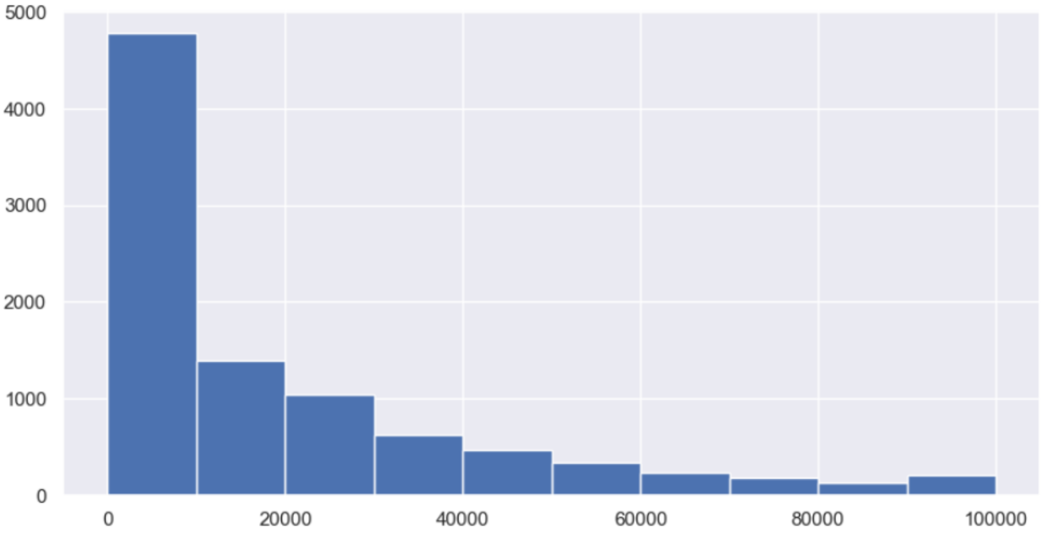
```
[739]:
#5.2.3 Histogram for Rating
#5.2.3_2 How are the ratings distributed? -- The data is negatively skewed. Some apps have higher Ratings than the mean.
plt.hist(df['Rating'])
plt.show()
```

- Histogram for Size

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

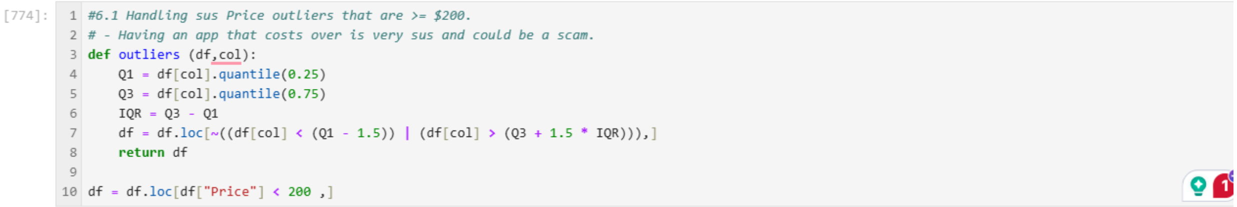


```
[749]:
#5.2.4 Histogram for Size
#5.2.4_1 Observations -- The Size of the apps is positively skewed.
plt.hist(df['Size'])
plt.show()
```

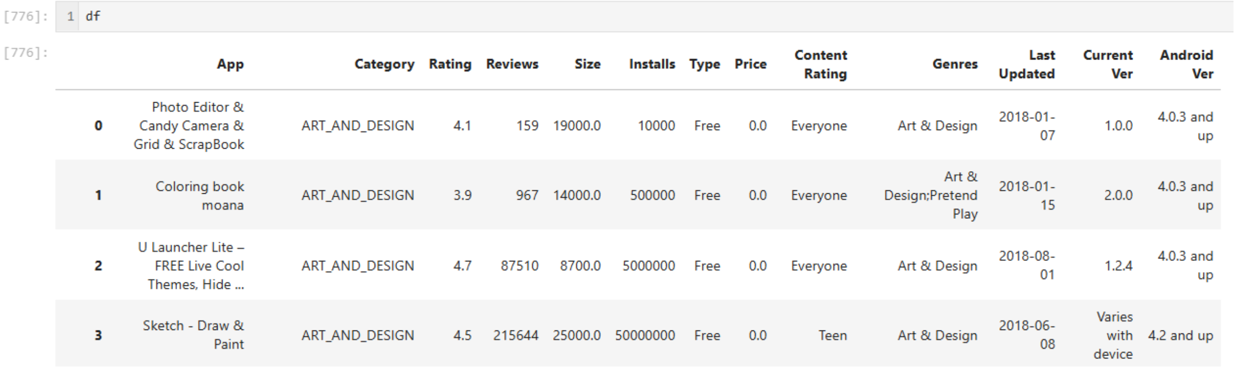


6. Outlier treatment:
    1. Price: From the box plot, it seems like there are some apps with very high price. A price of $200 for an application on the Play Store is very high and suspicious!
        1. Check out the records with very high price
            1. Is 200 indeed a high price?
        2. Drop these as most seem to be junk apps

Outlier Treatment



```
[774]:
#6.1 Handling sus Price outliers that are >= $200.
# - Having an app that costs over is very sus and could be a scam.
def outliers (df,col):
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    df = df.loc[~((df[col] < (Q1 - 1.5)) | (df[col] > (Q3 + 1.5 * IQR))),]
    return df

df = df.loc[df["Price"] < 200 ,]
```

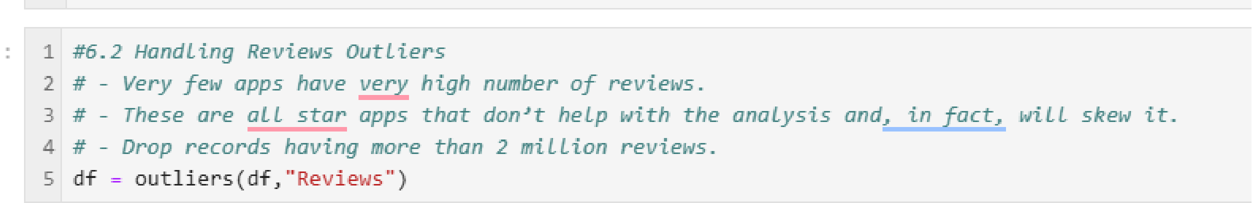


```
[776]:
df
```

[776]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8700.0	5000000	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	2018-06-08	Varies with device	4.2 and up

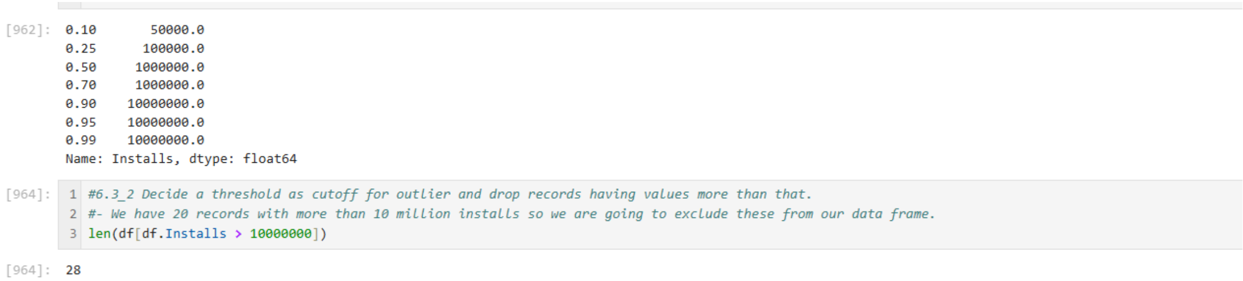
2. Reviews: Very few apps have very high number of reviews. These are all star apps that don’t help with the analysis and, in fact, will skew it. Drop records having more than 2 million reviews.



```
:  #6.2 Handling Reviews Outliers
   # - Very few apps have very high number of reviews.
   # - These are all star apps that don’t help with the analysis and, in fact, will skew it.
   # - Drop records having more than 2 million reviews.
   df = outliers(df,"Reviews")
```

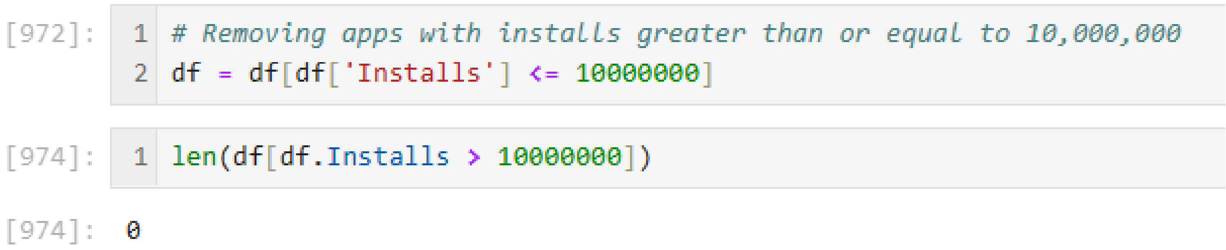
3. Installs: There seems to be some outliers in this field too. Apps having very high number of installs should be dropped from the analysis.

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



```
[962]: 0.10       50000.0
       0.25      100000.0
       0.50     1000000.0
       0.70     1000000.0
       0.90    10000000.0
       0.95    10000000.0
       0.99    10000000.0
       Name: Installs, dtype: float64

[964]: #6.3_2 Decide a threshold as cutoff for outlier and drop records having values more than that.
       #- We have 20 records with more than 10 million installs so we are going to exclude these from our data frame.
       len(df[df.Installs > 10000000])

[964]: 28
```

    2. Decide a threshold as cutoff for outlier and drop records having values more than that



```
[972]: # Removing apps with installs greater than or equal to 10,000,000
       df = df[df['Installs'] <= 10000000]

[974]: len(df[df.Installs > 10000000])

[974]: 0
```

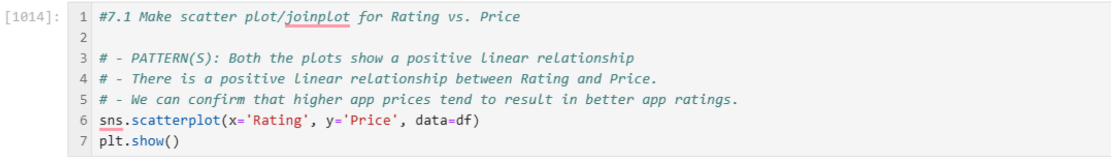
7. Bivariate analysis: Let’s look at how the available predictors relate to the variable of interest, i.e., our target variable rating. Make scatter plots (for numeric features) and box plots (for character features) to assess the relations between rating and the other features.

1. Make scatter plot/joinplot for Rating vs. Price

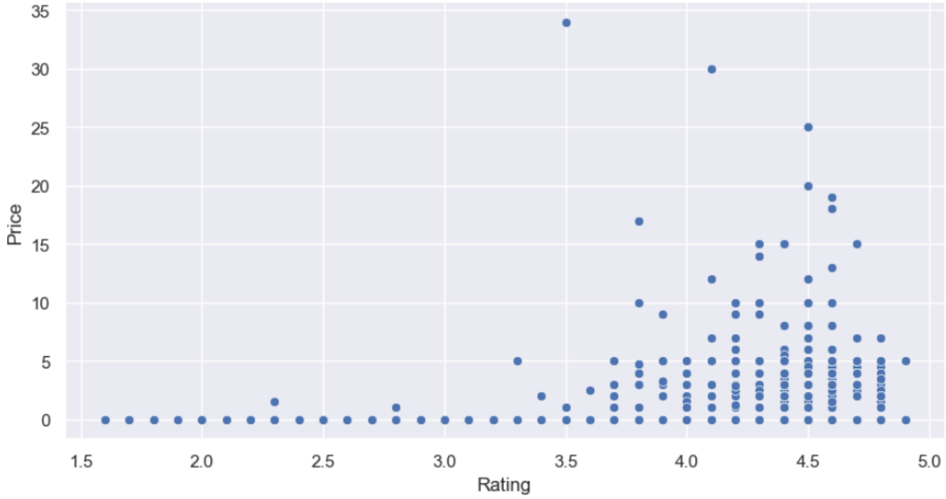
    1. What pattern do you observe? Does rating increase with price?

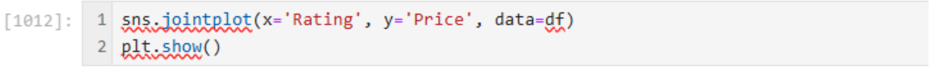
- There is a positive linear relationship between Rating and Price.

- We can confirm that higher app prices tend to result in better app ratings.

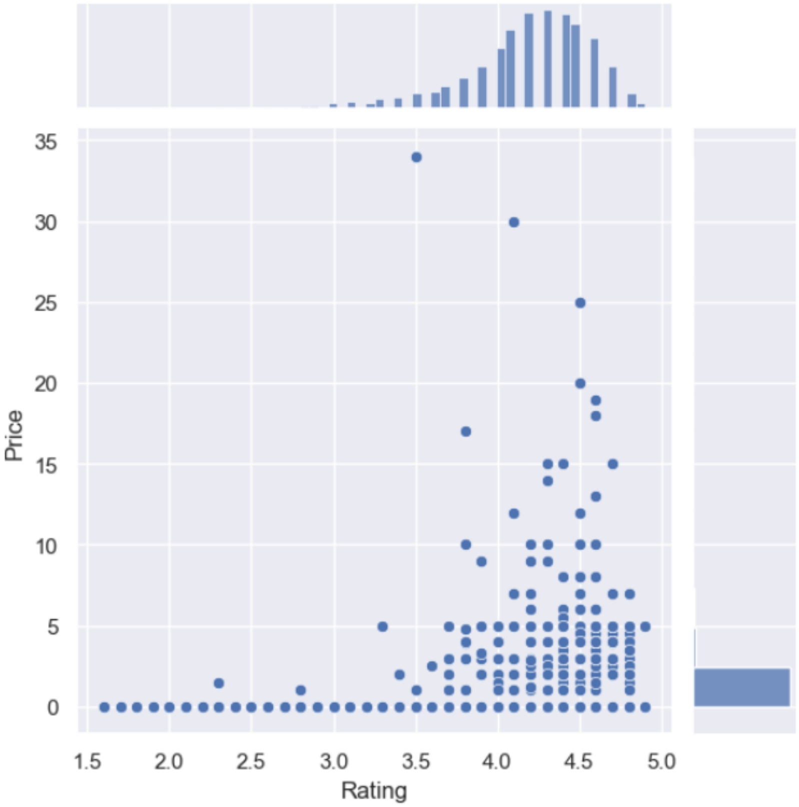


```
[1014]:
#7.1 Make scatter plot/joinplot for Rating vs. Price

# - PATTERN(S): Both the plots show a positive linear relationship
# - There is a positive linear relationship between Rating and Price.
# - We can confirm that higher app prices tend to result in better app ratings.
sns.scatterplot(x='Rating', y='Price', data=df)
plt.show()
```



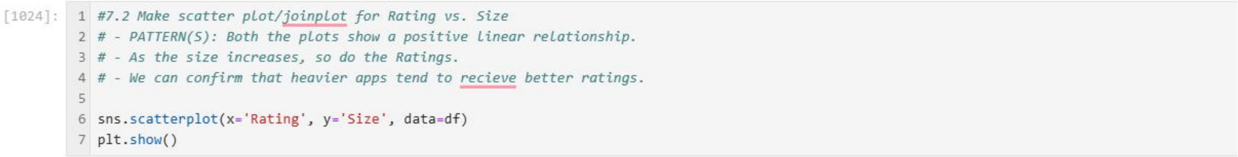


```
[1012]:
sns.jointplot(x='Rating', y='Price', data=df)
plt.show()
```

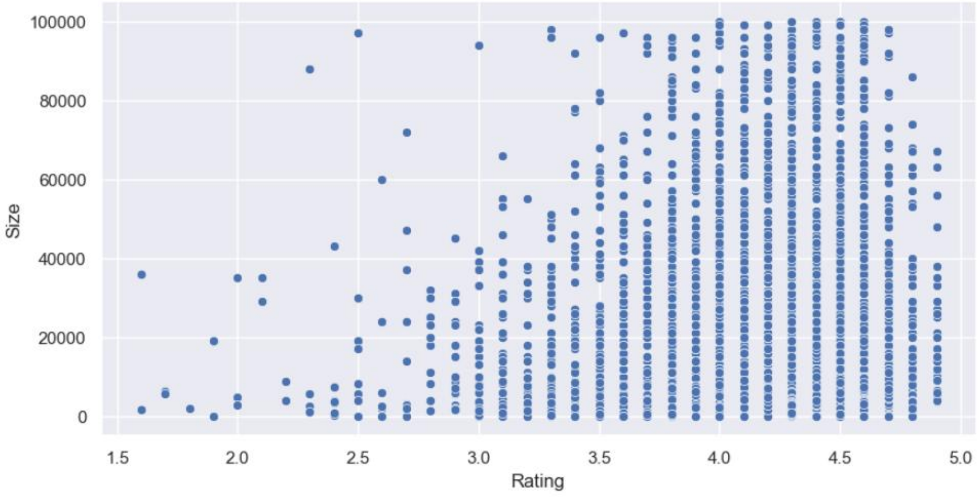


2. Make scatter plot/joinplot for Rating vs. Size

    1. Are heavier apps rated better? Yes, we can confirm that heavier apps tend to recieve better ratings.

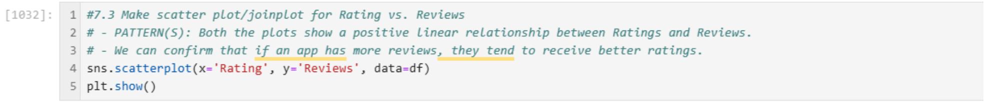


```
[1024]:
#7.2 Make scatter plot/joinplot for Rating vs. Size
# - PATTERN(S): Both the plots show a positive linear relationship.
# - As the size increases, so do the Ratings.
# - We can confirm that heavier apps tend to recieve better ratings.

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

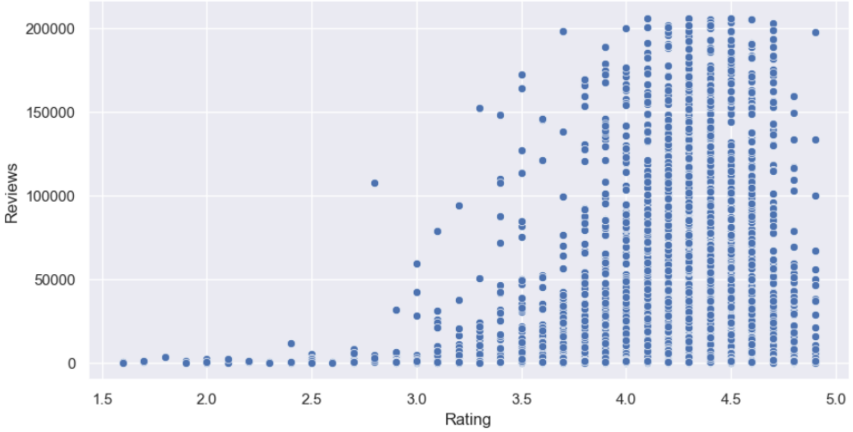


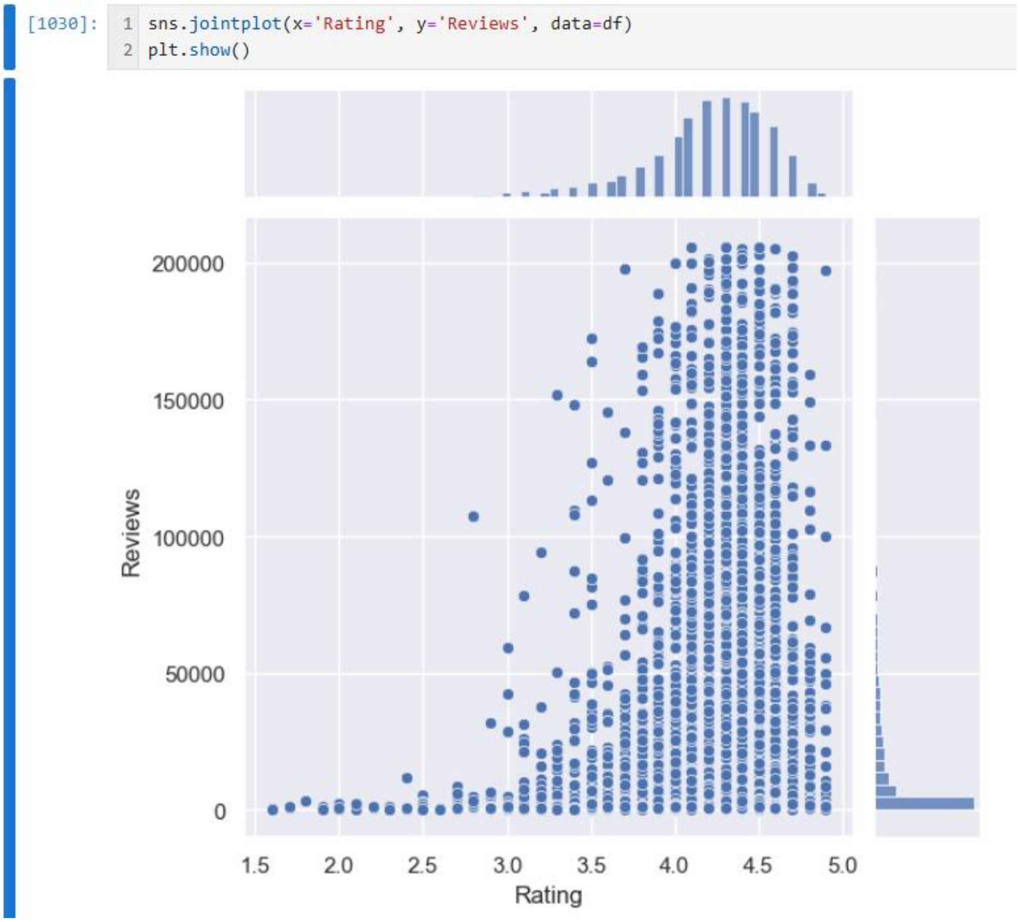
3. Make scatter plot/joinplot for Rating vs. Reviews

    1. Does more review mean a better rating always? We can confirm that if an app has more reviews, they tend to receive better ratings.



```
[1032]:
#7.3 Make scatter plot/joinplot for Rating vs. Reviews
# - PATTERN(S): Both the plots show a positive linear relationship between Ratings and Reviews.
# - We can confirm that if an app has more reviews, they tend to receive better ratings.
sns.scatterplot(x='Rating', y='Reviews', data=df)
plt.show()
```

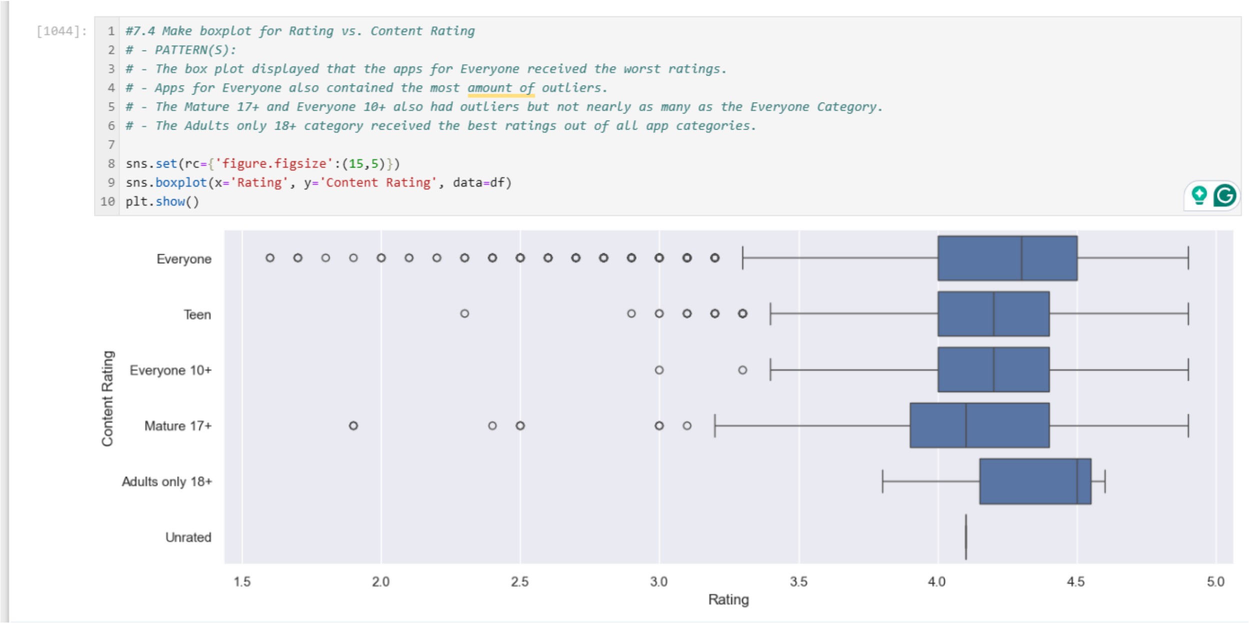




```
[1030]:
sns.jointplot(x='Rating', y='Reviews', data=df)
plt.show()
```

4. Make boxplot for Rating vs. Content Rating

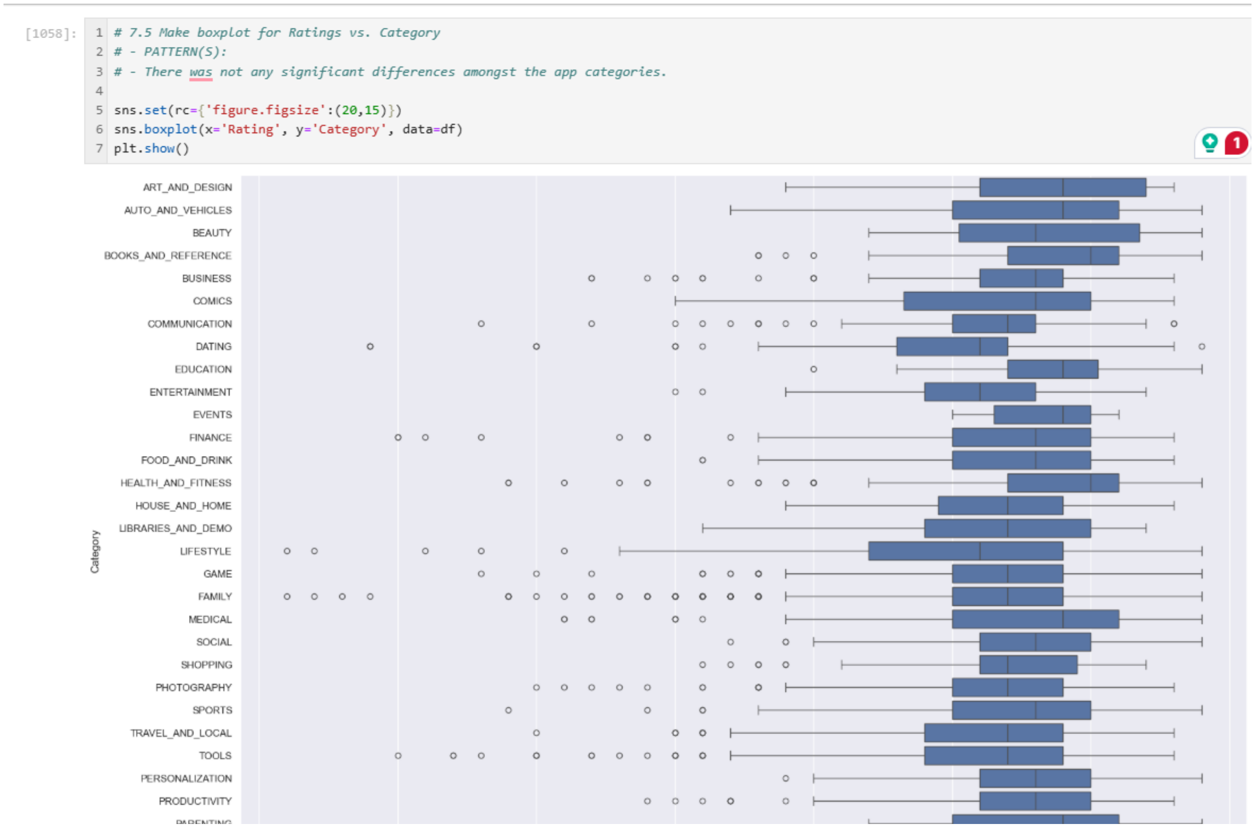
   1. Is there any difference in the ratings? Are some types liked better?



```
[1044]:
#7.4 Make boxplot for Rating vs. Content Rating
# - PATTERN(S):
# - The box plot displayed that the apps for Everyone received the worst ratings.
# - Apps for Everyone also contained the most amount of outliers.
# - The Mature 17+ and Everyone 10+ also had outliers but not nearly as many as the Everyone Category.
# - The Adults only 18+ category received the best ratings out of all app categories.

sns.set(rc={'figure.figsize':(15,5)})
sns.boxplot(x='Rating', y='Content Rating', data=df)
plt.show()
```

5. Make boxplot for Ratings vs. Category

    1. Which genre has the best ratings?



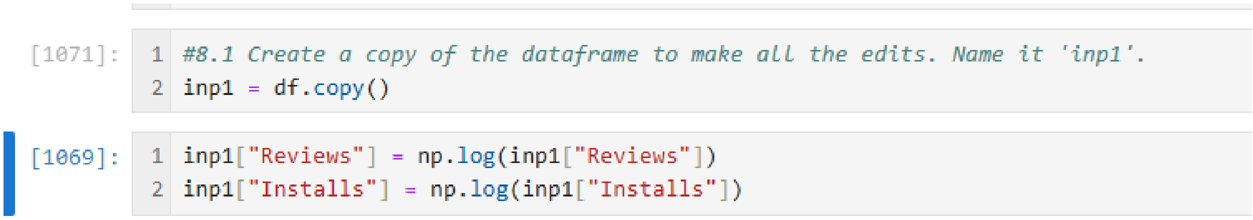
[1058]:

```
# 7.5 Make boxplot for Ratings vs. Category
# - PATTERN(S):
# - There was not any significant differences amongst the app categories.

sns.set(rc={'figure.figsize':(20,15)})
sns.boxplot(x='Rating', y='Category', data=df)
plt.show()
```

8. Data preprocessing

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

1. Reviews and Install have some values that are still relatively very high. Before building a linear regression model, you need to reduce the skew. Apply log transformation (np.log1p) to Reviews and Installs.



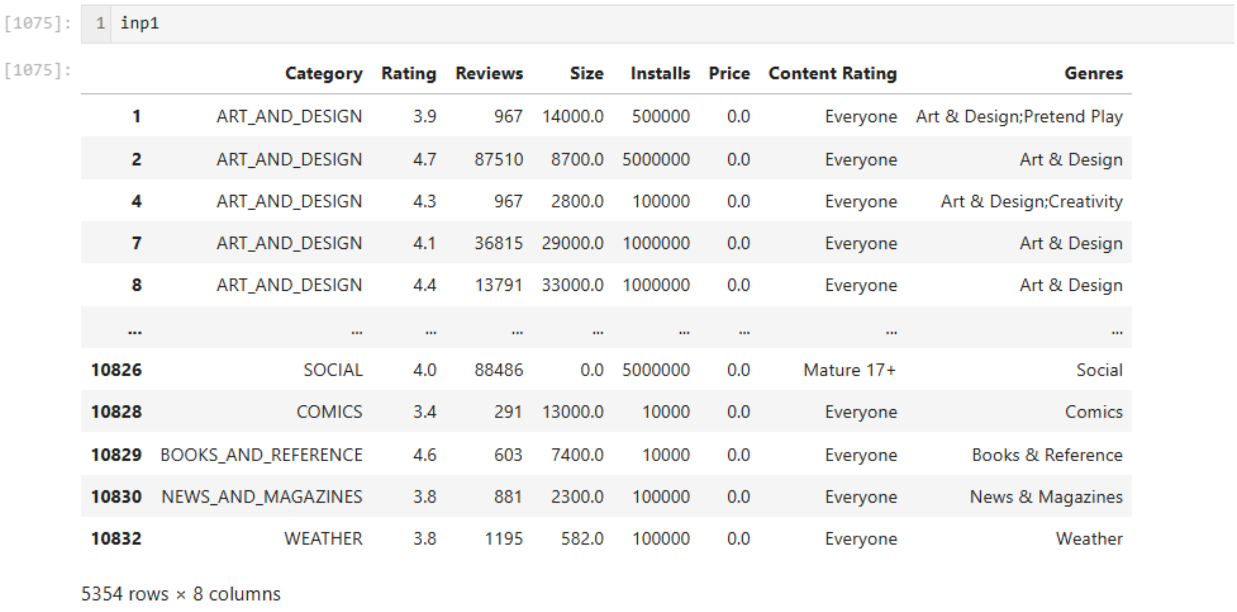
[1071]:

```
#8.1 Create a copy of the dataframe to make all the edits. Name it 'inp1'.
inp1 = df.copy()
```

[1069]:

```
inp1["Reviews"] = np.log(inp1["Reviews"])
inp1["Installs"] = np.log(inp1["Installs"])
```

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



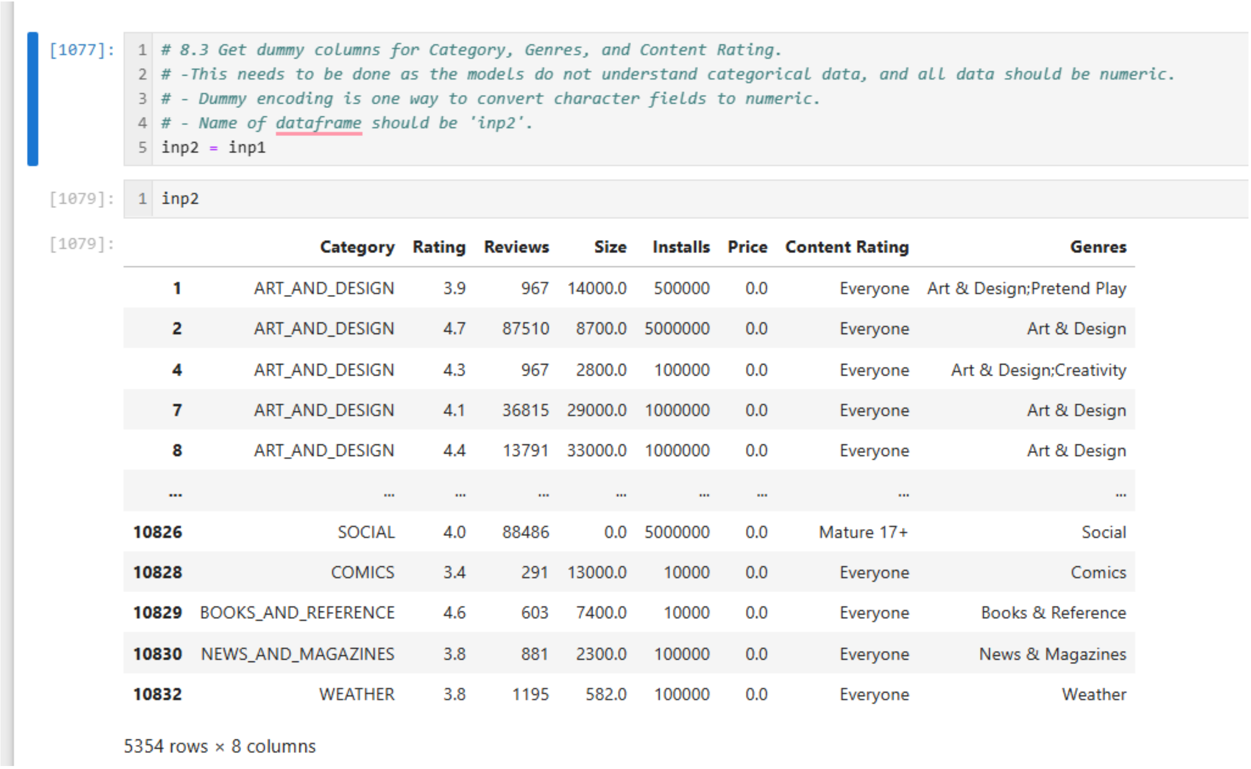
```
[1075]: inp1
```

[1075]:

	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres
1	ART_AND_DESIGN	3.9	967	14000.0	500000	0.0	Everyone	Art & Design;Pretend Play
2	ART_AND_DESIGN	4.7	87510	8700.0	5000000	0.0	Everyone	Art & Design
4	ART_AND_DESIGN	4.3	967	2800.0	100000	0.0	Everyone	Art & Design;Creativity
7	ART_AND_DESIGN	4.1	36815	29000.0	1000000	0.0	Everyone	Art & Design
8	ART_AND_DESIGN	4.4	13791	33000.0	1000000	0.0	Everyone	Art & Design
...	...	...	...	...	...	...	...	...
10826	SOCIAL	4.0	88486	0.0	5000000	0.0	Mature 17+	Social
10828	COMICS	3.4	291	13000.0	10000	0.0	Everyone	Comics
10829	BOOKS_AND_REFERENCE	4.6	603	7400.0	10000	0.0	Everyone	Books & Reference
10830	NEWS_AND_MAGAZINES	3.8	881	2300.0	100000	0.0	Everyone	News & Magazines
10832	WEATHER	3.8	1195	582.0	100000	0.0	Everyone	Weather

5354 rows × 8 columns

3. Get dummy columns for Category, Genres, and Content Rating. This needs to be done as the models do not understand categorical data, and all data should be numeric. Dummy encoding is one way to convert character fields to numeric. Name of dataframe should be **inp2**.



```
[1077]: # 8.3 Get dummy columns for Category, Genres, and Content Rating.
        # -This needs to be done as the models do not understand categorical data, and all data should be numeric.
        # - Dummy encoding is one way to convert character fields to numeric.
        # - Name of dataframe should be 'inp2'.
        inp2 = inp1
```

```
[1079]: inp2
```

[1079]:

	Category	Rating	Reviews	Size	Installs	Price	Content Rating	Genres
1	ART_AND_DESIGN	3.9	967	14000.0	500000	0.0	Everyone	Art & Design;Pretend Play
2	ART_AND_DESIGN	4.7	87510	8700.0	5000000	0.0	Everyone	Art & Design
4	ART_AND_DESIGN	4.3	967	2800.0	100000	0.0	Everyone	Art & Design;Creativity
7	ART_AND_DESIGN	4.1	36815	29000.0	1000000	0.0	Everyone	Art & Design
8	ART_AND_DESIGN	4.4	13791	33000.0	1000000	0.0	Everyone	Art & Design
...	...	...	...	...	...	...	...	...
10826	SOCIAL	4.0	88486	0.0	5000000	0.0	Mature 17+	Social
10828	COMICS	3.4	291	13000.0	10000	0.0	Everyone	Comics
10829	BOOKS_AND_REFERENCE	4.6	603	7400.0	10000	0.0	Everyone	Books & Reference
10830	NEWS_AND_MAGAZINES	3.8	881	2300.0	100000	0.0	Everyone	News & Magazines
10832	WEATHER	3.8	1195	582.0	100000	0.0	Everyone	Weather

5354 rows × 8 columns

[1083]:

```
# Get unique values for Category column
inp2['Category'].unique()
```

[1083]:

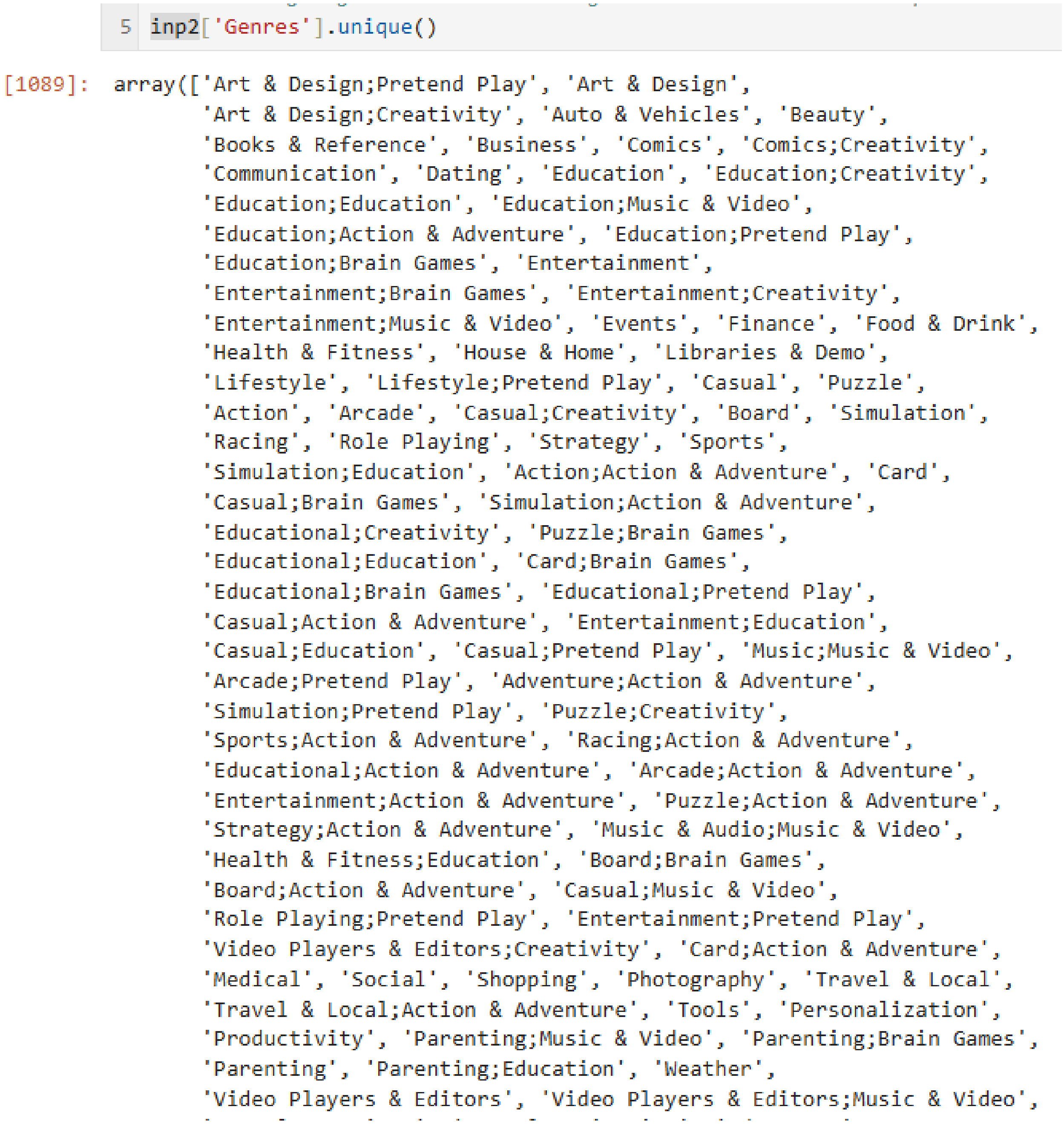
```
array(['ART_AND_DESIGN', 'AUTO_AND_VEHICLES', 'BEAUTY',
       'BOOKS_AND_REFERENCE', 'BUSINESS', 'COMICS', 'COMMUNICATION',
       'DATING', 'EDUCATION', 'ENTERTAINMENT', 'EVENTS', 'FINANCE',
       'FOOD_AND_DRINK', 'HEALTH_AND_FITNESS', 'HOUSE_AND_HOME',
       'LIBRARIES_AND_DEMO', 'LIFESTYLE', '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)
```

[1085]:

```
# Storing the Category column into 'x' varible that will and delete Category column from inp2.
# + Concatinate the encoded columns to inp2 df.
def create_dummies(df, col):
    df[col] = pd.Categorical(df[col])

    x = df[[col]]
    del df[col]

    dummies = pd.get_dummies(x,prefix=col)
    df = pd.concat([df,dummies], axis = 1)
    return df

inp2 = create_dummies(inp2, 'Category')
```

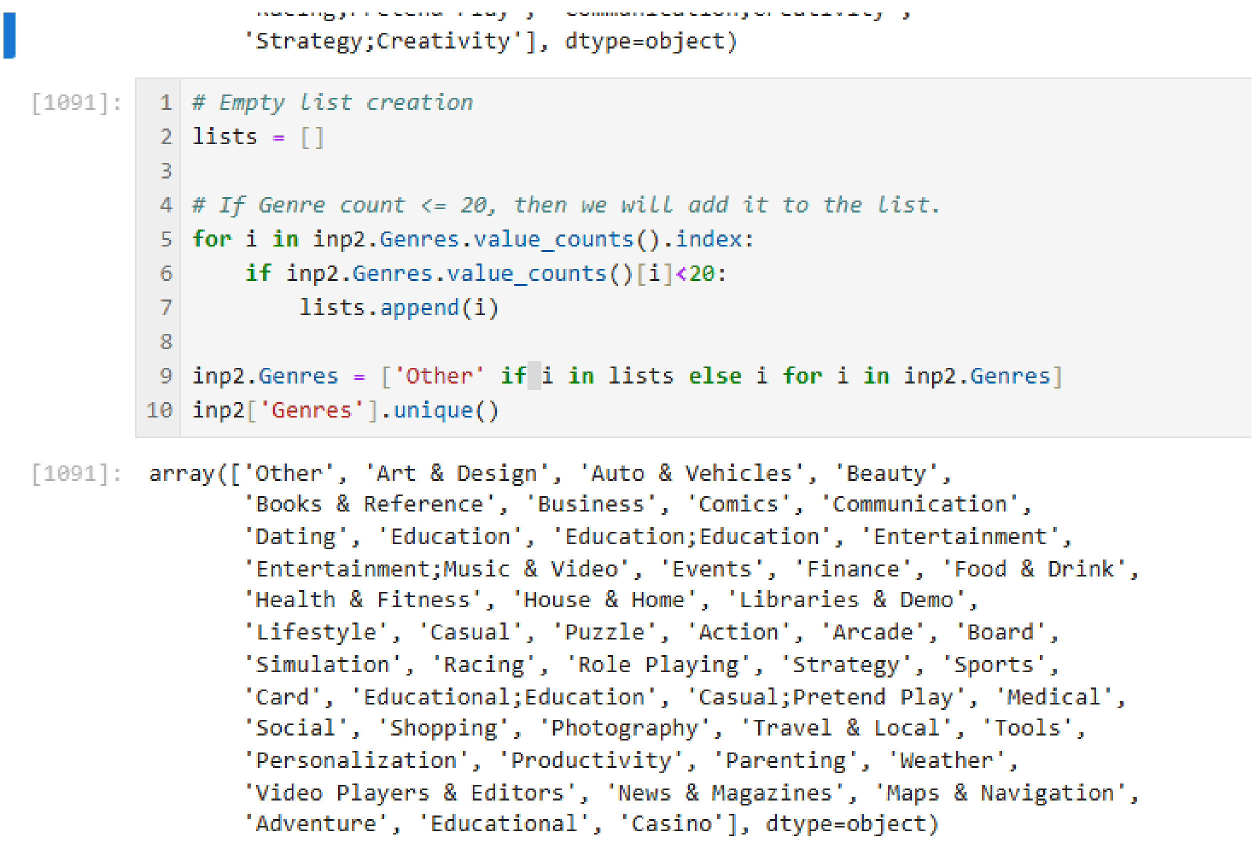
[1089]:

```
# Get unique values in Gthe enres column
# ANALYSIS:
# - We have too many categories under the Genre column.
# - We are going to eliminate some genres that have small smaple sizes and put them under a misculaneous column (aka. 'Other')
inp2['Genres'].unique()
```



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

```
[1089]: array(['Art & Design;Pretend Play', 'Art & Design',
               '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', 'Casual', 'Puzzle',
               'Action', 'Arcade', 'Casual;Creativity', 'Board', 'Simulation',
               'Racing', 'Role Playing', 'Strategy', 'Sports',
               'Simulation;Education', 'Action;Action & Adventure', 'Card',
               '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',
               'Arcade;Pretend Play', 'Adventure;Action & Adventure',
               'Simulation;Pretend Play', 'Puzzle;Creativity',
               'Sports;Action & Adventure', 'Racing;Action & Adventure',
               'Educational;Action & Adventure', 'Arcade;Action & Adventure',
               'Entertainment;Action & Adventure', 'Puzzle;Action & Adventure',
               'Strategy;Action & Adventure', 'Music & Audio;Music & Video',
               'Health & Fitness;Education', 'Board;Brain Games',
               'Board;Action & Adventure', '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', 'Personalization',
               'Productivity', 'Parenting;Music & Video', 'Parenting;Brain Games',
               'Parenting', 'Parenting;Education', 'Weather',
               'Video Players & Editors', 'Video Players & Editors;Music & Video',
```



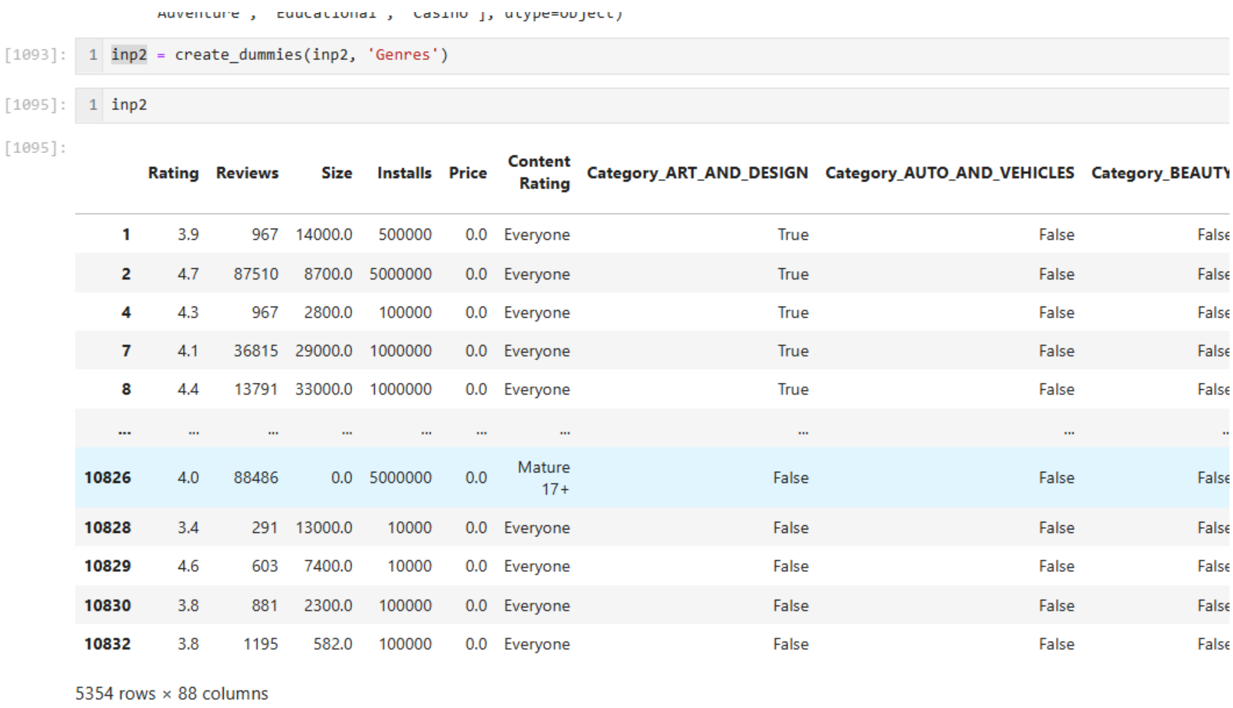
```
       'Racing;Pretend Play', 'Communication;Creativity',
       'Strategy;Creativity'], dtype=object)
```

[1091]:

```
# Empty list creation
lists = []

# If Genre count <= 20, then we will add it to the list.
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]
inp2['Genres'].unique()
```

[1091]:

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



```
       'Adventure', 'Educational', 'Casino'], dtype=object)
```

[1093]:

```
inp2 = create_dummies(inp2, 'Genres')
```

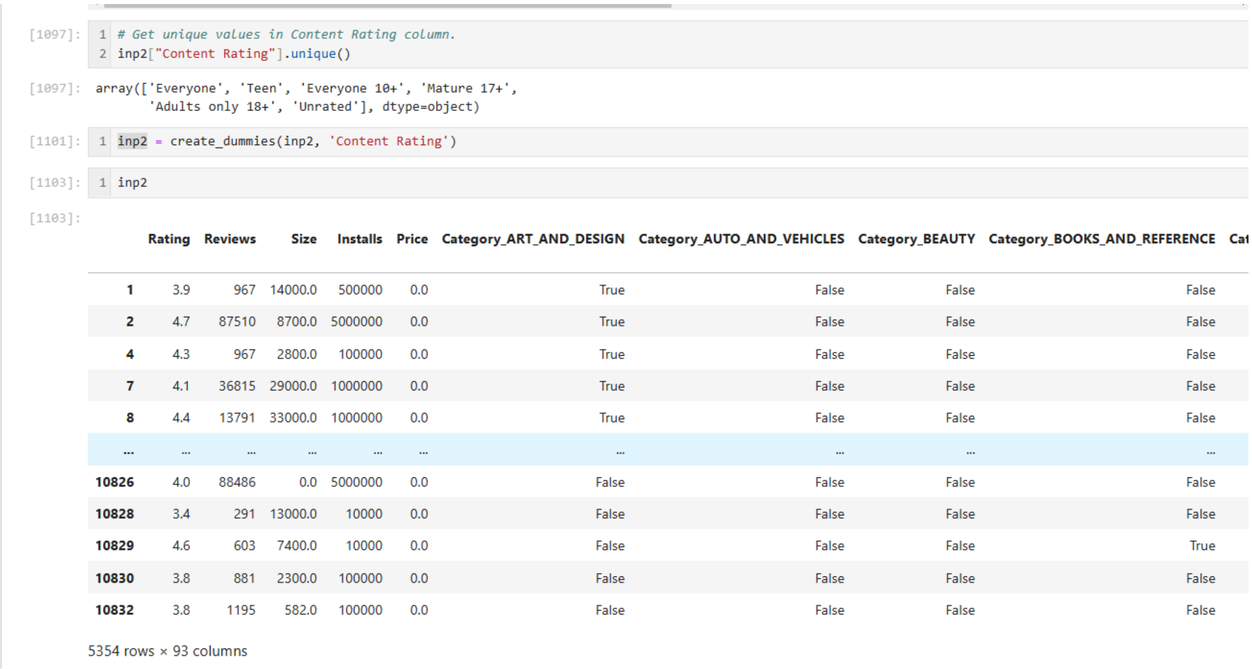
[1095]:

```
inp2
```

[1095]:

	Rating	Reviews	Size	Installs	Price	Content Rating	Category_ART_AND_DESIGN	Category_AUTO_AND_VEHICLES	Category_BEAUTY
1	3.9	967	14000.0	500000	0.0	Everyone	True	False	False
2	4.7	87510	8700.0	5000000	0.0	Everyone	True	False	False
4	4.3	967	2800.0	100000	0.0	Everyone	True	False	False
7	4.1	36815	29000.0	1000000	0.0	Everyone	True	False	False
8	4.4	13791	33000.0	1000000	0.0	Everyone	True	False	False
...	...	...	...	...	...	...	...	...	...
10826	4.0	88486	0.0	5000000	0.0	Mature 17+	False	False	False
10828	3.4	291	13000.0	10000	0.0	Everyone	False	False	False
10829	4.6	603	7400.0	10000	0.0	Everyone	False	False	False
10830	3.8	881	2300.0	100000	0.0	Everyone	False	False	False
10832	3.8	1195	582.0	100000	0.0	Everyone	False	False	False

5354 rows × 88 columns

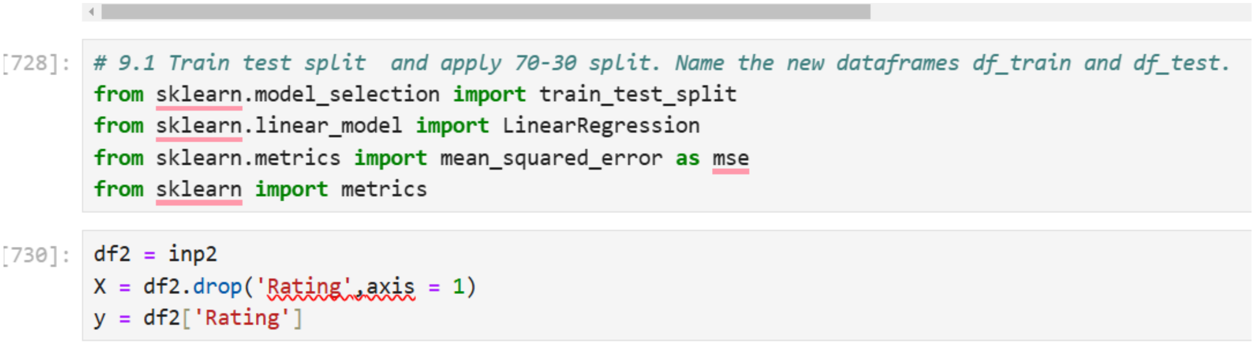


```
[1097]: # Get unique values in Content Rating column.
        inp2["Content Rating"].unique()

[1097]: array(['Everyone', 'Teen', 'Everyone 10+', 'Mature 17+',
               'Adults only 18+', 'Unrated'], dtype=object)

[1101]: inp2 = create_dummies(inp2, 'Content Rating')

[1103]: inp2

[1103]:
```

	Rating	Reviews	Size	Installs	Price	Category_ART_AND_DESIGN	Category_AUTO_AND_VEHICLES	Category_BEAUTY	Category_BOOKS_AND_REFERENCE	Cat
1	3.9	967	14000.0	500000	0.0	True	False	False	False	
2	4.7	87510	8700.0	5000000	0.0	True	False	False	False	
4	4.3	967	2800.0	100000	0.0	True	False	False	False	
7	4.1	36815	29000.0	1000000	0.0	True	False	False	False	
8	4.4	13791	33000.0	1000000	0.0	True	False	False	False	
...	...	...	...	...	...	...	...	...	...	
10826	4.0	88486	0.0	5000000	0.0	False	False	False	False	
10828	3.4	291	13000.0	10000	0.0	False	False	False	False	
10829	4.6	603	7400.0	10000	0.0	False	False	False	True	
10830	3.8	881	2300.0	100000	0.0	False	False	False	False	
10832	3.8	1195	582.0	100000	0.0	False	False	False	False	

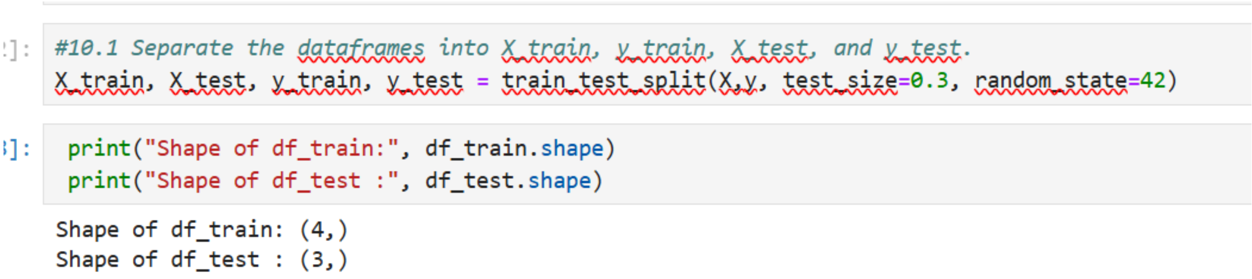
5354 rows × 93 columns

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



```
[728]: # 9.1 Train test split  and apply 70-30 split. Name the new dataframes df_train and df_test.
       from sklearn.model_selection import train_test_split
       from sklearn.linear_model import LinearRegression
       from sklearn.metrics import mean_squared_error as mse
       from sklearn import metrics

[730]: df2 = inp2
       X = df2.drop('Rating',axis = 1)
       y = df2['Rating']
```

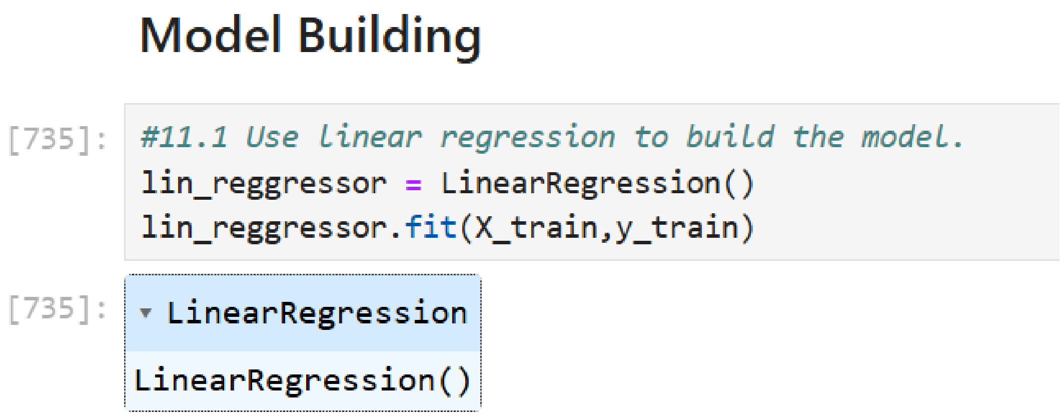
10. Separate the dataframes into X_train, y_train, X_test, and y_test.



```
]: #10.1 Separate the dataframes into X_train, y_train, X_test, and y_test.
   X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=42)

]:  print("Shape of df_train:", df_train.shape)
    print("Shape of df_test :", df_test.shape)

   Shape of df_train: (4,)
   Shape of df_test : (3,)
```

11 . Model building

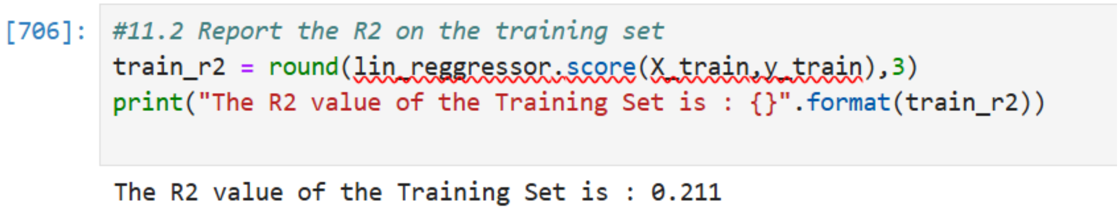
- Use linear regression as the technique



Model Building

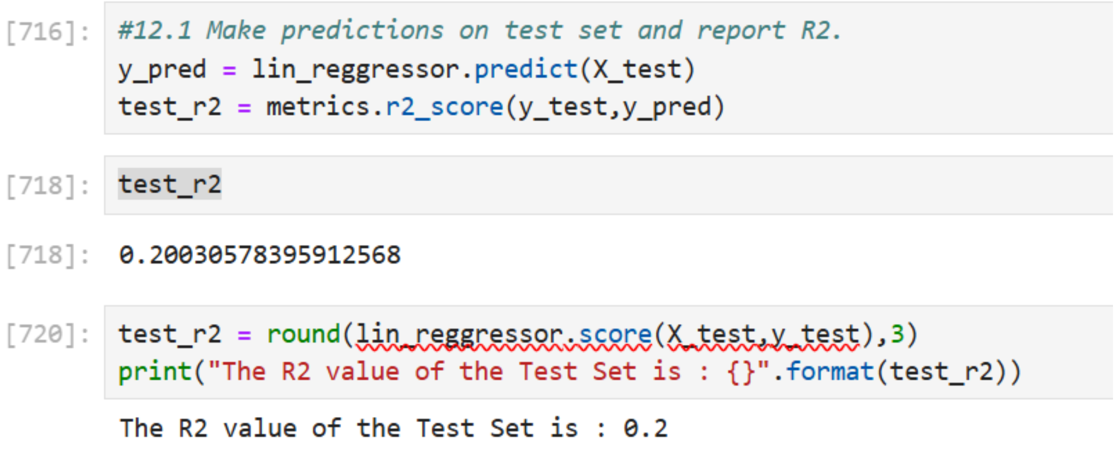
```
[735]: #11.1 Use linear regression to build the model.
       lin_reggressor = LinearRegression()
       lin_reggressor.fit(X_train,y_train)

[735]: ▾ LinearRegression
       LinearRegression()
```

- Report the R2 on the train set



```
[706]: #11.2 Report the R2 on the training set
       train_r2 = round(lin_reggressor.score(X_train,y_train),3)
       print("The R2 value of the Training Set is : {}".format(train_r2))

       The R2 value of the Training Set is : 0.211
```

12. Make predictions on test set and report R2.



```
[716]: #12.1 Make predictions on test set and report R2.
       y_pred = lin_reggressor.predict(X_test)
       test_r2 = metrics.r2_score(y_test,y_pred)

[718]: test_r2

[718]: 0.20030578395912568

[720]: test_r2 = round(lin_reggressor.score(X_test,y_test),3)
       print("The R2 value of the Test Set is : {}".format(test_r2))

       The R2 value of the Test Set is : 0.2
```