```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv('/content/drive/MyDrive/Datasets/googleplaystore.csv')
```

```
df.columns
```

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

```
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
```

```
df.describe()
```

	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

Preprocessing of Data

```
df.duplicated().sum()
```

```
np.int64(483)
```

```
df.drop_duplicates(inplace=True)
```

```
df.duplicated().sum()
```

```
np.int64(0)
```

```
df.columns
```

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

```
df['Reviews'].dtype
```

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

```
df1=df.copy()
```

```
df1.reset_index(drop=True,inplace=True)
```

```
df1[~df1.Reviews.str.isnumeric()]
```

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver

```
df1=df1.drop(index=9990)
```

```
df1[~df1.Reviews.str.isnumeric()]
```

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver

```
df1["Reviews"]=df1['Reviews'].astype(int)
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10357 entries, 0 to 10357
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10357 non-null  object 
 1   Category        10357 non-null  object 
 2   Rating          8892 non-null   float64
 3   Reviews         10357 non-null  int64  
 4   Size            10357 non-null  object 
 5   Installs        10357 non-null  object 
 6   Type            10356 non-null  object 
 7   Price           10357 non-null  object 
 8   Content Rating  10357 non-null  object 
 9   Genres          10357 non-null  object 
 10  Last Updated    10357 non-null  object 
 11  Current Ver     10349 non-null  object 
 12  Android Ver     10355 non-null  object 
dtypes: float64(1), int64(1), object(11)
memory usage: 1.1+ MB
```

```
df["Size"].unique()
```

```
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)
```

```
def size_process(item):
  if str(item)[-1]=='M':
    res=float(str(item).replace('M',' '))
    res=res*1024
    return res
  elif str(item)[-1]=='k':
    res=float(str(item).replace('k',' '))
    return res
  else:
    return str(np.nan)
```

```
df1['Size']=df1['Size'].apply(size_process)
```

```
df1.Size.dtype
```

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

```
df1.Size=df1.Size.astype('float')
```

```
df1.Size.dtype
```

```
dtype('float64')
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10357 entries, 0 to 10357
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10357 non-null  object 
 1   Category        10357 non-null  object 
 2   Rating          8892 non-null   float64
 3   Reviews         10357 non-null  int64  
 4   Size            8831 non-null   float64
 5   Installs        10357 non-null  object 
 6   Type            10356 non-null  object 
 7   Price           10357 non-null  object 
 8   Content Rating  10357 non-null  object 
 9   Genres          10357 non-null  object 
 10  Last Updated    10357 non-null  object 
 11  Current Ver     10349 non-null  object 
 12  Android Ver     10355 non-null  object 
dtypes: float64(2), int64(1), object(10)
memory usage: 1.1+ MB
```

```
df1.Installs.unique()
```

```
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)
```

```
df1['Installs'] = df1['Installs'].str.replace('+','', regex=False).str.replace(',','', regex=False)
```

```
df1['Installs']=df1['Installs'].astype(int)
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10357 entries, 0 to 10357
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10357 non-null  object 
 1   Category        10357 non-null  object 
 2   Rating          8892 non-null   float64
 3   Reviews         10357 non-null  int64  
 4   Size            8831 non-null   float64
 5   Installs        10357 non-null  int64  
 6   Type            10356 non-null  object 
 7   Price           10357 non-null  object 
 8   Content Rating  10357 non-null  object 
 9   Genres          10357 non-null  object 
 10  Last Updated    10357 non-null  object 
 11  Current Ver     10349 non-null  object 
 12  Android Ver     10355 non-null  object 
dtypes: float64(2), int64(2), object(9)
memory usage: 1.1+ MB
```

```
df1.Price.unique()
```

```
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)
```

```
df1['Price'] = df1['Price'].str.replace('$','', regex=False)
```

```
df1['Price']=df1['Price'].astype(float)
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10357 entries, 0 to 10357
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10357 non-null  object 
 1   Category        10357 non-null  object 
 2   Rating          8892 non-null   float64
 3   Reviews         10357 non-null  int64  
 4   Size            8831 non-null   float64
 5   Installs        10357 non-null  int64  
 6   Type            10356 non-null  object 
 7   Price           10357 non-null  float64
 8   Content Rating  10357 non-null  object 
 9   Genres          10357 non-null  object 
 10  Last Updated    10357 non-null  object 
 11  Current Ver     10349 non-null  object 
 12  Android Ver     10355 non-null  object 
dtypes: float64(3), int64(2), object(8)
memory usage: 1.1+ MB
```

```
df1['Last Updated'] = pd.to_datetime(df1['Last Updated'])
```

```
df1['day']=df1['Last Updated'].dt.day
df1['month']=df1['Last Updated'].dt.month
df1['year']=df1['Last Updated'].dt.year
```

```
df1.dtypes
```

	0
App	object
Category	object
Rating	float64
Reviews	int64
Size	float64
Installs	int64
Type	object
Price	float64
Content Rating	object
Genres	object
Last Updated	datetime64[ns]
Current Ver	object
Android Ver	object
day	int32
month	int32
year	int32

dtype: object

```
df1.drop('Last Updated',axis=1,inplace=True)
```

```
df1['Current Ver'].unique()
```

```
array(['1.0.0', '2.0.0', '1.2.4', ..., '1.0.612928', '0.3.4', '2.0.148.0'],
      dtype=object)
```

```
df1['Android Ver'].unique()
```

```
array(['4.0.3 and up', '4.2 and up', '4.4 and up', '2.3 and up',
       '3.0 and up', '4.1 and up', '4.0 and up', '2.3.3 and up',
       'Varies with device', '2.2 and up', '5.0 and up', '6.0 and up',
       '1.6 and up', '1.5 and up', '2.1 and up', '7.0 and up',
       '5.1 and up', '4.3 and up', '4.0.3 - 7.1.1', '2.0 and up',
       '3.2 and up', '4.4W and up', '7.1 and up', '7.0 - 7.1.1',
       '8.0 and up', '5.0 - 8.0', '3.1 and up', '2.0.1 and up',
       '4.1 - 7.1.1', nan, '5.0 - 6.0', '1.0 and up', '2.2 - 7.1.1',
       '5.0 - 7.1.1'], dtype=object)
```

```
df1['Android Ver']=df1['Android Ver'].str.replace(' and up','').str.replace('Varies with device','')
```

```
df1[df1.duplicated("App")]
```

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Current
260	Quick PDF Scanner + OCR FREE	BUSINESS	4.2	80804	NaN	5000000	Free	0.0	Everyone	Business	Varies with de
261	OfficeSuite : Free Office + PDF Editor	BUSINESS	4.3	1002859	35840.0	100000000	Free	0.0	Everyone	Business	9.7.1
262	Slack	BUSINESS	4.4	51510	NaN	5000000	Free	0.0	Everyone	Business	Varies with de
348	Messenger – Text and Video Chat for Free	COMMUNICATION	4.0	56646578	NaN	1000000000	Free	0.0	Everyone	Communication	Varies with de
349	imo free video calls and chat	COMMUNICATION	4.3	4785988	11264.0	500000000	Free	0.0	Everyone	Communication	9.8.00000001
...	...	...	...	...	...	...	...	...	...	...	...

```
df1=df1.drop_duplicates(subset="App",keep="first")
```

```
df1[df1.duplicated("App")]
```

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Current Ver	Android Ver	day	month	year

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9659 entries, 0 to 10357
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   App             9659 non-null   object
 1   Category        9659 non-null   object
 2   Rating          8196 non-null   float64
 3   Reviews         9659 non-null   int64
 4   Size            8432 non-null   float64
 5   Installs        9659 non-null   int64
 6   Type            9658 non-null   object
 7   Price           9659 non-null   float64
 8   Content Rating  9659 non-null   object
 9   Genres          9659 non-null   object
 10  Current Ver     9651 non-null   object
 11  Android Ver     9657 non-null   object
 12  day             9659 non-null   int32
 13  month           9659 non-null   int32
 14  year            9659 non-null   int32
dtypes: float64(3), int32(3), int64(2), object(7)
memory usage: 1.1+ MB
```

```
df1.isnull().sum()
```

	0
App	0
Category	0
Rating	1463
Reviews	0
Size	1227
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

```
df1.dropna(inplace=True)
```

Exploratory Data Analysis

```
df1.columns
```

```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Current Ver', 'Android Ver',
       'day', 'month', 'year'],
      dtype='object')
```

```
categorical_features=[feature for feature in df1.columns if df1[feature].dtype=='O']
```

```
categorical_features
```

```
['App',
 'Category',
 'Type',
```

```
 'Content Rating',
 'Genres',
 'Current Ver',
 'Android Ver']
```

```
numerical_features=[feature for feature in df1.columns if df1[feature].dtype!='O']
```

```
numerical_features
```

```
['Rating', 'Reviews', 'Size', 'Installs', 'Price', 'day', 'month', 'year']
```

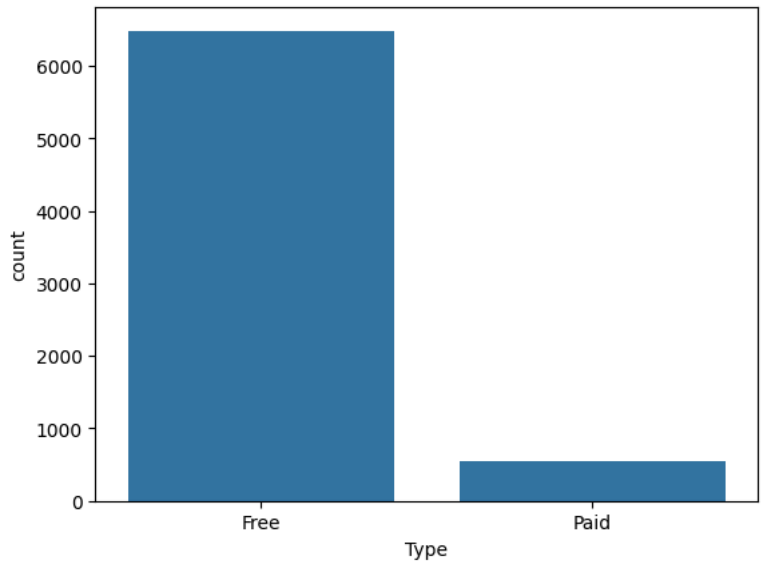
```
#for categorical features>> Frequency Plot,Bar Chart,Pie Chart....
#for numerical features>> Histogram,Dist Plot,Box Plot ,Line Chart ,Scatterplot.....
```

```
for col in categorical_features:
  print(f"{col}:{df1[col].value_counts(normalize=True)*100}")
```

```
EVENTS                  0.541253
BEAUTY                  0.526990
Name: proportion, dtype: float64
Type:Type
Free    92.323031
Paid     7.676969
Name: proportion, dtype: float64
Content Rating:Content Rating
Everyone           80.843185
Teen               11.095286
Mature 17+          4.329868
Everyone 10+        3.688933
Adults only 18+     0.028486
Unrated             0.014243
Name: proportion, dtype: float64
Genres:Genres
Tools                        8.901866
Entertainment                5.939325
Education                    5.569007
Action                       3.931064
Personalization              3.902578
                               ...
Puzzle;Education             0.014243
Role Playing;Brain Games     0.014243
Strategy;Education           0.014243
Racing;Pretend Play          0.014243
Strategy;Creativity          0.014243
Name: proportion, Length: 111, dtype: float64
Current Ver:Current Ver
1.0              6.395100
1.1              2.706167
1.2              1.780373
2.0              1.652186
1.3              1.637943
                   ...
1.03.123.0713    0.014243
1.0.0.96         0.014243
41.0             0.014243
2.4.1.485300     0.014243
12.2             0.014243
Name: proportion, Length: 2508, dtype: float64
Android Ver:Android Ver
4.1              24.440963
4.0.3            15.766985
4.0              14.513602
4.4               9.542800
2.3               7.662726
5.0               6.010540
4.2               4.244410
2.3.3             3.233158
3.0               2.862840
2.2               2.862840
4.3               2.506765
2.1               1.595214
1.6               1.239140
                  0.655177
6.0               0.555476
7.0               0.555476
3.2               0.441533
```

```
sns.countplot(x=df1['Type'])
```

```
<Axes: xlabel='Type', ylabel='count'>
```




```
df['Type'].value_counts(normalize=True)*100
```

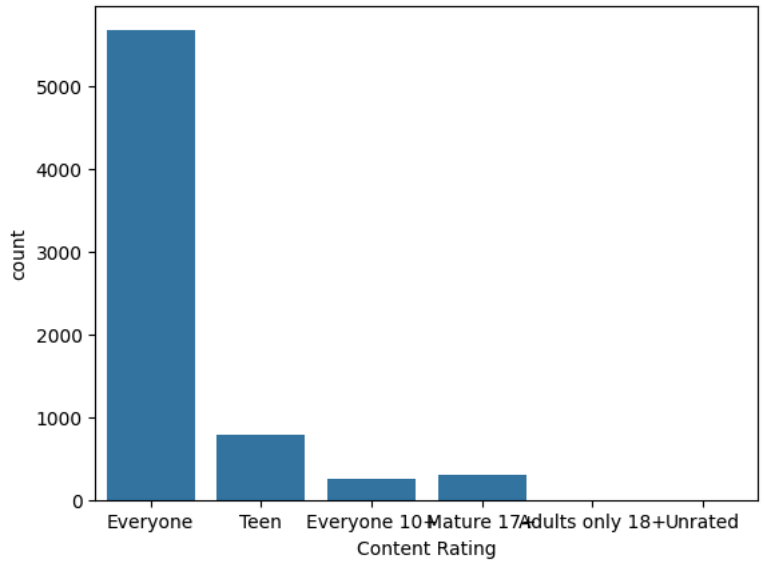
Type	proportion
Free	92.604036
Paid	7.386309
0	0.009655

dtype: float64

Approx 93% apps are free and 7% are paid

```
sns.countplot(x=df1['Content Rating'])
```

```
<Axes: xlabel='Content Rating', ylabel='count'>
```




```
df1['Content Rating'].value_counts(normalize=True)*100
```

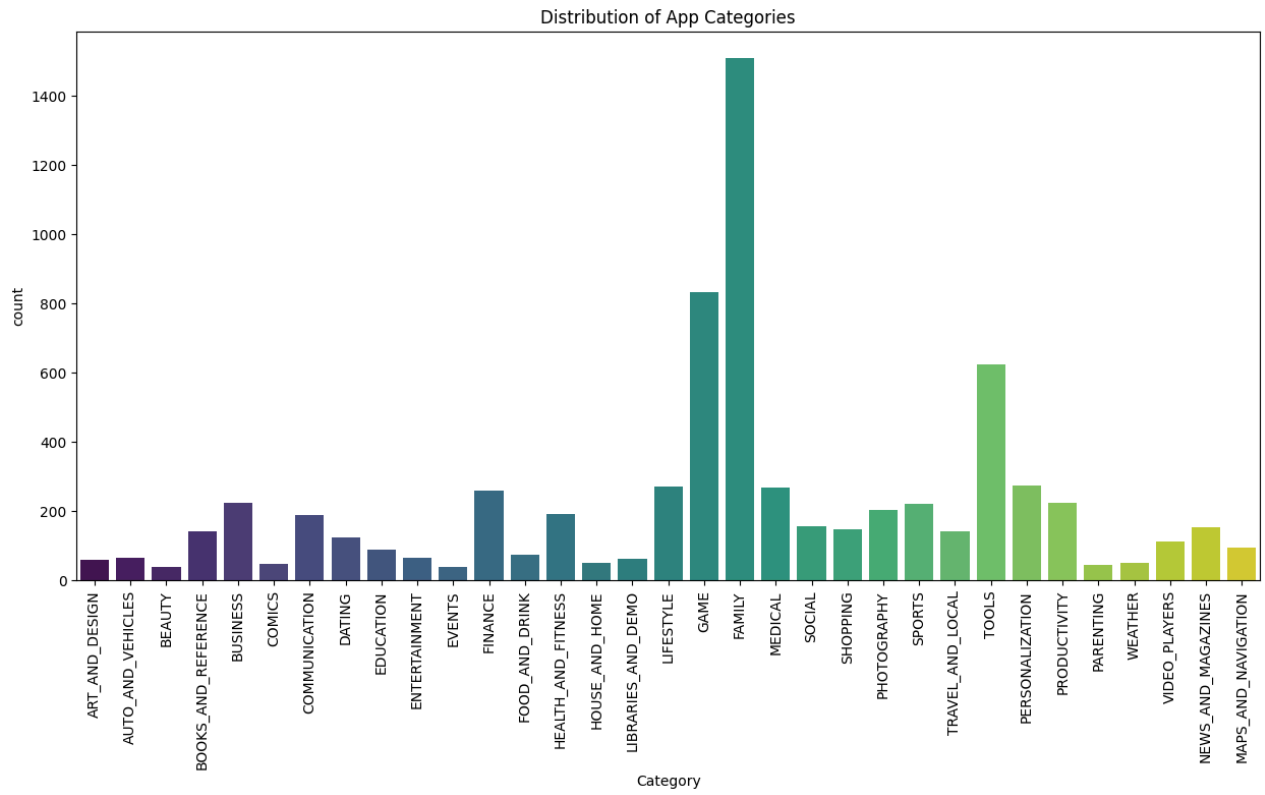
Content Rating	proportion
Everyone	80.843185
Teen	11.095286
Mature 17+	4.329868
Everyone 10+	3.688933
Adults only 18+	0.028486
Unrated	0.014243

dtype: float64

Maximum Apps are Available for everyone i.e. approx 83%

```
#Question >> How many apps belongs to each category
```

```
plt.figure(figsize=(15, 7))
sns.countplot(x='Category', data=df1, hue='Category', legend=False, palette='viridis')
plt.title('Distribution of App Categories')
plt.xticks(rotation=90)
plt.show()
```



The 'FAMILY' and 'GAME' categories have the highest number of apps.

```
#Which top 10 Categories have the highest number of apps?
```

```
top_10_categories = df1['Category'].value_counts().head(10)
display(top_10_categories)
```

Category	count
FAMILY	1511
GAME	832
TOOLS	625
PERSONALIZATION	274
LIFESTYLE	269
MEDICAL	266
FINANCE	258
PRODUCTIVITY	223
BUSINESS	222
SPORTS	221

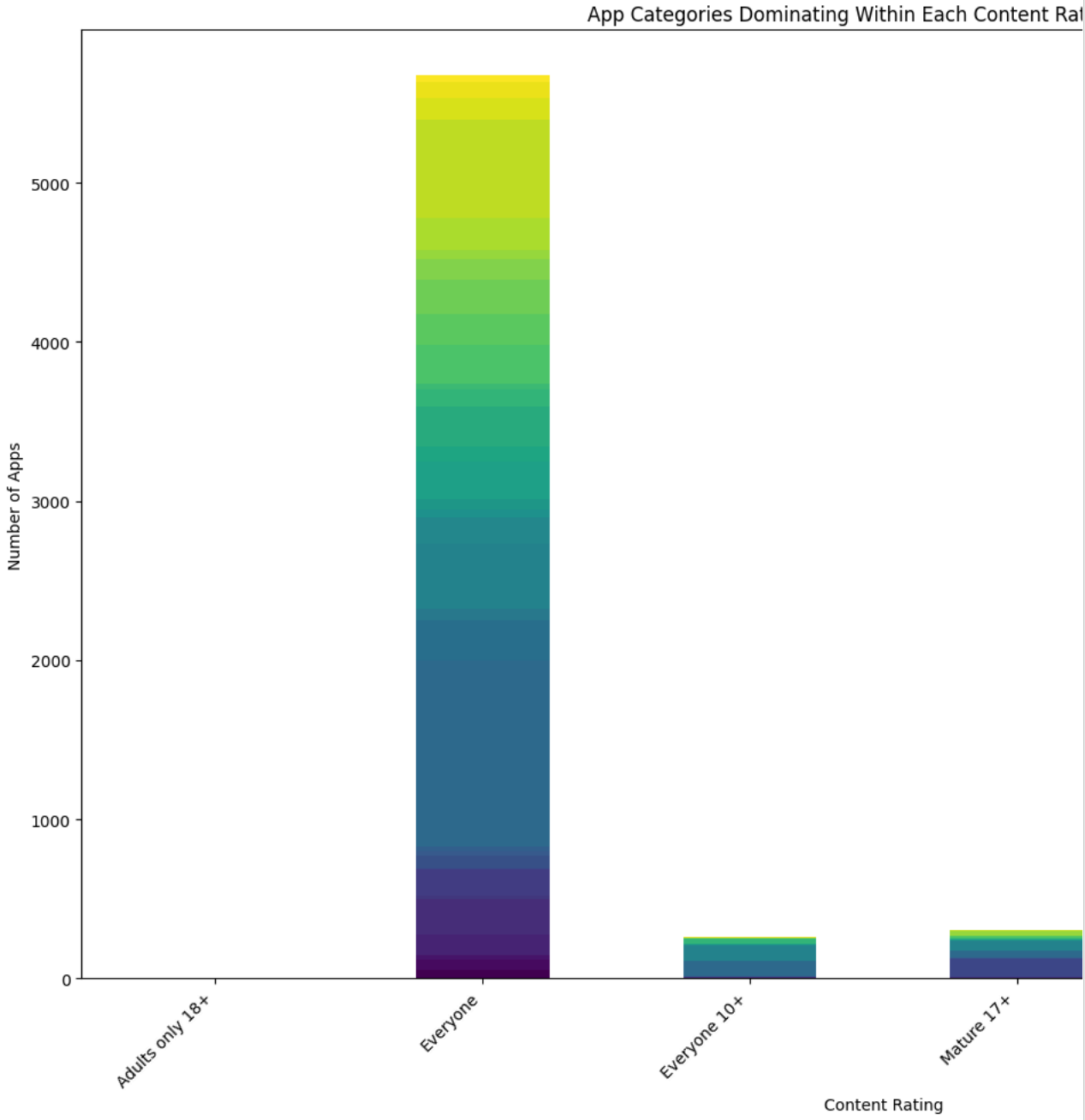
dtype: int64

The top 10 app categories are led by 'FAMILY' and 'GAME', indicating their dominance in the dataset.

```
#Which Categories dominate within each Content Rating group?
```

```
category_content_rating = df1.groupby(['Content Rating', 'Category']).size().unstack(fill_value=0)

# Plotting
plt.figure(figsize=(18, 10))
category_content_rating.plot(kind='bar', stacked=True, figsize=(18,10), cmap='viridis')
plt.title('App Categories Dominating Within Each Content Rating Group')
plt.xlabel('Content Rating')
plt.ylabel('Number of Apps')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Category', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

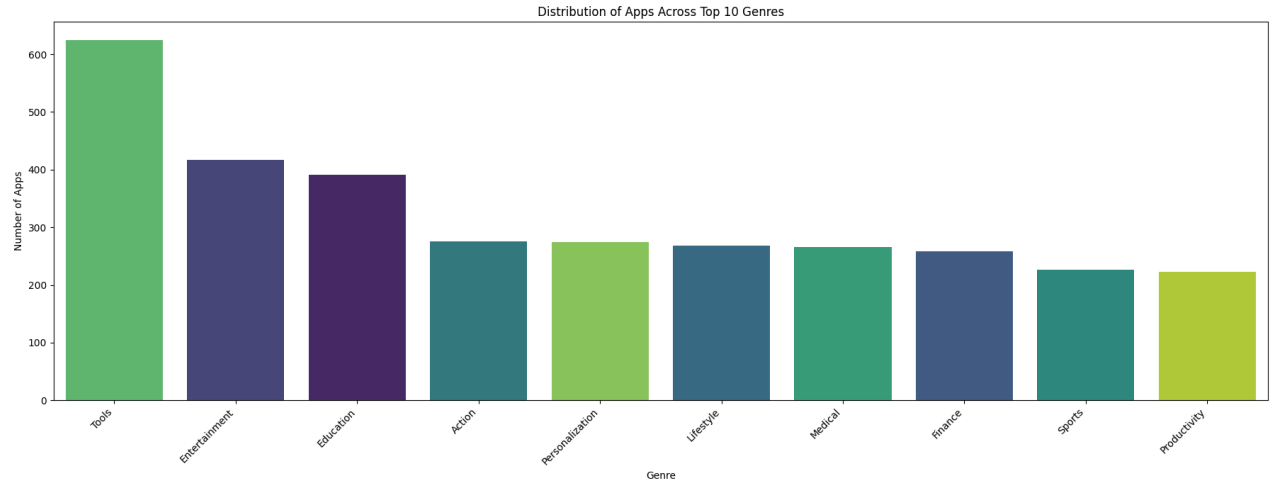
```
<Figure size 1800x1000 with 0 Axes>
```



The 'Everyone' content rating primarily features 'FAMILY' and 'GAME' apps, while 'Mature 17+' and 'Adults only 18+' are dominated by 'DATING' and 'COMMUNICATION' apps.

```
#How are apps distributed across different Genres, Top 10?
```

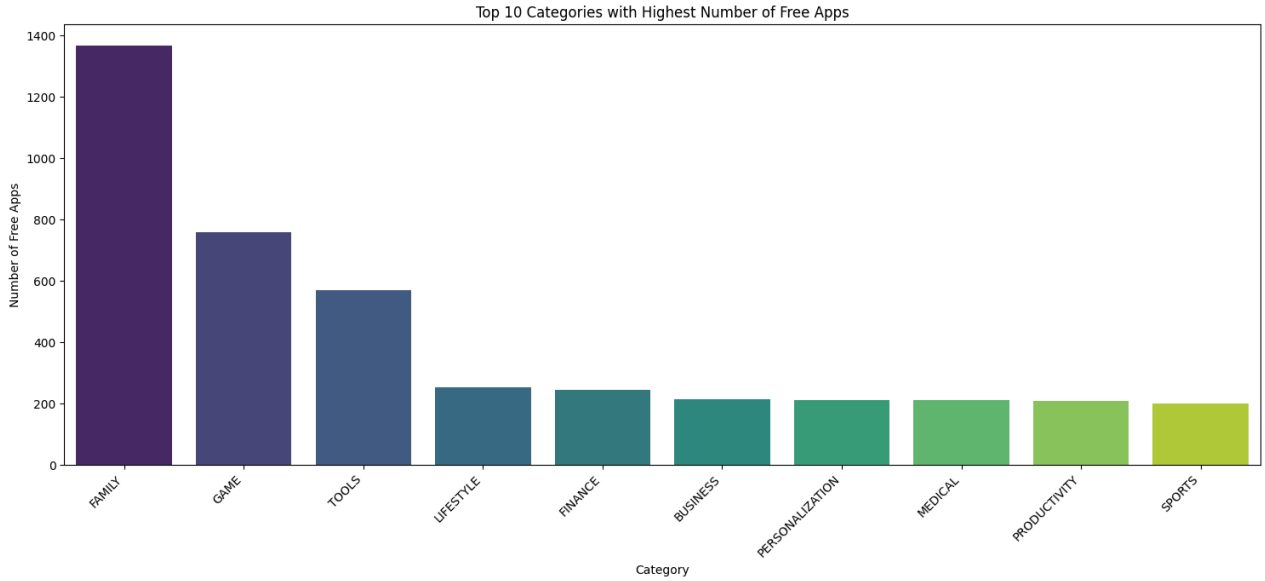
```
top_10_genres = df1['Genres'].value_counts().head(10).index
df_top10_genres = df1[df1['Genres'].isin(top_10_genres)]

plt.figure(figsize=(18, 7))
sns.countplot(x='Genres', data=df_top10_genres, hue='Genres', legend=False, palette='viridis', order=top_10_genres)
plt.title('Distribution of Apps Across Top 10 Genres')
plt.xlabel('Genre')
plt.ylabel('Number of Apps')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



Among the top 10 genres, 'Tools' is the most popular, followed closely by 'Entertainment' and 'Education', highlighting their significant presence in the app market.

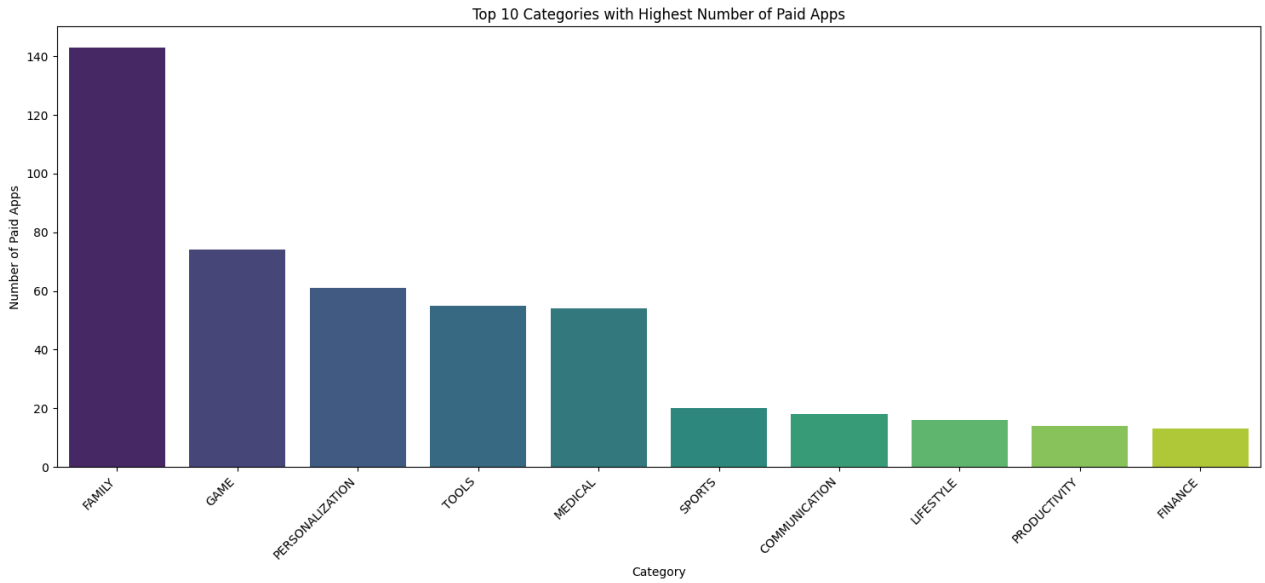
```
#Which Categories have the highest number of Paid apps?

#Which Categories have the highest number of Free apps?
```

```
free_apps = df1[df1['Type'] == 'Free']
top_free_categories = free_apps['Category'].value_counts().head(10)

plt.figure(figsize=(15, 7))
sns.barplot(x=top_free_categories.index, y=top_free_categories.values, hue=top_free_categories.index, legend=False, palette
plt.title('Top 10 Categories with Highest Number of Free Apps')
plt.xlabel('Category')
plt.ylabel('Number of Free Apps')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



The 'FAMILY', 'GAME', and 'TOOLS' categories dominate the landscape of free applications, much like their prominence in paid apps.

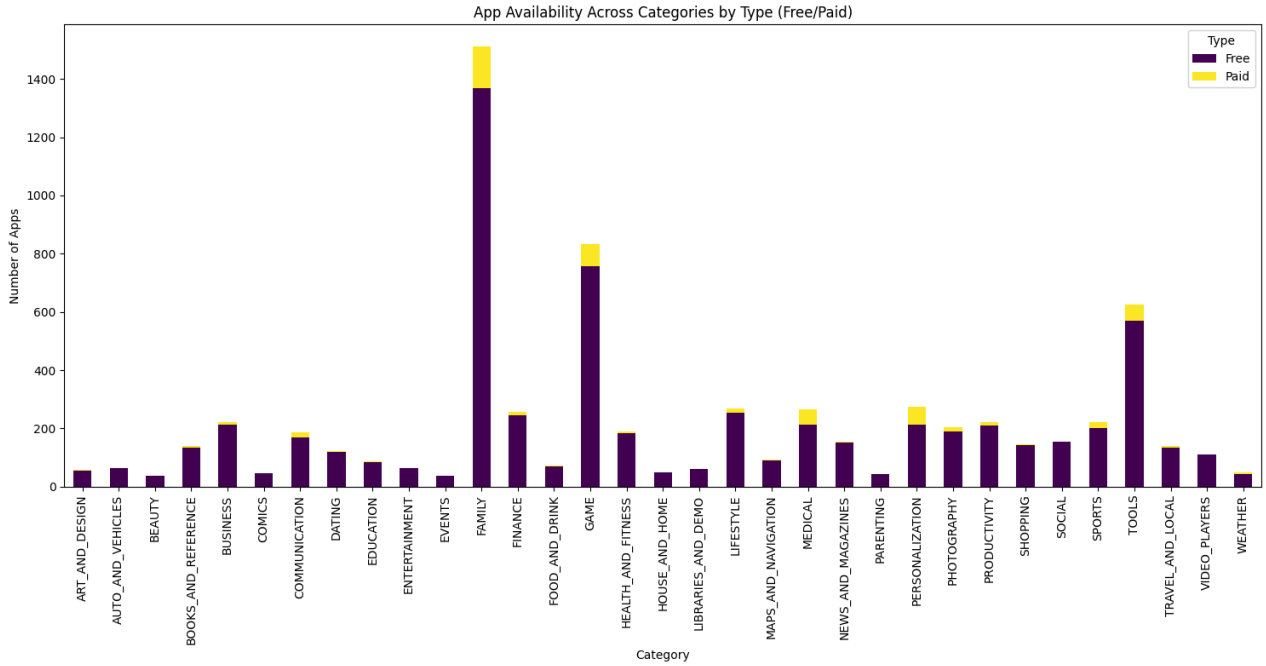
```
paid_apps = df1[df1['Type'] == 'Paid']
top_paid_categories = paid_apps['Category'].value_counts().head(10)

plt.figure(figsize=(15, 7))
sns.barplot(x=top_paid_categories.index, y=top_paid_categories.values, hue=top_paid_categories.index, legend=False, palette
plt.title('Top 10 Categories with Highest Number of Paid Apps')
plt.xlabel('Category')
plt.ylabel('Number of Paid Apps')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



The 'FAMILY', 'MEDICAL', and 'GAME' categories have the highest number of paid applications.

```
#How does app availability vary across Categories and Type (Free/Paid)?
```

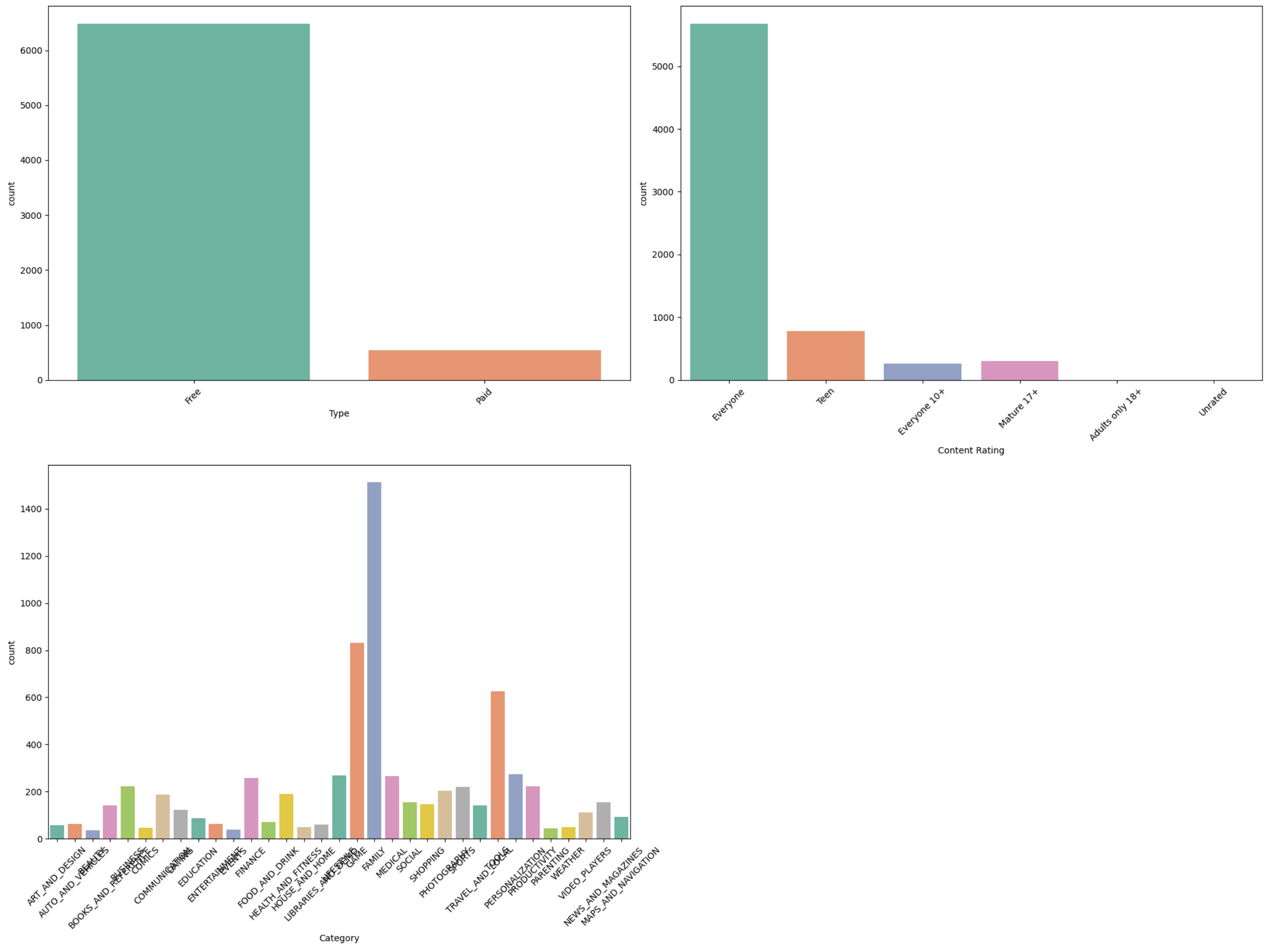
```
category_type_counts = df1.groupby(['Category', 'Type']).size().unstack(fill_value=0)

category_type_counts.plot(kind='bar', stacked=True, figsize=(15, 8), cmap='viridis')
plt.title('App Availability Across Categories by Type (Free/Paid)')
plt.xlabel('Category')
plt.ylabel('Number of Apps')
plt.xticks(rotation=90)
plt.legend(title='Type')
plt.tight_layout()
plt.show()
```



Across most categories, free applications significantly outnumber paid applications, with 'FAMILY' and 'GAME' categories having the highest counts for both types.

```
plt.figure(figsize=(20, 15))
Category=['Type','Content Rating','Category']
for i in range(0,len(Category)):
  plt.subplot(2,2,i+1)
  sns.countplot(x=df1[Category[i]],palette="Set2")
  plt.xlabel(Category[i])
  plt.xticks(rotation=45)
  plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-1799554324.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

  sns.countplot(x=df1[Category[i]],palette="Set2")
/tmp/ipython-input-1799554324.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

  sns.countplot(x=df1[Category[i]],palette="Set2")
/tmp/ipython-input-1799554324.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

  sns.countplot(x=df1[Category[i]],palette="Set2")
```




```
#which category has highest installation
```

```
df1.columns
```

```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Current Ver', 'Android Ver',
       'day', 'month', 'year'],
      dtype='object')
```

```
df1.groupby('Category')['Installs'].sum().sort_values(ascending=False).reset_index()
```

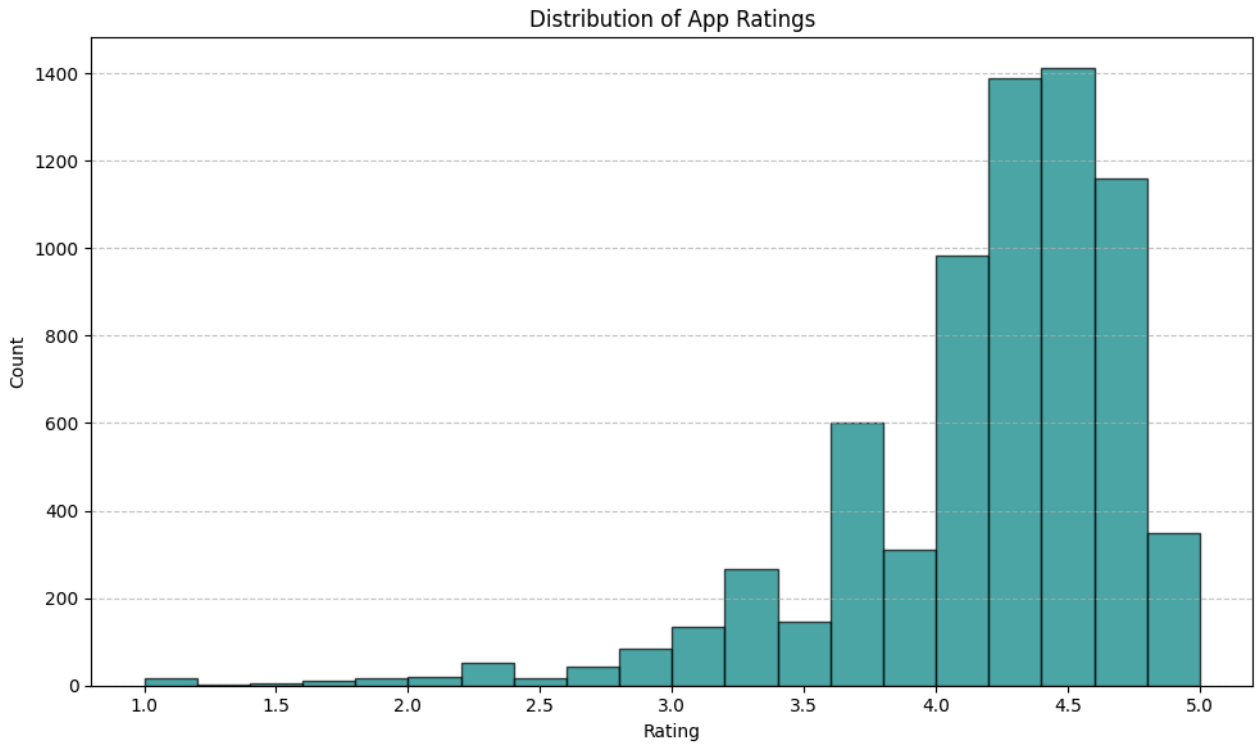
	Category	Installs
0	GAME	11532352717
1	FAMILY	3552661580
2	TOOLS	2879553500
3	COMMUNICATION	1817915530
4	PHOTOGRAPHY	1493893130
5	PRODUCTIVITY	1296302080
6	NEWS_AND_MAGAZINES	1190900550
7	PERSONALIZATION	895131930
8	VIDEO_PLAYERS	866662200
9	SPORTS	806311465
10	HEALTH_AND_FITNESS	756456220
11	SHOPPING	710731540
12	ENTERTAINMENT	637960000
13	SOCIAL	558240475
14	LIFESTYLE	404519120
15	BUSINESS	386282920
16	FINANCE	244587300
17	TRAVEL_AND_LOCAL	228638300
18	MAPS_AND_NAVIGATION	174015560
19	EDUCATION	157202000
20	FOOD_AND_DRINK	136467750
21	WEATHER	129296500
22	BOOKS_AND_REFERENCE	114784155
23	ART_AND_DESIGN	99228100
24	DATING	84592410
25	HOUSE_AND_HOME	51482000
26	LIBRARIES_AND_DEMO	49983000
27	AUTO_AND_VEHICLES	43769800
28	MEDICAL	31550176
29	PARENTING	23566010
30	COMICS	17431100
31	BEAUTY	13416200
32	EVENTS	10648400

Games App has highest Installation

```
#What is the distribution of app Ratings across the dataset?
```

```
plt.figure(figsize=(10, 6))
plt.hist(df1['Rating'], bins=20, color='teal', edgecolor='black', alpha=0.7)
plt.xlabel('Rating')
plt.ylabel('Count')
plt.title('Distribution of App Ratings')
plt.grid(axis='y', linestyle='--', alpha=0.7)
```

```
plt.tight_layout()
plt.show()
```



Most apps have ratings concentrated in the higher range (above 4), indicating a generally positive user experience.

```
#Which Categories have the highest average Ratings?
```

```
average_ratings_by_category = df1.groupby('Category')['Rating'].mean().sort_values(ascending=False).head(10)
display(average_ratings_by_category)
```

Category	Rating
EVENTS	4.478947
ART_AND_DESIGN	4.381034
EDUCATION	4.373864
PARENTING	4.347727
PERSONALIZATION	4.324453
BOOKS_AND_REFERENCE	4.322695
BEAUTY	4.291892
SOCIAL	4.257692
WEATHER	4.242000
GAME	4.235697

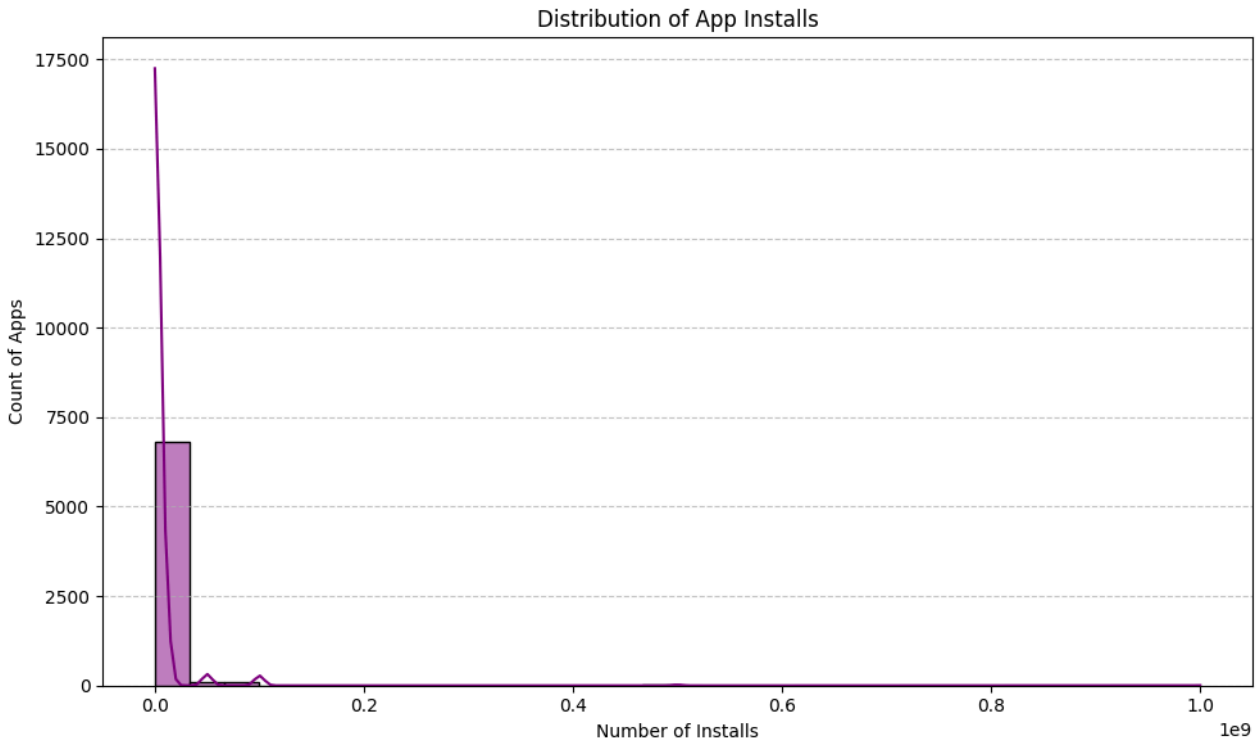
dtype: float64

The 'EVENTS', 'EDUCATION', and 'ART_AND_DESIGN' categories boast the highest average app ratings, suggesting high user satisfaction in these areas.

```
#How are Installs distributed across all apps?
```

```
Start coding or generate with AI.
```

```
plt.figure(figsize=(10, 6))
sns.histplot(df1['Installs'], bins=30, kde=True, color='purple', edgecolor='black')
plt.title('Distribution of App Installs')
plt.xlabel('Number of Installs')
plt.ylabel('Count of Apps')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



The distribution of app installs is heavily skewed towards lower values, indicating that most apps have fewer installations.

```
#Which Categories have the highest total number of Installs?
```

```
total_installs_by_category = df1.groupby('Category')['Installs'].sum().sort_values(ascending=False).head(10)
display(total_installs_by_category)
```

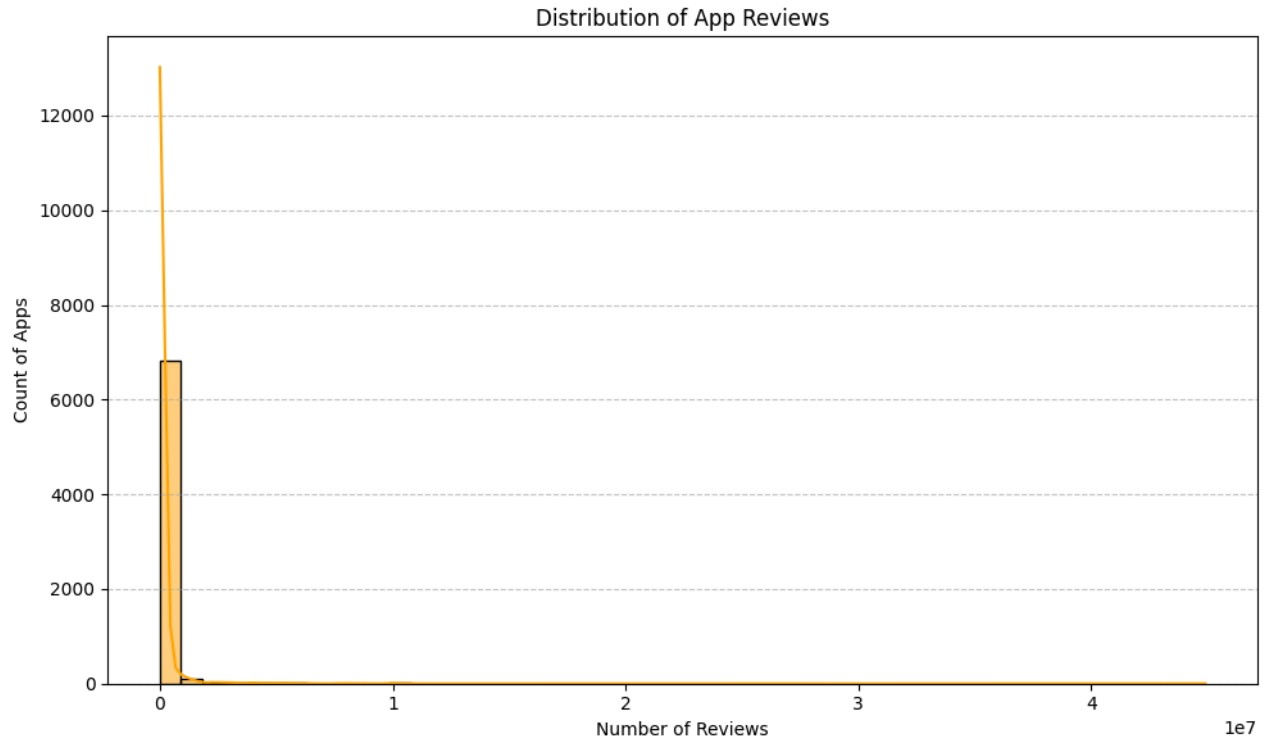
Category	Installs
GAME	11532352717
FAMILY	3552661580
TOOLS	2879553500
COMMUNICATION	1817915530
PHOTOGRAPHY	1493893130
PRODUCTIVITY	1296302080
NEWS_AND_MAGAZINES	1190900550
PERSONALIZATION	895131930
VIDEO_PLAYERS	866662200
SPORTS	806311465

dtype: int64

The 'GAME' category has by far the highest total number of installs, significantly surpassing other categories like 'FAMILY' and 'TOOLS'.

```
#What is the distribution of Reviews among apps?
```

```
plt.figure(figsize=(10, 6))
sns.histplot(df1['Reviews'], bins=50, kde=True, color='orange', edgecolor='black')
plt.title('Distribution of App Reviews')
plt.xlabel('Number of Reviews')
plt.ylabel('Count of Apps')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



The distribution of app reviews is heavily skewed towards lower values, indicating that most apps have a relatively small number of reviews.