In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:
```
df=pd.read_csv(r"C:\Users\Mukul\Downloads\googleplaystore.csv")
```

In [3]:
```
df.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	A
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Des
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	A
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	A
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Desig

In [4]:
```
df.tail()
```

Out[4]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53M	5,000+	Free	0	E
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6M	100+	Free	0	E
10838	Parkinson Exercices FR	MEDICAL	NaN	3	9.5M	1,000+	Free	0	E
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	Varies with device	1,000+	Free	0	
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19M	10,000,000+	Free	0	E

In [5]:
```
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
```

In [6]:
```
df.sample()
```

Out[6]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	La Update
5088	Ag Weather Tools	WEATHER	NaN	0	4.6M	500+	Free	0	Everyone	Weather	Janua 9, 20

In [7]:
```
df.columns
```

Out[7]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
       'Android Ver'],
      dtype='object')
```

In [8]:
```
df.shape
```

Out[8]:
```
(10841, 13)
```

In [9]:
```
df.duplicated().sum()
```

Out[9]:
```
483
```

In [10]:
```
df[df.duplicated()]
```

Out[10]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
229	Quick PDF Scanner + OCR FREE	BUSINESS	4.2	80805	Varies with device	5,000,000+	Free	0	Everyone
236	Box	BUSINESS	4.2	159872	Varies with device	10,000,000+	Free	0	Everyone
239	Google My Business	BUSINESS	4.4	70991	Varies with device	5,000,000+	Free	0	Everyone
256	ZOOM Cloud Meetings	BUSINESS	4.4	31614	37M	10,000,000+	Free	0	Everyone
261	join.me - Simple Meetings	BUSINESS	4.0	6989	Varies with device	1,000,000+	Free	0	Everyone
...	...	...	...	...	...	...	...	...	...
8643	Wunderlist: To-Do List & Tasks	PRODUCTIVITY	4.6	404610	Varies with device	10,000,000+	Free	0	Everyone
8654	TickTick: To Do List with Reminder, Day Planner	PRODUCTIVITY	4.6	25370	Varies with device	1,000,000+	Free	0	Everyone
8658	ColorNote Notepad Notes	PRODUCTIVITY	4.6	2401017	Varies with device	100,000,000+	Free	0	Everyone
10049	Airway Ex - Intubate. Anesthetize. Train.	MEDICAL	4.3	123	86M	10,000+	Free	0	Everyone
10768	AAFP	MEDICAL	3.8	63	24M	10,000+	Free	0	Everyone

483 rows × 13 columns

In [11]:
```
df.describe()
```

Out[11]:

	Rating
count	9367.000000
mean	4.193338
std	0.537431
min	1.000000
25%	4.000000
50%	4.300000
75%	4.500000
max	19.000000

In [12]:
```
df.describe(include='all')
```

Out[12]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Ge
count	10841	10841	9367.000000	10841	10841	10841	10840	10841	10840	1
unique	9660	34	NaN	6002	462	22	3	93	6	
top	ROBLOX	FAMILY	NaN	0	Varies with device	1,000,000+	Free	0	Everyone	
freq	9	1972	NaN	596	1695	1579	10039	10040	8714	
mean	NaN	NaN	4.193338	NaN	NaN	NaN	NaN	NaN	NaN	
std	NaN	NaN	0.537431	NaN	NaN	NaN	NaN	NaN	NaN	
min	NaN	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	
25%	NaN	NaN	4.000000	NaN	NaN	NaN	NaN	NaN	NaN	
50%	NaN	NaN	4.300000	NaN	NaN	NaN	NaN	NaN	NaN	
75%	NaN	NaN	4.500000	NaN	NaN	NaN	NaN	NaN	NaN	
max	NaN	NaN	19.000000	NaN	NaN	NaN	NaN	NaN	NaN	

In [13]:
```
df['Reviews'].head()
```

Out[13]:
```
0       159
1       967
2     87510
3    215644
4       967
Name: Reviews, dtype: object
```

In [14]:
```
df['Reviews'].shape
```

Out[14]:
```
(10841,)
```

In [ ]:
```

```

In [15]:
```
df['Reviews'].dtypes
```

Out[15]:
```
dtype('O')
```

In [16]:
```
df.Reviews.str.isnumeric().sum()
```

Out[16]:
```
10840
```

In [17]:
```
~df['Reviews'].str.isnumeric()
```

Out[17]:
```
0        False
1        False
2        False
3        False
4        False
         ...
10836    False
10837    False
10838    False
10839    False
10840    False
Name: Reviews, Length: 10841, dtype: bool
```

In [18]:
```
df['Reviews'][10470:10480]  # indicate the 1 text data 10472
```

Out[18]:
```
10470        49
10471      1042
10472      3.0M
10473    134203
10474        37
10475       132
10476       552
10477       128
10478         4
10479       382
Name: Reviews, dtype: object
```

In [19]:
```
df[~df['Reviews'].str.isnumeric()]
```

Out[19]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0M	1,000+	Free	0	Everyone	NaN	February 11, 2018

In [ ]:
```
```

In [20]:
```
df_copy=df.copy()
df_copy
```

Out[20]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	E
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	E
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	E
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	E
...	...	...	...	...	...	...	...	...	
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53M	5,000+	Free	0	E
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6M	100+	Free	0	E
10838	Parkinson Exercices FR	MEDICAL	NaN	3	9.5M	1,000+	Free	0	E
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	Varies with device	1,000+	Free	0	
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19M	10,000,000+	Free	0	E

10841 rows × 13 columns

In [ ]:

```
```

In [21]:

```
df_copy.drop(df_copy.index[10472],inplace=True)
```

In [22]:

```
df_copy[10469:10475]
```

Out[22]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
10469	TownWiFi | Wi-Fi Everywhere	COMMUNICATION	3.9	2372	58M	500,000+	Free	0	Everyone
10470	Jazz Wi-Fi	COMMUNICATION	3.4	49	4.0M	10,000+	Free	0	Everyone
10471	Xposed Wi-Fi-Pwd	PERSONALIZATION	3.5	1042	404k	100,000+	Free	0	Everyone
10473	osmino Wi-Fi: free WiFi	TOOLS	4.2	134203	4.1M	10,000,000+	Free	0	Everyone
10474	Sat-Fi Voice	COMMUNICATION	3.4	37	14M	1,000+	Free	0	Everyone
10475	Wi-Fi Visualizer	TOOLS	3.9	132	2.6M	50,000+	Free	0	Everyone

In [23]:

```
df_copy.shape
```

Out[23]:

```
(10840, 13)
```

In [24]:

```
df['Reviews'].dtypes
```

Out[24]:

```
dtype('O')
```

In [25]:

```
df_copy['Reviews']=df_copy['Reviews'].astype('int')
```

In [26]:

```
df_copy['Reviews'].dtypes
```

Out[26]:

```
dtype('int32')
```

In [27]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10840 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10840 non-null  object 
 1   Category        10840 non-null  object 
 2   Rating          9366 non-null   float64
 3   Reviews         10840 non-null  int32  
 4   Size            10840 non-null  object 
 5   Installs        10840 non-null  object 
 6   Type            10839 non-null  object 
 7   Price           10840 non-null  object 
 8   Content Rating  10840 non-null  object 
 9   Genres          10840 non-null  object 
 10  Last Updated    10840 non-null  object 
 11  Current Ver     10832 non-null  object 
 12  Android Ver     10838 non-null  object 
dtypes: float64(1), int32(1), object(11)
memory usage: 1.1+ MB
```

In [28]:
```
df['Size'].unique()
```

Out[28]:
```
array(['19M', '14M', '8.7M', '25M', '2.8M', '5.6M', '29M', '33M', '3.1M',
       '28M', '12M', '20M', '21M', '37M', '2.7M', '5.5M', '17M', '39M',
       '31M', '4.2M', '7.0M', '23M', '6.0M', '6.1M', '4.6M', '9.2M',
       '5.2M', '11M', '24M', 'Varies with device', '9.4M', '15M', '10M',
       '1.2M', '26M', '8.0M', '7.9M', '56M', '57M', '35M', '54M', '201k',
       '3.6M', '5.7M', '8.6M', '2.4M', '27M', '2.5M', '16M', '3.4M',
       '8.9M', '3.9M', '2.9M', '38M', '32M', '5.4M', '18M', '1.1M',
       '2.2M', '4.5M', '9.8M', '52M', '9.0M', '6.7M', '30M', '2.6M',
       '7.1M', '3.7M', '22M', '7.4M', '6.4M', '3.2M', '8.2M', '9.9M',
       '4.9M', '9.5M', '5.0M', '5.9M', '13M', '73M', '6.8M', '3.5M',
       '4.0M', '2.3M', '7.2M', '2.1M', '42M', '7.3M', '9.1M', '55M',
       '23k', '6.5M', '1.5M', '7.5M', '51M', '41M', '48M', '8.5M', '46M',
       '8.3M', '4.3M', '4.7M', '3.3M', '40M', '7.8M', '8.8M', '6.6M',
       '5.1M', '61M', '66M', '79k', '8.4M', '118k', '44M', '695k', '1.6M',
       '6.2M', '18k', '53M', '1.4M', '3.0M', '5.8M', '3.8M', '9.6M',
       '45M', '63M', '49M', '77M', '4.4M', '4.8M', '70M', '6.9M', '9.3M',
       '10.0M', '8.1M', '36M', '84M', '97M', '2.0M', '1.9M', '1.8M',
       '5.3M', '47M', '556k', '526k', '76M', '7.6M', '59M', '9.7M', '78M',
       '72M', '43M', '7.7M', '6.3M', '334k', '34M', '93M', '65M', '79M',
       '100M', '58M', '50M', '68M', '64M', '67M', '60M', '94M', '232k',
       '99M', '624k', '95M', '8.5k', '41k', '292k', '11k', '80M', '1.7M',
       '74M', '62M', '69M', '75M', '98M', '85M', '82M', '96M', '87M',
       '71M', '86M', '91M', '81M', '92M', '83M', '88M', '704k', '862k',
       '899k', '378k', '266k', '375k', '1.3M', '975k', '980k', '4.1M',
       '89M', '696k', '544k', '525k', '920k', '779k', '853k', '720k',
       '713k', '772k', '318k', '58k', '241k', '196k', '857k', '51k',
       '953k', '865k', '251k', '930k', '540k', '313k', '746k', '203k',
       '26k', '314k', '239k', '371k', '220k', '730k', '756k', '91k',
       '293k', '17k', '74k', '14k', '317k', '78k', '924k', '902k', '818k',
       '81k', '939k', '169k', '45k', '475k', '965k', '90M', '545k', '61k',
       '283k', '655k', '714k', '93k', '872k', '121k', '322k', '1.0M',
       '976k', '172k', '238k', '549k', '206k', '954k', '444k', '717k',
       '210k', '609k', '308k', '705k', '306k', '904k', '473k', '175k',
       '350k', '383k', '454k', '421k', '70k', '812k', '442k', '842k',
       '417k', '412k', '459k', '478k', '335k', '782k', '721k', '430k',
       '429k', '192k', '200k', '460k', '728k', '496k', '816k', '414k',
       '506k', '887k', '613k', '243k', '569k', '778k', '683k', '592k',
       '319k', '186k', '840k', '647k', '191k', '373k', '437k', '598k',
       '716k', '585k', '982k', '222k', '219k', '55k', '948k', '323k',
       '691k', '511k', '951k', '963k', '25k', '554k', '351k', '27k',
       '82k', '208k', '913k', '514k', '551k', '29k', '103k', '898k',
       '743k', '116k', '153k', '209k', '353k', '499k', '173k', '597k',
       '809k', '122k', '411k', '400k', '801k', '787k', '237k', '50k',
       '643k', '986k', '97k', '516k', '837k', '780k', '961k', '269k',
       '20k', '498k', '600k', '749k', '642k', '881k', '72k', '656k',
       '601k', '221k', '228k', '108k', '940k', '176k', '33k', '663k',
       '34k', '942k', '259k', '164k', '458k', '245k', '629k', '28k',
       '288k', '775k', '785k', '636k', '916k', '994k', '309k', '485k',
       '914k', '903k', '608k', '500k', '54k', '562k', '847k', '957k',
       '688k', '811k', '270k', '48k', '329k', '523k', '921k', '874k',
       '981k', '784k', '280k', '24k', '518k', '754k', '892k', '154k',
       '860k', '364k', '387k', '626k', '161k', '879k', '39k', '970k',
       '170k', '141k', '160k', '144k', '143k', '190k', '376k', '193k',
```

```
'246k', '73k', '658k', '992k', '253k', '420k', '404k', '1,000+',
'470k', '226k', '240k', '89k', '234k', '257k', '861k', '467k',
'157k', '44k', '676k', '67k', '552k', '885k', '1020k', '582k',
'619k'], dtype=object)
```

In [ ]:

In [29]:
```
df_copy['Size']=df_copy['Size'].str.replace('M','000')
df_copy['Size']=df_copy['Size'].str.replace('k','')
df_copy['Size']=df_copy['Size'].replace("Varies with device",np.nan)
df_copy['Size']=df_copy['Size'].astype('float')
```

In [30]:
```
df_copy['Size'].unique()
```

Out[30]:
```
array([1.90e+04, 1.40e+04, 8.70e+00, 2.50e+04, 2.80e+00, 5.60e+00,
       2.90e+04, 3.30e+04, 3.10e+00, 2.80e+04, 1.20e+04, 2.00e+04,
       2.10e+04, 3.70e+04, 2.70e+00, 5.50e+00, 1.70e+04, 3.90e+04,
       3.10e+04, 4.20e+00, 7.00e+00, 2.30e+04, 6.00e+00, 6.10e+00,
       4.60e+00, 9.20e+00, 5.20e+00, 1.10e+04, 2.40e+04,      nan,
       9.40e+00, 1.50e+04, 1.00e+04, 1.20e+00, 2.60e+04, 8.00e+00,
       7.90e+00, 5.60e+04, 5.70e+04, 3.50e+04, 5.40e+04, 2.01e+02,
       3.60e+00, 5.70e+00, 8.60e+00, 2.40e+00, 2.70e+04, 2.50e+00,
       1.60e+04, 3.40e+00, 8.90e+00, 3.90e+00, 2.90e+00, 3.80e+04,
       3.20e+04, 5.40e+00, 1.80e+04, 1.10e+00, 2.20e+00, 4.50e+00,
       9.80e+00, 5.20e+04, 9.00e+00, 6.70e+00, 3.00e+04, 2.60e+00,
       7.10e+00, 3.70e+00, 2.20e+04, 7.40e+00, 6.40e+00, 3.20e+00,
       8.20e+00, 9.90e+00, 4.90e+00, 9.50e+00, 5.00e+00, 5.90e+00,
       1.30e+04, 7.30e+04, 6.80e+00, 3.50e+00, 4.00e+00, 2.30e+00,
       7.20e+00, 2.10e+00, 4.20e+04, 7.30e+00, 9.10e+00, 5.50e+04,
       2.30e+01, 6.50e+00, 1.50e+00, 7.50e+00, 5.10e+04, 4.10e+04,
       4.80e+04, 8.50e+00, 4.60e+04, 8.30e+00, 4.30e+00, 4.70e+00,
       3.30e+00, 4.00e+04, 7.80e+00, 8.80e+00, 6.60e+00, 5.10e+00,
       6.10e+04, 6.60e+04, 7.90e+01, 8.40e+00, 1.18e+02, 4.40e+04,
       6.95e+02, 1.60e+00, 6.20e+00, 1.80e+01, 5.30e+04, 1.40e+00,
       3.00e+00, 5.80e+00, 3.80e+00, 9.60e+00, 4.50e+04, 6.30e+04,
       4.90e+04, 7.70e+04, 4.40e+00, 4.80e+00, 7.00e+04, 6.90e+00,
       9.30e+00, 1.00e+01, 8.10e+00, 3.60e+04, 8.40e+04, 9.70e+04,
       2.00e+00, 1.90e+00, 1.80e+00, 5.30e+00, 4.70e+04, 5.56e+02,
       5.26e+02, 7.60e+04, 7.60e+00, 5.90e+04, 9.70e+00, 7.80e+04,
       7.20e+04, 4.30e+04, 7.70e+00, 6.30e+00, 3.34e+02, 3.40e+04,
       9.30e+04, 6.50e+04, 7.90e+04, 1.00e+05, 5.80e+04, 5.00e+04,
       6.80e+04, 6.40e+04, 6.70e+04, 6.00e+04, 9.40e+04, 2.32e+02,
       9.90e+04, 6.24e+02, 9.50e+04, 4.10e+01, 2.92e+02, 1.10e+01,
       8.00e+04, 1.70e+00, 7.40e+04, 6.20e+04, 6.90e+04, 7.50e+04,
       9.80e+04, 8.50e+04, 8.20e+04, 9.60e+04, 8.70e+04, 7.10e+04,
       8.60e+04, 9.10e+04, 8.10e+04, 9.20e+04, 8.30e+04, 8.80e+04,
       7.04e+02, 8.62e+02, 8.99e+02, 3.78e+02, 2.66e+02, 3.75e+02,
       1.30e+00, 9.75e+02, 9.80e+02, 4.10e+00, 8.90e+04, 6.96e+02,
       5.44e+02, 5.25e+02, 9.20e+02, 7.79e+02, 8.53e+02, 7.20e+02,
       7.13e+02, 7.72e+02, 3.18e+02, 5.80e+01, 2.41e+02, 1.96e+02,
       8.57e+02, 5.10e+01, 9.53e+02, 8.65e+02, 2.51e+02, 9.30e+02,
       5.40e+02, 3.13e+02, 7.46e+02, 2.03e+02, 2.60e+01, 3.14e+02,
       2.39e+02, 3.71e+02, 2.20e+02, 7.30e+02, 7.56e+02, 9.10e+01,
       2.93e+02, 1.70e+01, 7.40e+01, 1.40e+01, 3.17e+02, 7.80e+01,
       9.24e+02, 9.02e+02, 8.18e+02, 8.10e+01, 9.39e+02, 1.69e+02,
       4.50e+01, 4.75e+02, 9.65e+02, 9.00e+04, 5.45e+02, 6.10e+01,
       2.83e+02, 6.55e+02, 7.14e+02, 9.30e+01, 8.72e+02, 1.21e+02,
       3.22e+02, 1.00e+00, 9.76e+02, 1.72e+02, 2.38e+02, 5.49e+02,
       2.06e+02, 9.54e+02, 4.44e+02, 7.17e+02, 2.10e+02, 6.09e+02,
       3.08e+02, 7.05e+02, 3.06e+02, 9.04e+02, 4.73e+02, 1.75e+02,
       3.50e+02, 3.83e+02, 4.54e+02, 4.21e+02, 7.00e+01, 8.12e+02,
       4.42e+02, 8.42e+02, 4.17e+02, 4.12e+02, 4.59e+02, 4.78e+02,
       3.35e+02, 7.82e+02, 7.21e+02, 4.30e+02, 4.29e+02, 1.92e+02,
       2.00e+02, 4.60e+02, 7.28e+02, 4.96e+02, 8.16e+02, 4.14e+02,
       5.06e+02, 8.87e+02, 6.13e+02, 2.43e+02, 5.69e+02, 7.78e+02,
       6.83e+02, 5.92e+02, 3.19e+02, 1.86e+02, 8.40e+02, 6.47e+02,
```

```
       1.91e+02, 3.73e+02, 4.37e+02, 5.98e+02, 7.16e+02, 5.85e+02,
       9.82e+02, 2.22e+02, 2.19e+02, 5.50e+01, 9.48e+02, 3.23e+02,
       6.91e+02, 5.11e+02, 9.51e+02, 9.63e+02, 2.50e+01, 5.54e+02,
       3.51e+02, 2.70e+01, 8.20e+01, 2.08e+02, 9.13e+02, 5.14e+02,
       5.51e+02, 2.90e+01, 1.03e+02, 8.98e+02, 7.43e+02, 1.16e+02,
       1.53e+02, 2.09e+02, 3.53e+02, 4.99e+02, 1.73e+02, 5.97e+02,
       8.09e+02, 1.22e+02, 4.11e+02, 4.00e+02, 8.01e+02, 7.87e+02,
       2.37e+02, 5.00e+01, 6.43e+02, 9.86e+02, 9.70e+01, 5.16e+02,
       8.37e+02, 7.80e+02, 9.61e+02, 2.69e+02, 2.00e+01, 4.98e+02,
       6.00e+02, 7.49e+02, 6.42e+02, 8.81e+02, 7.20e+01, 6.56e+02,
       6.01e+02, 2.21e+02, 2.28e+02, 1.08e+02, 9.40e+02, 1.76e+02,
       3.30e+01, 6.63e+02, 3.40e+01, 9.42e+02, 2.59e+02, 1.64e+02,
       4.58e+02, 2.45e+02, 6.29e+02, 2.80e+01, 2.88e+02, 7.75e+02,
       7.85e+02, 6.36e+02, 9.16e+02, 9.94e+02, 3.09e+02, 4.85e+02,
       9.14e+02, 9.03e+02, 6.08e+02, 5.00e+02, 5.40e+01, 5.62e+02,
       8.47e+02, 9.57e+02, 6.88e+02, 8.11e+02, 2.70e+02, 4.80e+01,
       3.29e+02, 5.23e+02, 9.21e+02, 8.74e+02, 9.81e+02, 7.84e+02,
       2.80e+02, 2.40e+01, 5.18e+02, 7.54e+02, 8.92e+02, 1.54e+02,
       8.60e+02, 3.64e+02, 3.87e+02, 6.26e+02, 1.61e+02, 8.79e+02,
       3.90e+01, 9.70e+02, 1.70e+02, 1.41e+02, 1.60e+02, 1.44e+02,
       1.43e+02, 1.90e+02, 3.76e+02, 1.93e+02, 2.46e+02, 7.30e+01,
       6.58e+02, 9.92e+02, 2.53e+02, 4.20e+02, 4.04e+02, 4.70e+02,
       2.26e+02, 2.40e+02, 8.90e+01, 2.34e+02, 2.57e+02, 8.61e+02,
       4.67e+02, 1.57e+02, 4.40e+01, 6.76e+02, 6.70e+01, 5.52e+02,
       8.85e+02, 1.02e+03, 5.82e+02, 6.19e+02])
```

In [31]:
```
df_copy['Size'].value_counts
```

Out[31]:
```
<bound method IndexOpsMixin.value_counts of 0        19000.0
1        14000.0
2            8.7
3        25000.0
4            2.8
           ...
10836    53000.0
10837        3.6
10838        9.5
10839        NaN
10840    19000.0
Name: Size, Length: 10840, dtype: float64>
```

In [32]:
```
#df_copy['Size']=df['Size'].astype('float32')
```

In [33]:
```
df_copy['Size'].dtype
```

Out[33]:
```
dtype('float64')
```

In [34]:
```
df_copy['Size'].head()
```

Out[34]:
```
0    19000.0
1    14000.0
2        8.7
3    25000.0
4        2.8
Name: Size, dtype: float64
```

In [35]:
```
df_copy['Size'][2]
```

Out[35]:
```
8.7
```

In [36]:
```
for i in df_copy['Size']:
    if 1<10:
        df_copy['Size']=df_copy['Size'].replace(i , i*1000)
```

In [37]:
```
df_copy['Size'].head()
```

Out[37]:
```
0    19000000.0
1    14000000.0
2        8700.0
3    25000000.0
4        2800.0
Name: Size, dtype: float64
```

In [38]:
```
df_copy['Size'].unique()
```

Out[38]:
```
array([1.90e+07, 1.40e+07, 8.70e+03, 2.50e+07, 2.80e+03, 5.60e+03,
       2.90e+07, 3.30e+07, 3.10e+03, 2.80e+07, 1.20e+07, 2.00e+07,
       2.10e+07, 3.70e+07, 2.70e+03, 5.50e+03, 1.70e+07, 3.90e+07,
       3.10e+07, 4.20e+03, 7.00e+03, 2.30e+07, 6.00e+03, 6.10e+03,
       4.60e+03, 9.20e+03, 5.20e+03, 1.10e+07, 2.40e+07,      nan,
       9.40e+03, 1.50e+07, 1.00e+07, 1.20e+03, 2.60e+07, 8.00e+03,
       7.90e+03, 5.60e+07, 5.70e+07, 3.50e+07, 5.40e+07, 2.01e+05,
       3.60e+03, 5.70e+03, 8.60e+03, 2.40e+03, 2.70e+07, 2.50e+03,
       1.60e+07, 3.40e+03, 8.90e+03, 3.90e+03, 2.90e+03, 3.80e+07,
       3.20e+07, 5.40e+03, 1.80e+07, 1.10e+03, 2.20e+03, 4.50e+03,
       9.80e+03, 5.20e+07, 9.00e+03, 6.70e+03, 3.00e+07, 2.60e+03,
       7.10e+03, 3.70e+03, 2.20e+07, 7.40e+03, 6.40e+03, 3.20e+03,
       8.20e+03, 9.90e+03, 4.90e+03, 9.50e+03, 5.00e+03, 5.90e+03,
       1.30e+07, 7.30e+07, 6.80e+03, 3.50e+03, 4.00e+03, 2.30e+03,
       7.20e+03, 2.10e+03, 4.20e+07, 7.30e+03, 9.10e+03, 5.50e+07,
       6.50e+03, 1.50e+03, 7.50e+03, 5.10e+07, 4.10e+07, 4.80e+07,
       8.50e+03, 4.60e+07, 8.30e+03, 4.30e+03, 4.70e+03, 3.30e+03,
       4.00e+07, 7.80e+03, 8.80e+03, 6.60e+03, 5.10e+03, 6.10e+07,
       6.60e+07, 7.90e+07, 8.40e+03, 1.18e+05, 4.40e+07, 6.95e+05,
       1.60e+03, 6.20e+03, 5.30e+07, 1.40e+03, 3.00e+03, 5.80e+03,
       3.80e+03, 9.60e+03, 4.50e+07, 6.30e+07, 4.90e+07, 7.70e+07,
       4.40e+03, 4.80e+03, 7.00e+07, 6.90e+03, 9.30e+03, 8.10e+03,
       3.60e+07, 8.40e+07, 9.70e+07, 2.00e+03, 1.90e+03, 1.80e+03,
       5.30e+03, 4.70e+07, 5.56e+05, 5.26e+05, 7.60e+07, 7.60e+03,
       5.90e+07, 9.70e+03, 7.80e+07, 7.20e+07, 4.30e+07, 7.70e+03,
       6.30e+03, 3.34e+05, 3.40e+07, 9.30e+07, 6.50e+07, 1.00e+08,
       5.80e+07, 5.00e+07, 6.80e+07, 6.40e+07, 6.70e+07, 6.00e+07,
       9.40e+07, 2.32e+05, 9.90e+07, 6.24e+05, 9.50e+07, 2.92e+05,
       8.00e+07, 1.70e+03, 7.40e+07, 6.20e+07, 6.90e+07, 7.50e+07,
       9.80e+07, 8.50e+07, 8.20e+07, 9.60e+07, 8.70e+07, 7.10e+07,
       8.60e+07, 9.10e+07, 8.10e+07, 9.20e+07, 8.30e+07, 8.80e+07,
       7.04e+05, 8.62e+05, 8.99e+05, 3.78e+05, 2.66e+05, 3.75e+05,
       1.30e+03, 9.75e+05, 9.80e+05, 4.10e+03, 8.90e+07, 6.96e+05,
       5.44e+05, 5.25e+05, 9.20e+05, 7.79e+05, 8.53e+05, 7.20e+05,
       7.13e+05, 7.72e+05, 3.18e+05, 2.41e+05, 1.96e+05, 8.57e+05,
       9.53e+05, 8.65e+05, 2.51e+05, 9.30e+05, 5.40e+05, 3.13e+05,
       7.46e+05, 2.03e+05, 3.14e+05, 2.39e+05, 3.71e+05, 2.20e+05,
       7.30e+05, 7.56e+05, 2.93e+05, 3.17e+05, 9.24e+05, 9.02e+05,
       8.18e+05, 9.39e+05, 1.69e+05, 4.75e+05, 9.65e+05, 9.00e+07,
       5.45e+05, 2.83e+05, 6.55e+05, 7.14e+05, 8.72e+05, 1.21e+05,
       3.22e+05, 1.00e+03, 9.76e+05, 1.72e+05, 2.38e+05, 5.49e+05,
       2.06e+05, 9.54e+05, 4.44e+05, 7.17e+05, 2.10e+05, 6.09e+05,
       3.08e+05, 7.05e+05, 3.06e+05, 9.04e+05, 4.73e+05, 1.75e+05,
       3.50e+05, 3.83e+05, 4.54e+05, 4.21e+05, 8.12e+05, 4.42e+05,
       8.42e+05, 4.17e+05, 4.12e+05, 4.59e+05, 4.78e+05, 3.35e+05,
       7.82e+05, 7.21e+05, 4.30e+05, 4.29e+05, 1.92e+05, 2.00e+05,
       4.60e+05, 7.28e+05, 4.96e+05, 8.16e+05, 4.14e+05, 5.06e+05,
       8.87e+05, 6.13e+05, 2.43e+05, 5.69e+05, 7.78e+05, 6.83e+05,
       5.92e+05, 3.19e+05, 1.86e+05, 8.40e+05, 6.47e+05, 1.91e+05,
       3.73e+05, 4.37e+05, 5.98e+05, 7.16e+05, 5.85e+05, 9.82e+05,
       2.22e+05, 2.19e+05, 9.48e+05, 3.23e+05, 6.91e+05, 5.11e+05,
       9.51e+05, 9.63e+05, 5.54e+05, 3.51e+05, 2.08e+05, 9.13e+05,
       5.14e+05, 5.51e+05, 1.03e+05, 8.98e+05, 7.43e+05, 1.16e+05,
       1.53e+05, 2.09e+05, 3.53e+05, 4.99e+05, 1.73e+05, 5.97e+05,
```

```
8.09e+05, 1.22e+05, 4.11e+05, 4.00e+05, 8.01e+05, 7.87e+05,
2.37e+05, 6.43e+05, 9.86e+05, 5.16e+05, 8.37e+05, 7.80e+05,
9.61e+05, 2.69e+05, 4.98e+05, 6.00e+05, 7.49e+05, 6.42e+05,
8.81e+05, 6.56e+05, 6.01e+05, 2.21e+05, 2.28e+05, 1.08e+05,
9.40e+05, 1.76e+05, 6.63e+05, 9.42e+05, 2.59e+05, 1.64e+05,
4.58e+05, 2.45e+05, 6.29e+05, 2.88e+05, 7.75e+05, 7.85e+05,
6.36e+05, 9.16e+05, 9.94e+05, 3.09e+05, 4.85e+05, 9.14e+05,
9.03e+05, 6.08e+05, 5.00e+05, 5.62e+05, 8.47e+05, 9.57e+05,
6.88e+05, 8.11e+05, 2.70e+05, 3.29e+05, 5.23e+05, 9.21e+05,
8.74e+05, 9.81e+05, 7.84e+05, 2.80e+05, 5.18e+05, 7.54e+05,
8.92e+05, 1.54e+05, 8.60e+05, 3.64e+05, 3.87e+05, 6.26e+05,
1.61e+05, 8.79e+05, 9.70e+05, 1.70e+05, 1.41e+05, 1.60e+05,
1.44e+05, 1.43e+05, 1.90e+05, 3.76e+05, 1.93e+05, 2.46e+05,
7.30e+04, 6.58e+05, 9.92e+05, 2.53e+05, 4.20e+05, 4.04e+05,
4.70e+05, 2.26e+05, 2.40e+05, 8.90e+04, 2.34e+05, 2.57e+05,
8.61e+05, 4.67e+05, 1.57e+05, 6.76e+05, 6.70e+04, 5.52e+05,
8.85e+05, 1.02e+06, 5.82e+05, 6.19e+05])
```

In [39]:
```
# We can short the size values or scale our data
df_copy['Size']=df_copy['Size']/1000
```

In [40]:
```
df_copy['Size'].unique()
```

Out[40]:
```
array([1.90e+04, 1.40e+04, 8.70e+00, 2.50e+04, 2.80e+00, 5.60e+00,
       2.90e+04, 3.30e+04, 3.10e+00, 2.80e+04, 1.20e+04, 2.00e+04,
       2.10e+04, 3.70e+04, 2.70e+00, 5.50e+00, 1.70e+04, 3.90e+04,
       3.10e+04, 4.20e+00, 7.00e+00, 2.30e+04, 6.00e+00, 6.10e+00,
       4.60e+00, 9.20e+00, 5.20e+00, 1.10e+04, 2.40e+04,      nan,
       9.40e+00, 1.50e+04, 1.00e+04, 1.20e+00, 2.60e+04, 8.00e+00,
       7.90e+00, 5.60e+04, 5.70e+04, 3.50e+04, 5.40e+04, 2.01e+02,
       3.60e+00, 5.70e+00, 8.60e+00, 2.40e+00, 2.70e+04, 2.50e+00,
       1.60e+04, 3.40e+00, 8.90e+00, 3.90e+00, 2.90e+00, 3.80e+04,
       3.20e+04, 5.40e+00, 1.80e+04, 1.10e+00, 2.20e+00, 4.50e+00,
       9.80e+00, 5.20e+04, 9.00e+00, 6.70e+00, 3.00e+04, 2.60e+00,
       7.10e+00, 3.70e+00, 2.20e+04, 7.40e+00, 6.40e+00, 3.20e+00,
       8.20e+00, 9.90e+00, 4.90e+00, 9.50e+00, 5.00e+00, 5.90e+00,
       1.30e+04, 7.30e+04, 6.80e+00, 3.50e+00, 4.00e+00, 2.30e+00,
       7.20e+00, 2.10e+00, 4.20e+04, 7.30e+00, 9.10e+00, 5.50e+04,
       6.50e+00, 1.50e+00, 7.50e+00, 5.10e+04, 4.10e+04, 4.80e+04,
       8.50e+00, 4.60e+04, 8.30e+00, 4.30e+00, 4.70e+00, 3.30e+00,
       4.00e+04, 7.80e+00, 8.80e+00, 6.60e+00, 5.10e+00, 6.10e+04,
       6.60e+04, 7.90e+04, 8.40e+00, 1.18e+02, 4.40e+04, 6.95e+02,
       1.60e+00, 6.20e+00, 5.30e+04, 1.40e+00, 3.00e+00, 5.80e+00,
       3.80e+00, 9.60e+00, 4.50e+04, 6.30e+04, 4.90e+04, 7.70e+04,
       4.40e+00, 4.80e+00, 7.00e+04, 6.90e+00, 9.30e+00, 8.10e+00,
       3.60e+04, 8.40e+04, 9.70e+04, 2.00e+00, 1.90e+00, 1.80e+00,
       5.30e+00, 4.70e+04, 5.56e+02, 5.26e+02, 7.60e+04, 7.60e+00,
       5.90e+04, 9.70e+00, 7.80e+04, 7.20e+04, 4.30e+04, 7.70e+00,
       6.30e+00, 3.34e+02, 3.40e+04, 9.30e+04, 6.50e+04, 1.00e+05,
       5.80e+04, 5.00e+04, 6.80e+04, 6.40e+04, 6.70e+04, 6.00e+04,
       9.40e+04, 2.32e+02, 9.90e+04, 6.24e+02, 9.50e+04, 2.92e+02,
       8.00e+04, 1.70e+00, 7.40e+04, 6.20e+04, 6.90e+04, 7.50e+04,
       9.80e+04, 8.50e+04, 8.20e+04, 9.60e+04, 8.70e+04, 7.10e+04,
       8.60e+04, 9.10e+04, 8.10e+04, 9.20e+04, 8.30e+04, 8.80e+04,
       7.04e+02, 8.62e+02, 8.99e+02, 3.78e+02, 2.66e+02, 3.75e+02,
       1.30e+00, 9.75e+02, 9.80e+02, 4.10e+00, 8.90e+04, 6.96e+02,
       5.44e+02, 5.25e+02, 9.20e+02, 7.79e+02, 8.53e+02, 7.20e+02,
       7.13e+02, 7.72e+02, 3.18e+02, 2.41e+02, 1.96e+02, 8.57e+02,
       9.53e+02, 8.65e+02, 2.51e+02, 9.30e+02, 5.40e+02, 3.13e+02,
       7.46e+02, 2.03e+02, 3.14e+02, 2.39e+02, 3.71e+02, 2.20e+02,
       7.30e+02, 7.56e+02, 2.93e+02, 3.17e+02, 9.24e+02, 9.02e+02,
       8.18e+02, 9.39e+02, 1.69e+02, 4.75e+02, 9.65e+02, 9.00e+04,
       5.45e+02, 2.83e+02, 6.55e+02, 7.14e+02, 8.72e+02, 1.21e+02,
       3.22e+02, 1.00e+00, 9.76e+02, 1.72e+02, 2.38e+02, 5.49e+02,
       2.06e+02, 9.54e+02, 4.44e+02, 7.17e+02, 2.10e+02, 6.09e+02,
       3.08e+02, 7.05e+02, 3.06e+02, 9.04e+02, 4.73e+02, 1.75e+02,
       3.50e+02, 3.83e+02, 4.54e+02, 4.21e+02, 8.12e+02, 4.42e+02,
       8.42e+02, 4.17e+02, 4.12e+02, 4.59e+02, 4.78e+02, 3.35e+02,
       7.82e+02, 7.21e+02, 4.30e+02, 4.29e+02, 1.92e+02, 2.00e+02,
       4.60e+02, 7.28e+02, 4.96e+02, 8.16e+02, 4.14e+02, 5.06e+02,
       8.87e+02, 6.13e+02, 2.43e+02, 5.69e+02, 7.78e+02, 6.83e+02,
       5.92e+02, 3.19e+02, 1.86e+02, 8.40e+02, 6.47e+02, 1.91e+02,
       3.73e+02, 4.37e+02, 5.98e+02, 7.16e+02, 5.85e+02, 9.82e+02,
       2.22e+02, 2.19e+02, 9.48e+02, 3.23e+02, 6.91e+02, 5.11e+02,
       9.51e+02, 9.63e+02, 5.54e+02, 3.51e+02, 2.08e+02, 9.13e+02,
       5.14e+02, 5.51e+02, 1.03e+02, 8.98e+02, 7.43e+02, 1.16e+02,
       1.53e+02, 2.09e+02, 3.53e+02, 4.99e+02, 1.73e+02, 5.97e+02,
```

```
       8.09e+02, 1.22e+02, 4.11e+02, 4.00e+02, 8.01e+02, 7.87e+02,
       2.37e+02, 6.43e+02, 9.86e+02, 5.16e+02, 8.37e+02, 7.80e+02,
       9.61e+02, 2.69e+02, 4.98e+02, 6.00e+02, 7.49e+02, 6.42e+02,
       8.81e+02, 6.56e+02, 6.01e+02, 2.21e+02, 2.28e+02, 1.08e+02,
       9.40e+02, 1.76e+02, 6.63e+02, 9.42e+02, 2.59e+02, 1.64e+02,
       4.58e+02, 2.45e+02, 6.29e+02, 2.88e+02, 7.75e+02, 7.85e+02,
       6.36e+02, 9.16e+02, 9.94e+02, 3.09e+02, 4.85e+02, 9.14e+02,
       9.03e+02, 6.08e+02, 5.00e+02, 5.62e+02, 8.47e+02, 9.57e+02,
       6.88e+02, 8.11e+02, 2.70e+02, 3.29e+02, 5.23e+02, 9.21e+02,
       8.74e+02, 9.81e+02, 7.84e+02, 2.80e+02, 5.18e+02, 7.54e+02,
       8.92e+02, 1.54e+02, 8.60e+02, 3.64e+02, 3.87e+02, 6.26e+02,
       1.61e+02, 8.79e+02, 9.70e+02, 1.70e+02, 1.41e+02, 1.60e+02,
       1.44e+02, 1.43e+02, 1.90e+02, 3.76e+02, 1.93e+02, 2.46e+02,
       7.30e+01, 6.58e+02, 9.92e+02, 2.53e+02, 4.20e+02, 4.04e+02,
       4.70e+02, 2.26e+02, 2.40e+02, 8.90e+01, 2.34e+02, 2.57e+02,
       8.61e+02, 4.67e+02, 1.57e+02, 6.76e+02, 6.70e+01, 5.52e+02,
       8.85e+02, 1.02e+03, 5.82e+02, 6.19e+02])
```

In [41]:
```
df_copy.columns
```

Out[41]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
       'Android Ver'],
      dtype='object')
```

In [42]:
```
df_copy.head(2)
```

Out[42]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10,000+	Free	0	Everyone	Ar
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500,000+	Free	0	Everyone	Desig

In [43]:
```
df_copy['Installs'].astype
```

Out[43]:
```
<bound method NDFrame.astype of 0            10,000+
1           500,000+
2         5,000,000+
3        50,000,000+
4           100,000+
             ...
10836         5,000+
10837           100+
10838         1,000+
10839         1,000+
10840    10,000,000+
Name: Installs, Length: 10840, dtype: object>
```

In [44]:
```
df_copy['Installs'].unique()
```

Out[44]:
```
array(['10,000+', '500,000+', '5,000,000+', '50,000,000+', '100,000+',
       '50,000+', '1,000,000+', '10,000,000+', '5,000+', '100,000,000+',
       '1,000,000,000+', '1,000+', '500,000,000+', '50+', '100+', '500+',
       '10+', '1+', '5+', '0+', '0'], dtype=object)
```

In [45]:
```
df_copy['Price'].unique()
```

Out[45]:
```
array(['0', '$4.99', '$3.99', '$6.99', '$1.49', '$2.99', '$7.99', '$5.99',
       '$3.49', '$1.99', '$9.99', '$7.49', '$0.99', '$9.00', '$5.49',
       '$10.00', '$24.99', '$11.99', '$79.99', '$16.99', '$14.99',
       '$1.00', '$29.99', '$12.99', '$2.49', '$10.99', '$1.50', '$19.99',
       '$15.99', '$33.99', '$74.99', '$39.99', '$3.95', '$4.49', '$1.70',
       '$8.99', '$2.00', '$3.88', '$25.99', '$399.99', '$17.99',
       '$400.00', '$3.02', '$1.76', '$4.84', '$4.77', '$1.61', '$2.50',
       '$1.59', '$6.49', '$1.29', '$5.00', '$13.99', '$299.99', '$379.99',
       '$37.99', '$18.99', '$389.99', '$19.90', '$8.49', '$1.75',
       '$14.00', '$4.85', '$46.99', '$109.99', '$154.99', '$3.08',
       '$2.59', '$4.80', '$1.96', '$19.40', '$3.90', '$4.59', '$15.46',
       '$3.04', '$4.29', '$2.60', '$3.28', '$4.60', '$28.99', '$2.95',
       '$2.90', '$1.97', '$200.00', '$89.99', '$2.56', '$30.99', '$3.61',
       '$394.99', '$1.26', '$1.20', '$1.04'], dtype=object)
```

In [46]:
```
chars_to_remove=['+',',','$']
cols_to_clean=['Installs','Price']
for item in chars_to_remove:
    for col in cols_to_clean:
        df_copy[col]=df_copy[col].str.replace(item,'')
```

In [47]:
```
df_copy['Price'].unique()
```

Out[47]:
```
array(['0', '4.99', '3.99', '6.99', '1.49', '2.99', '7.99', '5.99',
       '3.49', '1.99', '9.99', '7.49', '0.99', '9.00', '5.49', '10.00',
       '24.99', '11.99', '79.99', '16.99', '14.99', '1.00', '29.99',
       '12.99', '2.49', '10.99', '1.50', '19.99', '15.99', '33.99',
       '74.99', '39.99', '3.95', '4.49', '1.70', '8.99', '2.00', '3.88',
       '25.99', '399.99', '17.99', '400.00', '3.02', '1.76', '4.84',
       '4.77', '1.61', '2.50', '1.59', '6.49', '1.29', '5.00', '13.99',
       '299.99', '379.99', '37.99', '18.99', '389.99', '19.90', '8.49',
       '1.75', '14.00', '4.85', '46.99', '109.99', '154.99', '3.08',
       '2.59', '4.80', '1.96', '19.40', '3.90', '4.59', '15.46', '3.04',
       '4.29', '2.60', '3.28', '4.60', '28.99', '2.95', '2.90', '1.97',
       '200.00', '89.99', '2.56', '30.99', '3.61', '394.99', '1.26',
       '1.20', '1.04'], dtype=object)
```

In [48]:
```
df_copy['Installs'].unique()
```

Out[48]:
```
array(['10000', '500000', '5000000', '50000000', '100000', '50000',
       '1000000', '10000000', '5000', '100000000', '1000000000', '1000',
       '500000000', '50', '100', '500', '10', '1', '5', '0'], dtype=object)
```

In [49]:
```
df_copy['Price']=df_copy['Price'].astype('float')
```

In [50]:
```
df_copy['Installs']=df_copy['Installs'].astype('int')
```

In [51]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10840 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10840 non-null  object 
 1   Category        10840 non-null  object 
 2   Rating          9366 non-null   float64
 3   Reviews         10840 non-null  int32  
 4   Size            9145 non-null   float64
 5   Installs        10840 non-null  int32  
 6   Type            10839 non-null  object 
 7   Price           10840 non-null  float64
 8   Content Rating  10840 non-null  object 
 9   Genres          10840 non-null  object 
 10  Last Updated    10840 non-null  object 
 11  Current Ver     10832 non-null  object 
 12  Android Ver     10838 non-null  object 
dtypes: float64(3), int32(2), object(8)
memory usage: 1.3+ MB
```

In [52]:
```
df_copy.head()
```

Out[52]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	De
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Desi

In [53]:
```
df_copy['Last Updated'].dtype
```

Out[53]:
```
dtype('O')
```

In [54]:
```
df_copy['Last Updated']
```

Out[54]:
```
0         January 7, 2018
1        January 15, 2018
2          August 1, 2018
3            June 8, 2018
4           June 20, 2018
               ...
10836       July 25, 2017
10837        July 6, 2018
10838    January 20, 2017
10839    January 19, 2015
10840       July 25, 2018
Name: Last Updated, Length: 10840, dtype: object
```

In [55]:
```
pd.to_datetime(df_copy['Last Updated'])
```

Out[55]:
```
0       2018-01-07
1       2018-01-15
2       2018-08-01
3       2018-06-08
4       2018-06-20
           ...
10836   2017-07-25
10837   2018-07-06
10838   2017-01-20
10839   2015-01-19
10840   2018-07-25
Name: Last Updated, Length: 10840, dtype: datetime64[ns]
```

In [56]:
```
df_copy['Last Updated']=pd.to_datetime(df_copy['Last Updated'])
```

In [57]:
```
df_copy.head()
```

Out[57]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	De
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Desi

In [ ]:
```
```

In [58]:
```
# segreted the day , month , year
df_copy['day']=df_copy['Last Updated'].dt.day
```

In [59]:
```
df_copy['month']=df_copy['Last Updated'].dt.month
```

In [60]:
```
df_copy['year']=df_copy['Last Updated'].dt.year
```

In [61]:
```
# sucessfully added the day, month , year column
df_copy.head()
```

Out[61]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	De
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Desi

In [62]:
```
# dropping the Last update columns
del df_copy['Last Updated']
```

Save the Cleane data to new CSV file

In [63]:
```
df_copy.to_csv("google_cleaned.csv",index=False)  # all my data is cleaned
```

In [64]:
```
df=pd.read_csv('google_cleaned.csv')
```

In [65]:
```
df.head() # indicate the top five data
```

Out[65]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	De
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Desi

In [66]:
```
df.tail() # indicate the last five data
```

Out[66]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	C
10835	Sya9a Maroc - FR	FAMILY	4.5	38	53000.0	5000	Free	0.0	Ev
10836	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6	100	Free	0.0	Ev
10837	Parkinson Exercices FR	MEDICAL	NaN	3	9.5	1000	Free	0.0	Ev
10838	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	NaN	1000	Free	0.0	
10839	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19000.0	10000000	Free	0.0	Ev

In [67]:
```
df.sample() # indicate the Sample
```

Out[67]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Cur
5616	Kingdoms at War: Hardcore PVP	FAMILY	4.3	34898	92000.0	1000000	Free	0.0	Teen	Strategy	

In [ ]:
```

```

In [68]:
```
df.shape  # indicate the shape
```

Out[68]:
```
(10840, 15)
```

In [69]:
```
# Check the information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10840 entries, 0 to 10839
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10840 non-null  object 
 1   Category        10840 non-null  object 
 2   Rating          9366 non-null   float64
 3   Reviews         10840 non-null  int64  
 4   Size            9145 non-null   float64
 5   Installs        10840 non-null  int64  
 6   Type            10839 non-null  object 
 7   Price           10840 non-null  float64
 8   Content Rating  10840 non-null  object 
 9   Genres          10840 non-null  object 
 10  Current Ver     10832 non-null  object 
 11  Android Ver     10838 non-null  object 
 12  day             10840 non-null  int64  
 13  month           10840 non-null  int64  
 14  year            10840 non-null  int64  
dtypes: float64(3), int64(5), object(7)
memory usage: 1.2+ MB
```

In [70]:
```
# check the null values
df.isnull().sum()
```

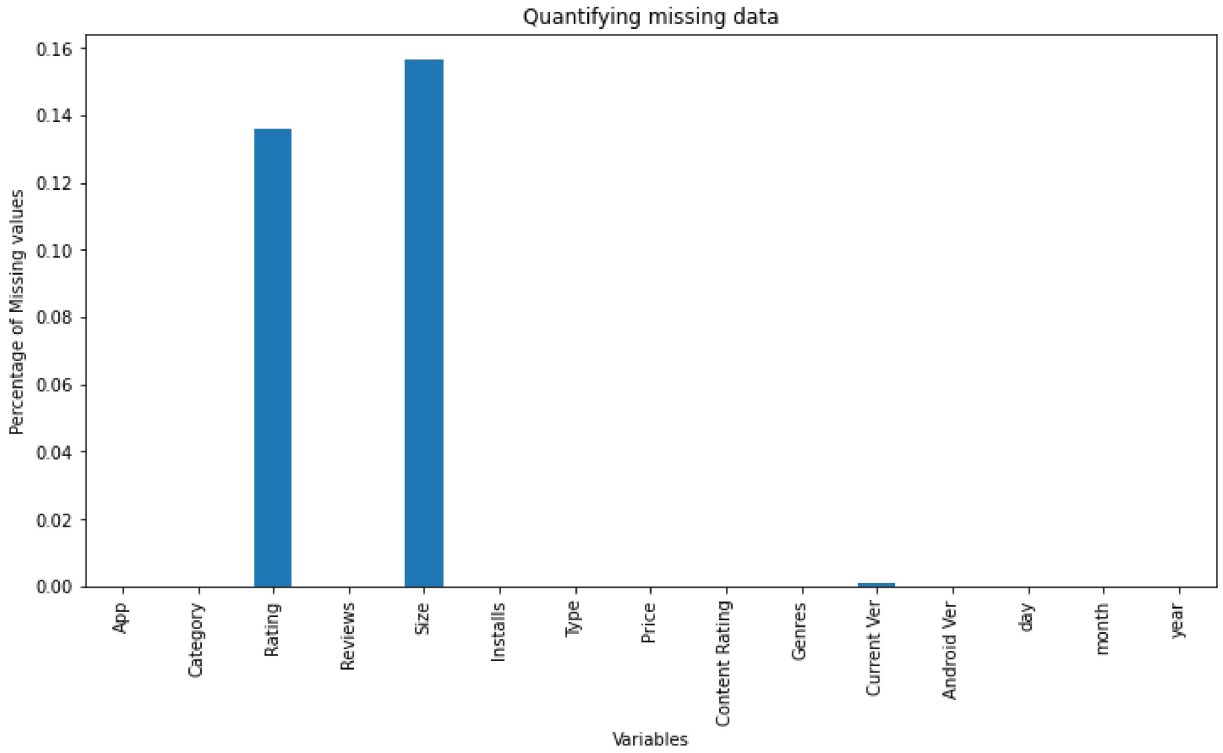
Out[70]:
```
App                0
Category           0
Rating          1474
Reviews            0
Size            1695
Installs           0
Type               1
Price              0
Content Rating     0
Genres             0
Current Ver        8
Android Ver        2
day                0
month              0
year               0
dtype: int64
```

In [71]:
```
# check the missing value
df.isnull().mean()*100
```

Out[71]:
```
App               0.000000
Category          0.000000
Rating           13.597786
Reviews           0.000000
Size             15.636531
Installs          0.000000
Type              0.009225
Price             0.000000
Content Rating    0.000000
Genres            0.000000
Current Ver       0.073801
Android Ver       0.018450
day               0.000000
month             0.000000
year              0.000000
dtype: float64
```

In [72]:
```
df.isnull().mean().plot.bar(figsize=(12,6))
plt.ylabel('Percentage of Missing values')
plt.xlabel('Variables')
plt.title('Quantifying missing data')
```

Out[72]:
```
Text(0.5, 1.0, 'Quantifying missing data')
```



In [73]:
```
#  check the Unique values
df.nunique()
```

Out[73]:
```
App               9659
Category            33
Rating              39
Reviews           6001
Size               423
Installs            20
Type                 2
Price               92
Content Rating       6
Genres             119
Current Ver       2831
Android Ver         33
day                 31
month               12
year                 9
dtype: int64
```

In [74]:
```
df.nunique().value_counts()
```

Out[74]:
```
33      2
9659    1
39      1
6001    1
423     1
20      1
2       1
92      1
6       1
119     1
2831    1
31      1
12      1
9       1
dtype: int64
```

In [75]:
```
df['App'].value_counts()
```

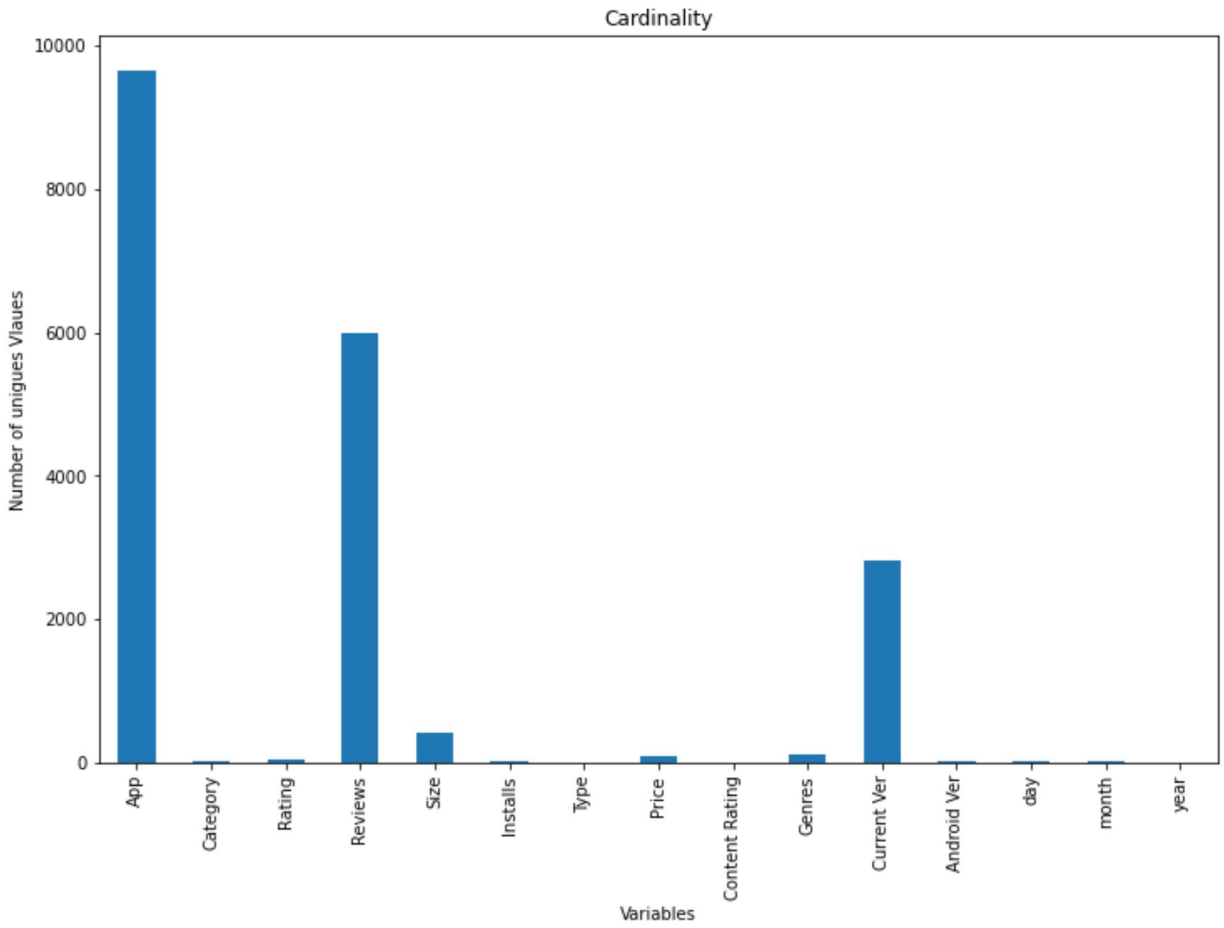
Out[75]:
```
ROBLOX                                                9
CBS Sports App - Scores, News, Stats & Watch Live     8
ESPN                                                  7
Duolingo: Learn Languages Free                        7
Candy Crush Saga                                      7
                                                     ..
Meet U - Get Friends for Snapchat, Kik & Instagram    1
U-Report                                              1
U of I Community Credit Union                         1
Waiting For U Launcher Theme                          1
iHoroscope - 2018 Daily Horoscope & Astrology         1
Name: App, Length: 9659, dtype: int64
```

In [76]:
```
df['Reviews']
```

Out[76]:
```
0           159
1           967
2         87510
3        215644
4           967
          ...
10835        38
10836         4
10837         3
10838       114
10839    398307
Name: Reviews, Length: 10840, dtype: int64
```

In [77]:
```
# show the unique values in graph
df.nunique().plot.bar(figsize=(12,8))
plt.xlabel("Variables")
plt.ylabel('Number of unigues Vlaues')
plt.title('Cardinality')
```

Out[77]:
```
Text(0.5, 1.0, 'Cardinality')
```



In [78]:
```
# check the Duplicated values
df.duplicated().sum()
```

Out[78]: 483

In [79]:
```
# Drop the Duplicated Values
df=df.drop_duplicates()
```

In [80]:
```
# after that Drop the duplicated values check the Shape
df.shape
```

Out[80]: (10357, 15)

Exploring the Data

Can you segregated the Categorical and Numerical Feature

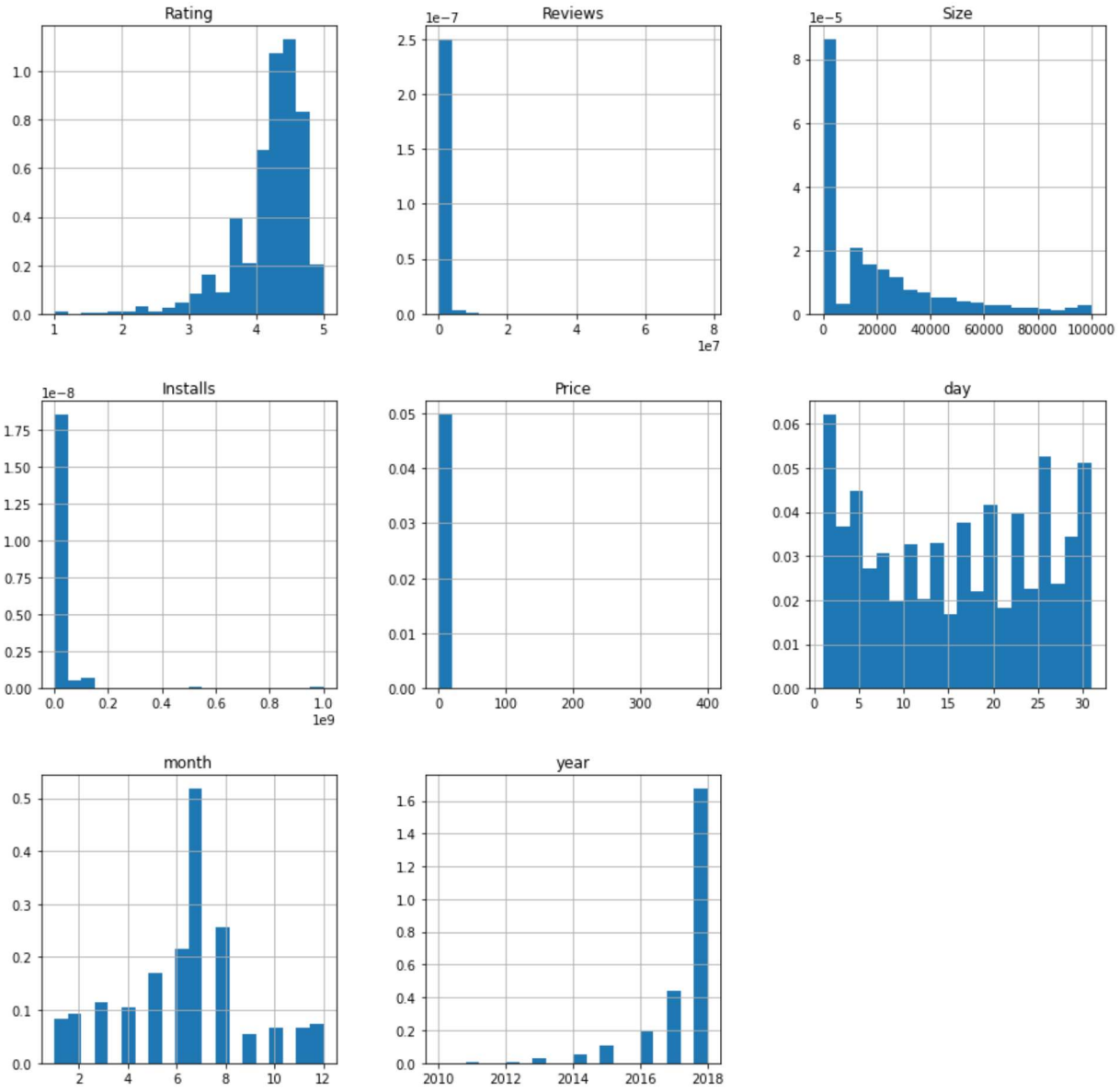
In [81]:
```
[feature for feature in df.columns ]
```

Out[81]:
```
['App',
 'Category',
 'Rating',
 'Reviews',
 'Size',
 'Installs',
 'Type',
 'Price',
 'Content Rating',
 'Genres',
 'Current Ver',
 'Android Ver',
 'day',
 'month',
 'year']
```

In [82]:
```
# Numerical Columns
num_columns=[feature for feature in df.columns if df[feature].dtype!='object']
num_columns
```

Out[82]: ['Rating', 'Reviews', 'Size', 'Installs', 'Price', 'day', 'month', 'year']

In [83]:
```
# show the Numerical Coloumns in histogram
df.hist(bins=20,figsize=(15,15),density=True)
plt.show()
```



In [84]:
```
# Categorical Columns
cat_columns=[feature for feature in df.columns if df[feature].dtype=='object']
cat_columns
```

Out[84]:
```
['App',
 'Category',
 'Type',
 'Content Rating',
 'Genres',
 'Current Ver',
 'Android Ver']
```

In [85]:
```
cat_columns
```

Out[85]:
```
['App',
 'Category',
 'Type',
 'Content Rating',
 'Genres',
 'Current Ver',
 'Android Ver']
```

In [86]:
```
# convert the the cat_columns in cat_df variable
cat_df=df[cat_columns]
```

In [87]:
```
cat_df
```

Out[87]:

	App	Category	Type	Content Rating	Genres	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	Free	Everyone	Art & Design	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	Free	Everyone	Art & Design;Pretend Play	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	Free	Everyone	Art & Design	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	Free	Teen	Art & Design	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	Free	Everyone	Art & Design;Creativity	1.1	4.4 and up
...	...	...	...	...	...	...	...
10835	Sya9a Maroc - FR	FAMILY	Free	Everyone	Education	1.48	4.1 and up
10836	Fr. Mike Schmitz Audio Teachings	FAMILY	Free	Everyone	Education	1.0	4.1 and up
10837	Parkinson Exercices FR	MEDICAL	Free	Everyone	Medical	1.0	2.2 and up
10838	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	Free	Mature 17+	Books & Reference	Varies with device	Varies with device
10839	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	Free	Everyone	Lifestyle	Varies with device	Varies with device

10357 rows × 7 columns

In [88]:
```
df['App'].value_counts()
```

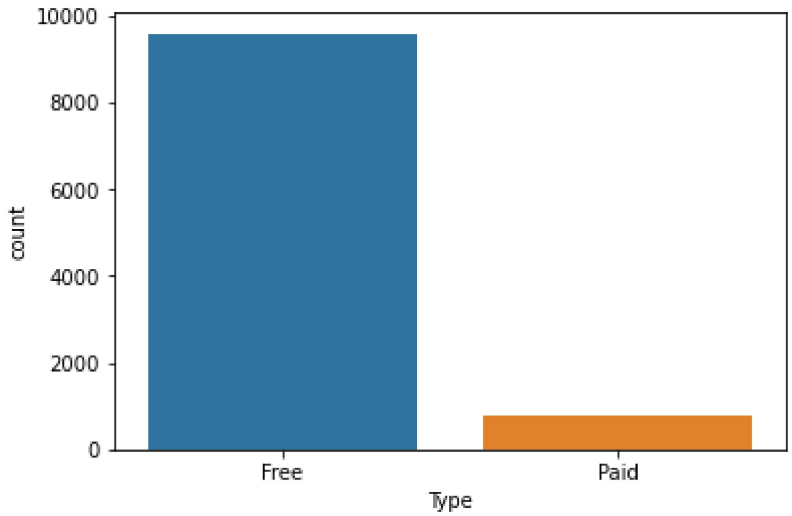
Out[88]:
```
ROBLOX                                               9
8 Ball Pool                                          7
Bubble Shooter                                       6
Helix Jump                                           6
Zombie Catchers                                      6
                                                    ..
Popsicle Launcher for Android P 9.0 launcher         1
PixelLab - Text on pictures                          1
P Launcher for Android™ 9.0                          1
Pacify (Android P theme) - Theme for Xperia™         1
iHoroscope - 2018 Daily Horoscope & Astrology        1
Name: App, Length: 9659, dtype: int64
```

In [89]:
```
cat_df['Type'].value_counts()
```

Out[89]:
```
Free    9591
Paid     765
Name: Type, dtype: int64
```

In [90]:
```
sns.countplot(cat_df['Type'])
```

Out[90]:
```
<AxesSubplot:xlabel='Type', ylabel='count'>
```

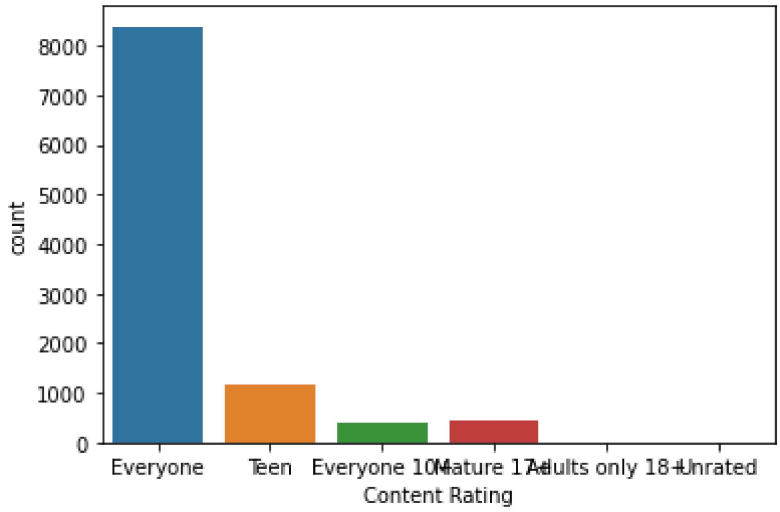


In [91]:
```
cat_df['Content Rating'].value_counts()
```

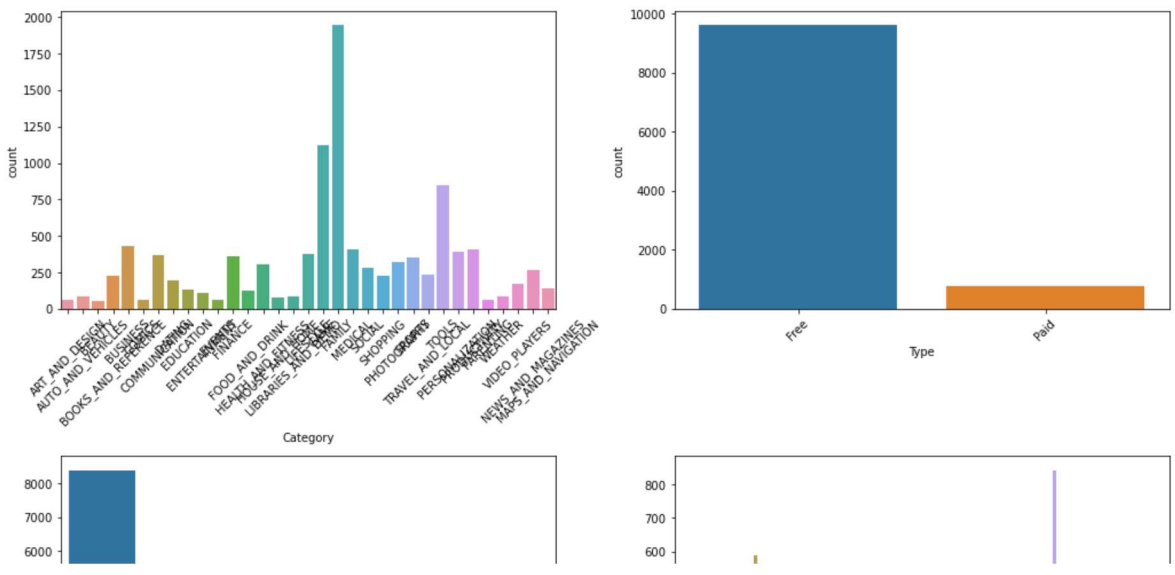
Out[91]:
```
Everyone           8382
Teen               1146
Mature 17+          447
Everyone 10+        377
Adults only 18+       3
Unrated               2
Name: Content Rating, dtype: int64
```

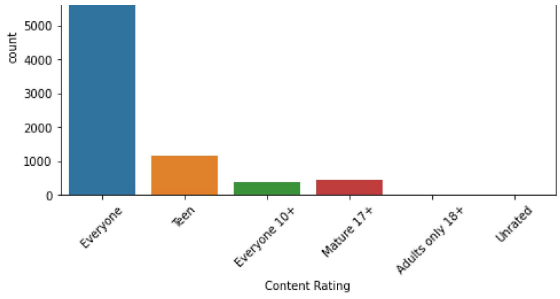
In [92]:
```
sns.countplot(cat_df['Content Rating'])
```

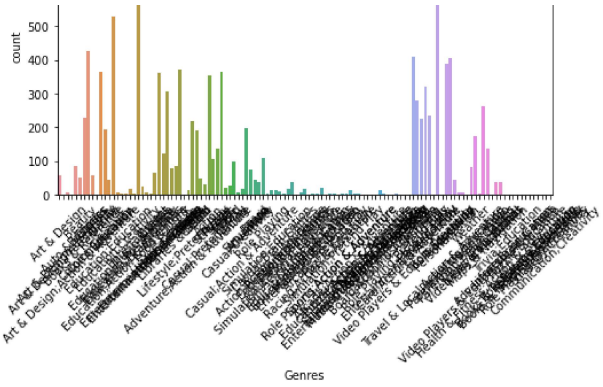
Out[92]: <AxesSubplot:xlabel='Content Rating', ylabel='count'>



In [93]:
```
plt.figure(figsize=(15,12))
plt.suptitle('Univeriate Analysis of Categorical Feature',fontsize = 20, fontweig
cat_columns=['Category','Type','Content Rating','Genres']
for i in range(0 , len(cat_columns)):
    plt.subplot(2 , 2, i+1)
    sns.countplot(x=df[cat_columns[i]])
    plt.xlabel(cat_columns[i])
    plt.xticks(rotation=45)
    plt.tight_layout()
```





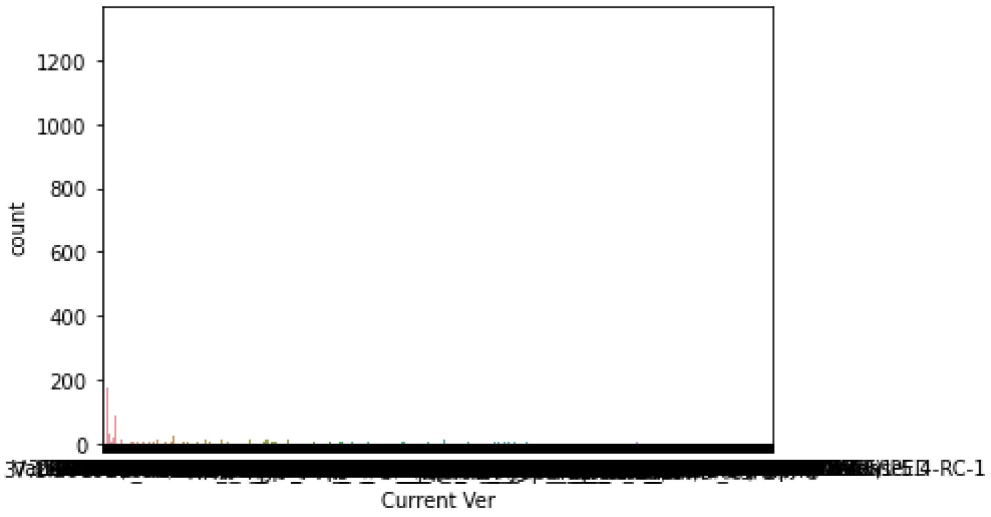


In [94]:
```
cat_df['Current Ver'].value_counts()
```

Out[94]:
```
Varies with device    1302
1.0                    802
1.1                    260
1.2                    177
2.0                    149
                      ...
3.18.5                   1
1.3.A.2.9                1
9.9.1.1910               1
7.1.34.28                1
2.0.148.0                1
Name: Current Ver, Length: 2831, dtype: int64
```

In [95]:
```
sns.countplot(cat_df['Current Ver'])
```

Out[95]:
```
<AxesSubplot:xlabel='Current Ver', ylabel='count'>
```

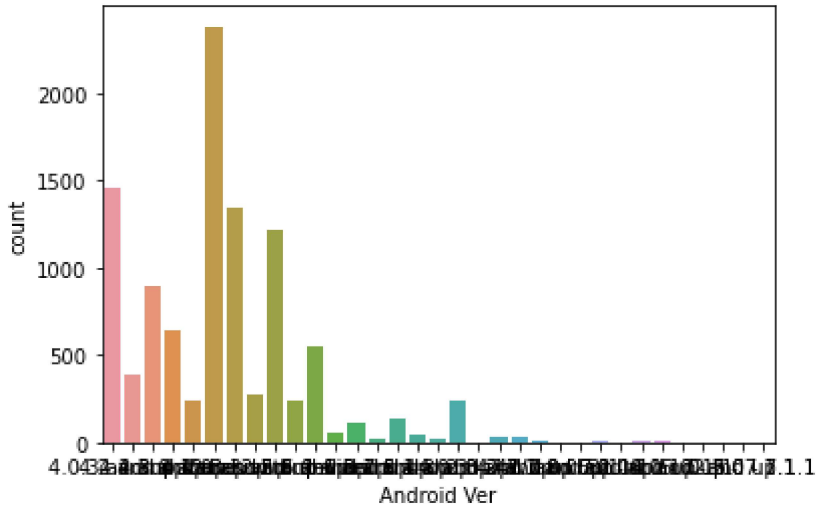


In [96]:
```
cat_df['Android Ver'].value_counts()  # top 10 Andriod Ver
```

Out[96]:
```
4.1 and up            2379
4.0.3 and up          1451
4.0 and up            1337
Varies with device    1221
4.4 and up             894
2.3 and up             643
5.0 and up             546
4.2 and up             387
2.3.3 and up           279
2.2 and up             239
3.0 and up             237
4.3 and up             235
2.1 and up             133
1.6 and up             116
6.0 and up              58
7.0 and up              42
3.2 and up              36
2.0 and up              32
5.1 and up              22
1.5 and up              20
4.4W and up             11
3.1 and up              10
2.0.1 and up             7
8.0 and up               6
7.1 and up               3
4.0.3 - 7.1.1            2
5.0 - 8.0                2
1.0 and up               2
7.0 - 7.1.1              1
4.1 - 7.1.1              1
5.0 - 6.0                1
2.2 - 7.1.1              1
5.0 - 7.1.1              1
Name: Android Ver, dtype: int64
```

In [97]:
```
sns.countplot(cat_df['Android Ver'])
```

Out[97]: <AxesSubplot:xlabel='Android Ver', ylabel='count'>



Which category has largest number of installation

In [98]:
```
df.groupby(['Category'])['Installs'].sum().sort_values(ascending=False)
```

Out[98]:
```
Category
GAME                   31544024415
COMMUNICATION          24152276251
SOCIAL                 12513867902
PRODUCTIVITY           12463091369
TOOLS                  11452771915
FAMILY                 10041692505
PHOTOGRAPHY             9721247655
TRAVEL_AND_LOCAL        6361887146
VIDEO_PLAYERS           6222002720
NEWS_AND_MAGAZINES      5393217760
SHOPPING                2573348785
ENTERTAINMENT           2455660000
PERSONALIZATION         2074494782
BOOKS_AND_REFERENCE     1916469576
SPORTS                  1528574498
HEALTH_AND_FITNESS      1361022512
BUSINESS                 863664865
FINANCE                  770348734
MAPS_AND_NAVIGATION      724281890
LIFESTYLE                534823539
EDUCATION                533952000
WEATHER                  426100520
FOOD_AND_DRINK           257898751
DATING                   206536107
HOUSE_AND_HOME           125212461
ART_AND_DESIGN           124338100
LIBRARIES_AND_DEMO        62995910
COMICS                    56086150
AUTO_AND_VEHICLES         53130211
MEDICAL                   42204177
PARENTING                 31521110
BEAUTY                    27197050
EVENTS                    15973161
Name: Installs, dtype: int64
```

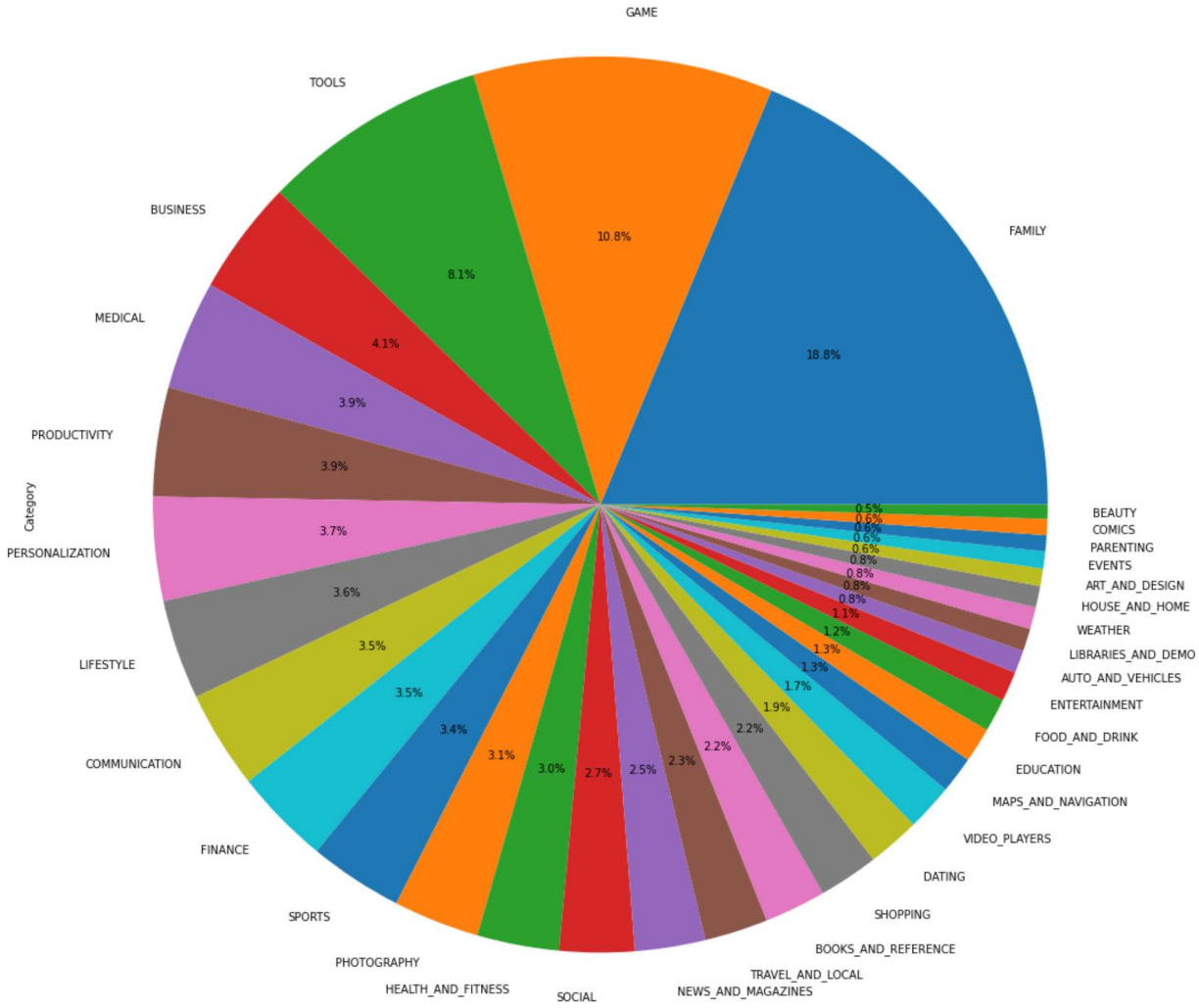
In [99]:
```
df.groupby(['Category'])['Installs'].sum().nlargest(10).plot.pie #  (kind="bar",j
```

Out[99]:
```
<bound method PlotAccessor.pie of <pandas.plotting._core.PlotAccessor object at
0x0000019380A155E0>>
```

Which one will be a most popular category

In [100]:
```
cat_df['Category'].value_counts().plot.pie(figsize=(18,20),autopct = "%1.1f%%")
```

Out[100]: <AxesSubplot:ylabel='Category'>



Statistical Based Analysis

In [101]:
```
# Check the statistical graph
df.describe()
```

Out[101]:

	Rating	Reviews	Size	Installs	Price	day	
count	8892.000000	1.035700e+04	8831.000000	1.035700e+04	10357.000000	10357.000000	10357
mean	4.187877	4.059046e+05	19585.031208	1.415776e+07	1.030800	15.619098	6
std	0.522377	2.696778e+06	24105.774828	8.023955e+07	16.278625	9.528658	2
min	1.000000	0.000000e+00	1.000000	0.000000e+00	0.000000	1.000000	1
25%	4.000000	3.200000e+01	5.500000	1.000000e+03	0.000000	6.000000	5
50%	4.300000	1.680000e+03	13000.000000	1.000000e+05	0.000000	16.000000	7
75%	4.500000	4.641600e+04	30000.000000	1.000000e+06	0.000000	24.000000	8
max	5.000000	7.815831e+07	100000.000000	1.000000e+09	400.000000	31.000000	12

In [102]:
```
df.describe().T
```

Out[102]:

	count	mean	std	min	25%	50%	75%	max
Rating	8892.0	4.187877e+00	5.223767e-01	1.0	4.0	4.3	4.5	5.000000e+00
Reviews	10357.0	4.059046e+05	2.696778e+06	0.0	32.0	1680.0	46416.0	7.815831e+07
Size	8831.0	1.958503e+04	2.410577e+04	1.0	5.5	13000.0	30000.0	1.000000e+05
Installs	10357.0	1.415776e+07	8.023955e+07	0.0	1000.0	100000.0	1000000.0	1.000000e+09
Price	10357.0	1.030800e+00	1.627863e+01	0.0	0.0	0.0	0.0	4.000000e+02
day	10357.0	1.561910e+01	9.528658e+00	1.0	6.0	16.0	24.0	3.100000e+01
month	10357.0	6.397026e+00	2.606359e+00	1.0	5.0	7.0	8.0	1.200000e+01
year	10357.0	2.017383e+03	1.112766e+00	2010.0	2017.0	2018.0	2018.0	2.018000e+03

In [103]:
```
# check the covariance
df.cov()
```

Out[103]:

	Rating	Reviews	Size	Installs	Price	day
Rating	2.728774e-01	1.043033e+05	1.050235e+03	2.296016e+06	-1.890523e-01	-5.085254e-02
Reviews	1.043033e+05	7.272611e+12	8.920583e+09	1.374060e+14	-4.134059e+05	-6.341046e+05
Size	1.050235e+03	8.920583e+09	5.810884e+08	1.704640e+11	-1.004941e+04	-2.001054e+03
Installs	2.296016e+06	1.374060e+14	1.704640e+11	6.438386e+15	-1.455975e+07	-3.329448e+07
Price	-1.890523e-01	-4.134059e+05	-1.004941e+04	-1.455975e+07	2.649936e+02	-1.315355e+00
day	-5.085254e-02	-6.341046e+05	-2.001054e+03	-3.329448e+07	-1.315355e+00	9.079532e+01
month	2.254426e-02	3.140523e+05	2.397261e+03	1.126110e+07	2.753658e-01	-1.846923e+00
year	7.965134e-02	2.272974e+05	4.903873e+03	7.801989e+06	-1.019317e-01	-1.615820e-01

In [104]:
```
# check the correlation
df.corr()
```

Out[104]:

	Rating	Reviews	Size	Installs	Price	day	month	year
Rating	1.000000	0.068732	0.076437	0.050886	-0.022355	-0.010102	0.016999	0.136547
Reviews	0.068732	1.000000	0.231009	0.634997	-0.009417	-0.024677	0.044681	0.075744
Size	0.076437	0.231009	1.000000	0.166187	-0.023659	-0.008872	0.036857	0.175555
Installs	0.050886	0.634997	0.166187	1.000000	-0.011147	-0.043546	0.053847	0.087380
Price	-0.022355	-0.009417	-0.023659	-0.011147	1.000000	-0.008480	0.006490	-0.005627
day	-0.010102	-0.024677	-0.008872	-0.043546	-0.008480	1.000000	-0.074367	-0.015239
month	0.016999	0.044681	0.036857	0.053847	0.006490	-0.074367	1.000000	-0.190497
year	0.136547	0.075744	0.175555	0.087380	-0.005627	-0.015239	-0.190497	1.000000

In [105]:
```
# Check the standard deviation
df.std()
```

Out[105]:
```
Rating      5.223767e-01
Reviews     2.696778e+06
Size        2.410577e+04
Installs    8.023955e+07
Price       1.627863e+01
day         9.528658e+00
month       2.606359e+00
year        1.112766e+00
dtype: float64
```

In [106]:
```
# Check the minimun values
df.min()
```

Out[106]:
```
App                "i DT" Fútbol. Todos Somos Técnicos.
Category                                  ART_AND_DESIGN
Rating                                               1.0
Reviews                                                0
Size                                                 1.0
Installs                                               0
Price                                                0.0
Content Rating                          Adults only 18+
Genres                                           Action
day                                                   1
month                                                 1
year                                               2010
dtype: object
```

In [107]:
```
df[['App','Rating']].min()  # indicate the got minimunm rating
```

Out[107]:
```
App       "i DT" Fútbol. Todos Somos Técnicos.
Rating                                     1.0
dtype: object
```

In [108]:
```
df.mean()  # Check the mean
```

Out[108]:
```
Rating      4.187877e+00
Reviews     4.059046e+05
Size        1.958503e+04
Installs    1.415776e+07
Price       1.030800e+00
day         1.561910e+01
month       6.397026e+00
year        2.017383e+03
dtype: float64
```

In [109]:
```
df.median() # Check the Median
```

Out[109]:
```
Rating           4.3
Reviews       1680.0
Size         13000.0
Installs    100000.0
Price            0.0
day             16.0
month            7.0
year          2018.0
dtype: float64
```

In [110]:
```
df.count()
```

Out[110]:
```
App               10357
Category          10357
Rating             8892
Reviews           10357
Size               8831
Installs          10357
Type              10356
Price             10357
Content Rating    10357
Genres            10357
Current Ver       10349
Android Ver       10355
day               10357
month             10357
year              10357
dtype: int64
```

In [111]:
```
# Check the Maximum Values
df.max()
```

Out[111]:
```
App               🔥 Football Wallpapers 4K | Full HD Backgrounds 😍
Category                                                    WEATHER
Rating                                                          5.0
Reviews                                                    78158306
Size                                                      100000.0
Installs                                                1000000000
Price                                                        400.0
Content Rating                                             Unrated
Genres                                                        Word
day                                                             31
month                                                           12
year                                                          2018
dtype: object
```

Which one is a most popular app and category got 5 Rating

In [112]:
```
df[['App','Category','Rating']].max() # indicate the got maximum rating ,
```

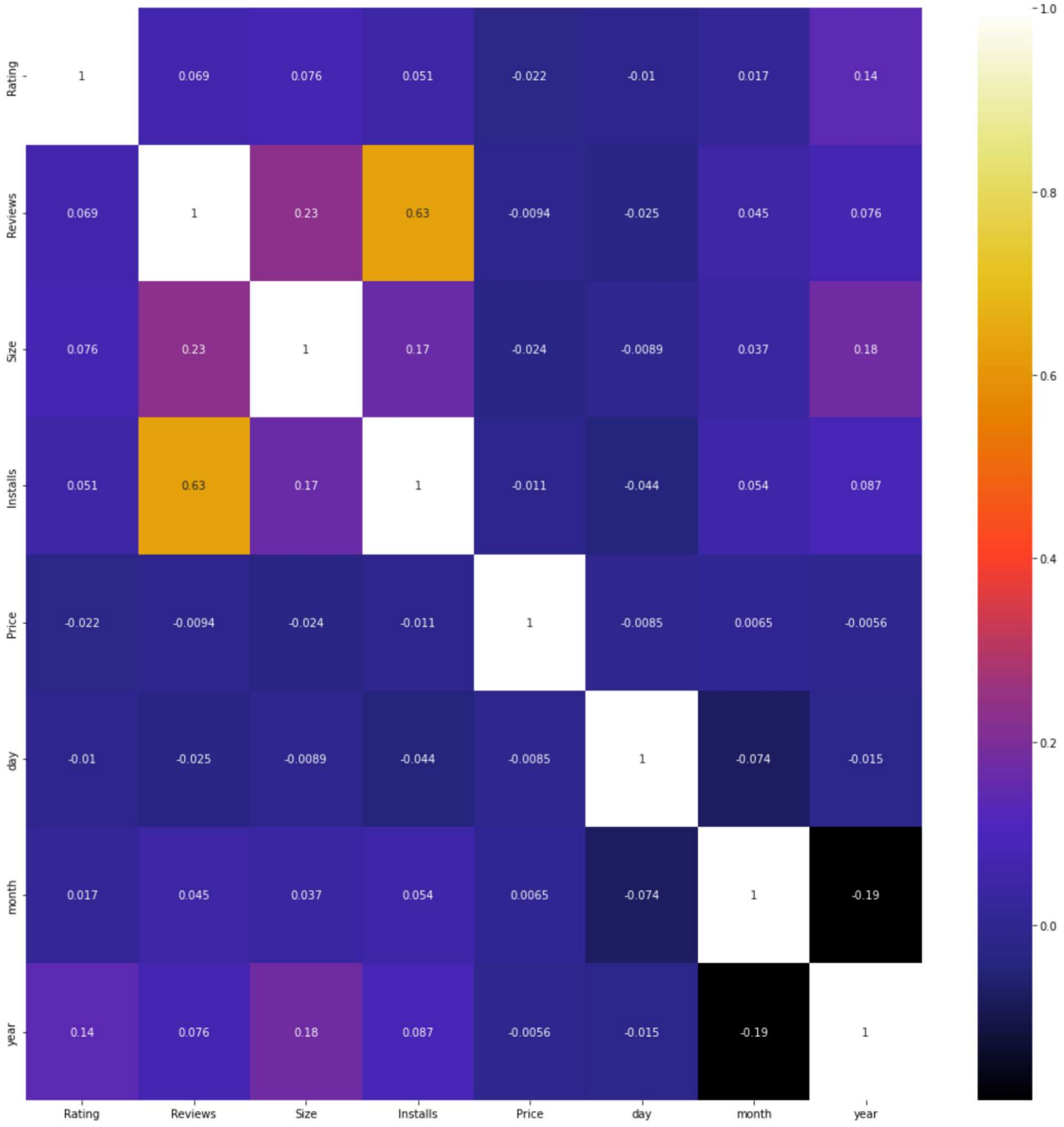
Out[112]:
```
App         🔥 Football Wallpapers 4K | Full HD Backgrounds 😍
Category                                              WEATHER
Rating                                                    5.0
dtype: object
```

Check the Correlation using the Heatmap

In [113]:

```
plt.figure(figsize=(18,18))
sns.heatmap(df.corr(),cmap ='CMRmap',annot=True)
```

Out[113]: <AxesSubplot:>



In [114]:
```
# check the Skewness
df.skew()
```

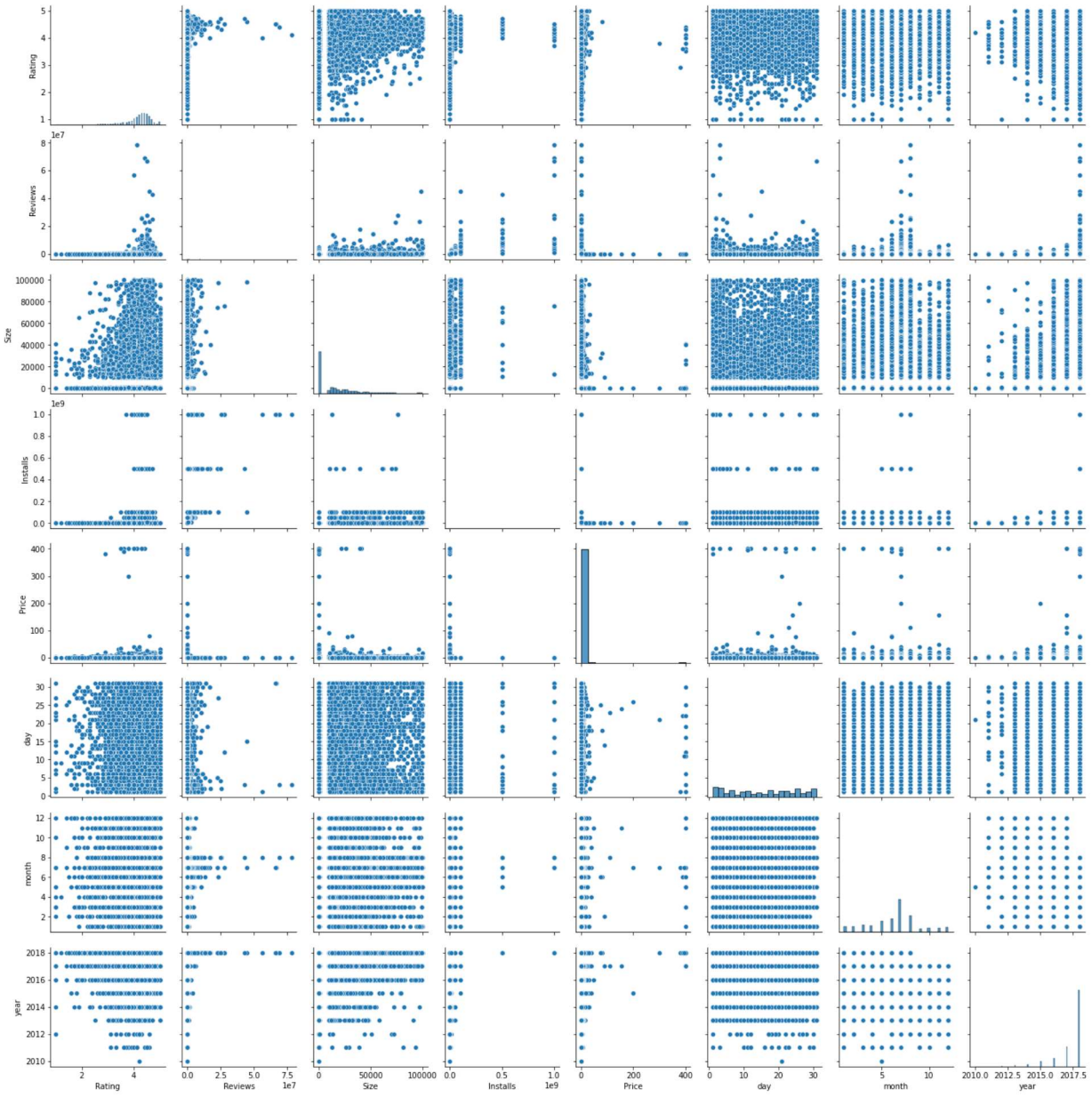
Out[114]:
```
Rating      -1.823517
Reviews     17.467335
Size         1.384928
Installs    10.126705
Price       23.324381
day         -0.002970
month       -0.088336
year        -2.246706
dtype: float64
```

In [ ]:
```

```

In [115]:
```
# check the pairplot
plt.figure(figsize=(20,20))
sns.pairplot(df)
plt.show()
```

```
<Figure size 1440x1440 with 0 Axes>
```



In [116]:
```
# check the quantile
df.quantile()
```

Out[116]:
```
Rating           4.3
Reviews       1680.0
Size         13000.0
Installs    100000.0
Price            0.0
day             16.0
month            7.0
year          2018.0
Name: 0.5, dtype: float64
```

Handle the missing values

In [117]:
```
df.isnull().sum()
```

Out[117]:
```
App                0
Category           0
Rating          1465
Reviews            0
Size            1526
Installs           0
Type               1
Price              0
Content Rating     0
Genres             0
Current Ver        8
Android Ver        2
day                0
month              0
year               0
dtype: int64
```

In [118]:
```
df.isnull().sum().sum()
```

Out[118]:
```
3002
```

In [119]:
```
df_copy=df.copy()
```

In [120]:
```
null_columns=[variable for variable in df_copy.columns if df_copy[variable].isnul
null_columns
```

Out[120]:
```
['Rating', 'Size', 'Type', 'Current Ver', 'Android Ver']
```

In [121]:
```
df_copy['Android Ver']
```

Out[121]:
```
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
                  ...
10835            4.1 and up
10836            4.1 and up
10837            2.2 and up
10838    Varies with device
10839    Varies with device
Name: Android Ver, Length: 10357, dtype: object
```

In [122]:
```
df_copy['Rating'].dtype # indiacte the Rating is Dependent columns
```

Out[122]:
```
dtype('float64')
```

In [123]:
```
# We are replacing the missing values of the Dependent columns with mean() using
df_copy['Rating']=df_copy['Rating'].fillna(df_copy['Rating'].mean())
df_copy['Size']= df_copy['Size'].fillna(df_copy['Size'].median()) # is there outl
df_copy['Type']=df_copy['Type'].fillna(df_copy['Type'].mode()[0]) # indicate Type
df_copy['Current Ver']=df_copy['Current Ver'].fillna(df_copy['Current Ver'].mode(
df_copy['Android Ver']=df_copy['Android Ver'].fillna(df_copy['Android Ver'].mode(
```

In [124]:
```
# after that check the missing value
```

In [125]:
```
df_copy.isnull().sum().sum()
```

Out[125]:
```
0
```