google-play-store-dataset

February 6, 2025

0.0.1 EDA And Feature Engineering Of Google Play Store Dataset

1) Problem statement. Today, 1.85 million different apps are available for users to download. Android users have even more from which to choose, with 2.56 million available through the Google Play Store. These apps have come to play a huge role in the way we live our lives today. Our Objective is to find the Most Popular Category, find the App with largest number of installs , the App with largest size etc.
2) Data Collection.

The data consists of 15 column and 10841 rows.

0.0.2 Steps We Are Going to Follow

1. Data Clearning
2. Exploratory Data Analysis
3. Feature Engineering

[3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

warnings.filterwarnings("ignore")

%matplotlib inline
```

[5]:
```
df=pd.read_csv('C:/Users/Dell/Downloads/Data Science/googleplaystore.csv')
df.head()
```

[5]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9						
2	U Launcher Lite - FREE Live Cool Themes, Hide …	ART_AND_DESIGN	4.7						
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5						
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3						

```
1     967   14M     500,000+  Free     0       Everyone
2   87510  8.7M   5,000,000+  Free     0       Everyone
3  215644   25M  50,000,000+  Free     0           Teen
4     967  2.8M     100,000+  Free     0       Everyone

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

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

[7]:
```
df.shape
```

[7]:
```
(10841, 13)
```

[9]:
```
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
```

[11]:
```
##summary of the dataset
df.describe()
```

[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
```

[13]:
```
## Missing Values
df.isnull().sum()
```

[13]:
```
App                0
Category           0
Rating          1474
Reviews            0
Size               0
Installs           0
Type               1
Price              0
Content Rating     1
Genres             0
Last Updated       0
Current Ver        8
Android Ver        3
dtype: int64
```

0.1 Insights and Observation

The dataset has missing values

0.2 Data Cleaning

[17]:
```
## check all the unique reviews
df['Reviews'].unique()
```

[17]:
```
array(['159', '967', '87510', …, '603', '1195', '398307'], dtype=object)
```

[19]:
```
## Numeric format Reviews are in string format and finding the total reviews
df['Reviews'].str.isnumeric().sum()
```

[19]:
```
10840
```

[21]:
```
## 1 Review left that is not in string numeric format
df[~df['Reviews'].str.isnumeric()]
```

[21]:
```
                                            App Category  Rating Reviews  \
10472  Life Made WI-Fi Touchscreen Photo Frame      1.9    19.0    3.0M

         Size Installs Type     Price Content Rating             Genres  \
10472  1,000+     Free    0  Everyone            NaN  February 11, 2018

      Last Updated Current Ver Android Ver
10472       1.0.19  4.0 and up         NaN
```

[23]:
```
df_copy = df.copy()
```

[25]:
```
df_copy =df_copy.drop(df_copy.index[10472])
```

[27]:
```
## Index with 3.0M review is deleted
df_copy[~df_copy['Reviews'].str.isnumeric()]
```

[27]:
```
Empty DataFrame
Columns: [App, Category, Rating, Reviews, Size, Installs, Type, Price, Content
Rating, Genres, Last Updated, Current Ver, Android Ver]
Index: []
```

[29]:
```
## Convert Review Datatype to int
df_copy['Reviews'] = df_copy['Reviews'].astype(int)
```

[31]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 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
```

[33]:
```
## Converting Size Datatype to float
df_copy['Size'].unique()
```

[33]:
```
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', '470k',
       '226k', '240k', '89k', '234k', '257k', '861k', '467k', '157k',
       '44k', '676k', '67k', '552k', '885k', '1020k', '582k', '619k'],
      dtype=object)
```

[38]:
```
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)
```

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

[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,
       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])
```

[124]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10839 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   App             10839 non-null  object
 1   Category        10839 non-null  object
 2   Rating          9365 non-null   float64
 3   Reviews         10839 non-null  int32
 4   Size            9144 non-null   float64
 5   Installs        10839 non-null  object
 6   Type            10838 non-null  object
 7   Price           10839 non-null  object
 8   Content Rating  10839 non-null  object
 9   Genres          10839 non-null  object
 10  Last Updated    10839 non-null  object
 11  Current Ver     10831 non-null  object
 12  Android Ver     10837 non-null  object
dtypes: float64(2), int32(1), object(10)
memory usage: 1.1+ MB
```

[42]:
```
df_copy['Size']
```

[42]:
```
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
```

[44]:
```
df['Installs'].unique()
```

[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', 'Free'], dtype=object)
```

[46]:
```
df['Price'].unique()
```

[46]:
```
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', 'Everyone', '$1.20', '$1.04'], dtype=object)
```

[48]:
```
chars_to_remove = ['+','$',',']
cols_to_clean = ['Installs','Price']

for item in chars_to_remove:
    for cols in cols_to_clean:
        df_copy[cols] = df_copy[cols].str.replace(item, '')
```

[50]:
```
df_copy['Installs'].unique()
```

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

[52]:
```
df_copy['Price'].unique()
```

[52]:
```
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)
```

[54]:
```
df_copy['Installs']= df_copy['Installs'].astype(int)
df_copy['Price'] = df_copy['Price'].astype(float)
```

[56]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 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: 1.1+ MB
```

[60]:
```
import pandas as pd

# Convert to datetime
df_copy['Last Updated'] = pd.to_datetime(df_copy['Last Updated'],␣
 ↪errors='coerce')

# Extract day, month, and year
df_copy['Day'] = df_copy['Last Updated'].dt.day
df_copy['Month'] = df_copy['Last Updated'].dt.month
df_copy['Year'] = df_copy['Last Updated'].dt.year
```

[64]:
```
# df_copy.drop(columns=['Last_Updated'], inplace=True)
```

[66]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10840 entries, 0 to 10840
```

```
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  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  Current Ver     10832 non-null  object
 11  Android Ver     10838 non-null  object
 12  Day             10840 non-null  int32
 13  Month           10840 non-null  int32
 14  Year            10840 non-null  int32
dtypes: float64(3), int32(5), object(7)
memory usage: 1.1+ MB
```

[68]:
```
df_copy.head()
```

[68]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Current Ver	Android Ver	Day	Month	Year
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	1.0.0	4.0.3 and up	7	1	2018
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2.0.0	4.0.3 and up	15	1	2018
2	U Launcher Lite – FREE Live Cool Themes, Hide …	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	Art & Design	1.2.4	4.0.3 and up	1	8	
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	Varies with device	4.2 and up	8	6	
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Art & Design;Creativity	1.1	4.4 and up	20	6	

```
2  2018
3  2018
4  2018
```

[70]:
```
df_copy.to_csv('Data/google_cleaned.csv')
```

1 EDA

[74]:
```
df_copy.head()
```

[74]:
```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite - FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7
3                              Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4               Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

   Reviews     Size  Installs  Type  Price Content Rating  \
0      159  19000.0     10000  Free    0.0       Everyone
1      967  14000.0    500000  Free    0.0       Everyone
2    87510      8.7   5000000  Free    0.0       Everyone
3   215644  25000.0  50000000  Free    0.0           Teen
4      967      2.8    100000  Free    0.0       Everyone

                      Genres         Current Ver   Android Ver  Day  Month  \
0               Art & Design               1.0.0  4.0.3 and up    7      1
1  Art & Design;Pretend Play               2.0.0  4.0.3 and up   15      1
2               Art & Design               1.2.4  4.0.3 and up    1      8
3               Art & Design  Varies with device    4.2 and up    8      6
4    Art & Design;Creativity                 1.1    4.4 and up   20      6

   Year
0  2018
1  2018
2  2018
3  2018
4  2018
```

[83]:
```
df_copy[df_copy.duplicated('App')].shape
```

[83]:
```
(1181, 15)
```

1.1 Observation

The Dataset has duplicate records

[85]:
```
df_copy = df_copy.drop_duplicates(subset = ['App'], keep='first')
```

[87]:
```
df_copy.shape
```

[87]:
```
(9659, 15)
```

2 Explore Data

[98]:
```
numeric_features = [feature for feature in df_copy.columns if df_copy[feature].
 ↪dtype != 'O']
categorical_features = [feature for feature in df_copy.columns if␣
 ↪df_copy[feature].dtype == 'O']

# print column
print('We have {} numerical features : {}'.format(len(numeric_features),␣
 ↪numeric_features))
print('\nWe have {} categorical features : {}'.
 ↪format(len(categorical_features), categorical_features))
```

```
We have 8 numerical features : ['Rating', 'Reviews', 'Size', 'Installs',
'Price', 'Day', 'Month', 'Year']

We have 7 categorical features : ['App', 'Category', 'Type', 'Content Rating',
'Genres', 'Current Ver', 'Android Ver']
```

2.1 Feature Information

1. App :- Name of the App
2. Category :- Category under which the App falls.
3. Rating :- Application's rating on playstore
4. Reviews :- Number of reviews of the App.
5. Size :- Size of the App.
6. Install :- Number of Installs of the App
7. Type :- If the App is free/paid
8. Price :- Price of the app (0 if it is Free)
9. Content Rating :- Appropiate Target Audience of the App.
10. Genres:- Genre under which the App falls.
11. Last Updated :- Date when the App was last updated
12. Current Ver :- Current Version of the Application
13. Android Ver :- Minimum Android Version required to run the App

[104]:
```
## Proportion of count data on categorical columns :
for col in categorical_features:
    print(df[col].value_counts(normalize = True)*100)
    print("----------------------------------------")
```

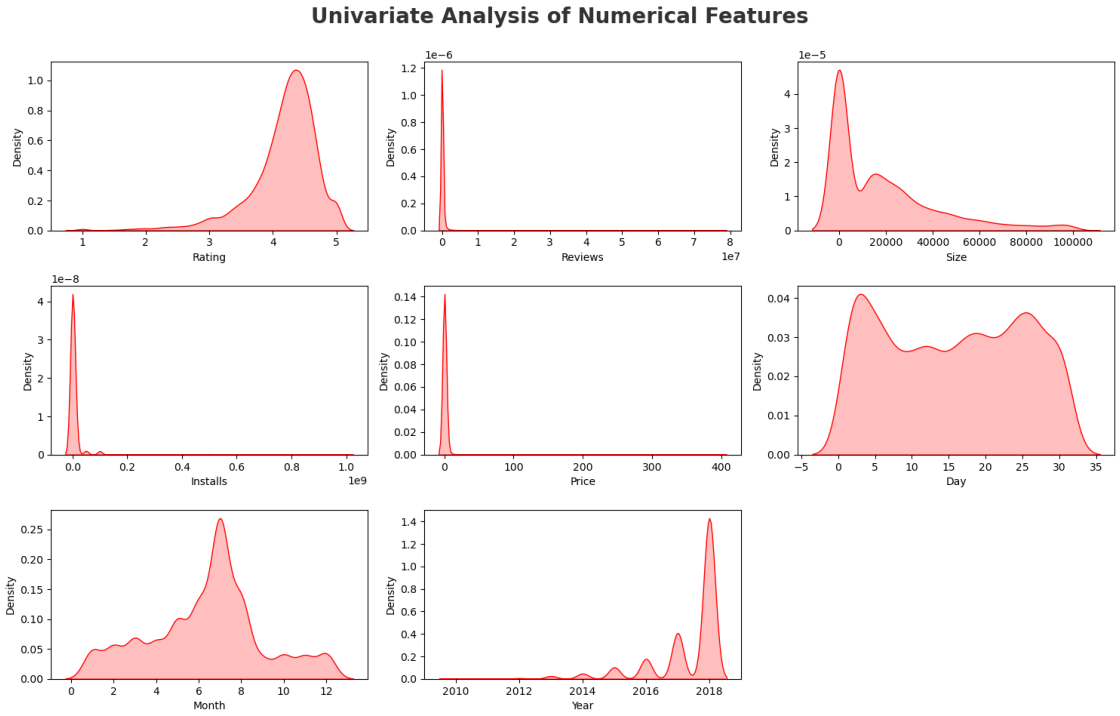
```
App
ROBLOX                                                 0.083018
CBS Sports App - Scores, News, Stats & Watch Live      0.073794
ESPN                                                   0.064570
Duolingo: Learn Languages Free                         0.064570
Candy Crush Saga                                       0.064570
                                                         …
Meet U - Get Friends for Snapchat, Kik & Instagram     0.009224
U-Report                                               0.009224
U of I Community Credit Union                          0.009224
Waiting For U Launcher Theme                           0.009224
iHoroscope - 2018 Daily Horoscope & Astrology          0.009224
Name: proportion, Length: 9660, dtype: float64
----------------------------------------
Category
FAMILY                18.190204
GAME                  10.552532
TOOLS                  7.776035
MEDICAL                4.270824
BUSINESS               4.243151
PRODUCTIVITY           3.911078
PERSONALIZATION        3.615903
COMMUNICATION          3.569781
SPORTS                 3.542109
LIFESTYLE              3.523660
FINANCE                3.376072
HEALTH_AND_FITNESS     3.145466
PHOTOGRAPHY            3.090121
SOCIAL                 2.721151
NEWS_AND_MAGAZINES     2.610460
SHOPPING               2.398303
TRAVEL_AND_LOCAL       2.379854
DATING                 2.158472
BOOKS_AND_REFERENCE    2.130800
VIDEO_PLAYERS          1.614242
EDUCATION              1.438982
ENTERTAINMENT          1.374412
MAPS_AND_NAVIGATION    1.263721
FOOD_AND_DRINK         1.171479
HOUSE_AND_HOME         0.811733
LIBRARIES_AND_DEMO     0.784061
AUTO_AND_VEHICLES      0.784061
WEATHER                0.756388
ART_AND_DESIGN         0.599576
EVENTS                 0.590351
PARENTING              0.553454
COMICS                 0.553454
BEAUTY                 0.488885
```

```
1.9                     0.009224
Name: proportion, dtype: float64
----------------------------------------
Type
Free    92.610701
Paid     7.380074
0        0.009225
Name: proportion, dtype: float64
----------------------------------------
Content Rating
Everyone           80.387454
Teen               11.143911
Mature 17+          4.603321
Everyone 10+        3.819188
Adults only 18+     0.027675
Unrated             0.018450
Name: proportion, dtype: float64
----------------------------------------
Genres
Tools                     7.766811
Entertainment             5.746702
Education                 5.064108
Medical                   4.270824
Business                  4.243151
                             …
Arcade;Pretend Play       0.009224
Card;Brain Games          0.009224
Lifestyle;Pretend Play    0.009224
Comics;Creativity         0.009224
Strategy;Creativity       0.009224
Name: proportion, Length: 120, dtype: float64
----------------------------------------
Current Ver
Varies with device    13.468107
1                      7.772547
1.1                    2.547771
1.2                    1.707745
2                      1.523124
                          …
5.1.0 free             0.009231
04.08.00               0.009231
2.10.06                0.009231
18.0.2                 0.009231
2.0.148.0              0.009231
Name: proportion, Length: 2784, dtype: float64
----------------------------------------
Android Ver
4.1 and up            22.614874
```

```
4.0.3 and up          13.849419
4.0 and up            12.686843
Varies with device    12.566894
4.4 and up             9.042259
2.3 and up             6.015870
5.0 and up             5.545304
4.2 and up             3.635357
2.3.3 and up           2.592729
2.2 and up             2.251338
4.3 and up             2.242111
3.0 and up             2.223658
2.1 and up             1.236390
1.6 and up             1.070308
6.0 and up             0.553608
7.0 and up             0.387525
3.2 and up             0.332165
2.0 and up             0.295257
5.1 and up             0.221443
1.5 and up             0.184536
4.4W and up            0.110722
3.1 and up             0.092268
2.0.1 and up           0.064588
8.0 and up             0.055361
7.1 and up             0.027680
4.0.3 - 7.1.1          0.018454
5.0 - 8.0              0.018454
1.0 and up             0.018454
7.0 - 7.1.1            0.009227
4.1 - 7.1.1            0.009227
5.0 - 6.0              0.009227
2.2 - 7.1.1            0.009227
5.0 - 7.1.1            0.009227
Name: proportion, dtype: float64
----------------------------------------
```

[140]:
```
## Proportion of count data on numerical columns
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize = (15,15))
plt.suptitle('Univariate Analysis of Numerical Features',fontsize = 20,␣
 ↪fontweight = 'bold', alpha=0.8, y=1)

for i in range (0,len(numeric_features)):
    plt.subplot(5,3,i+1)
    sns.kdeplot(x=df_copy[numeric_features[i]], shade=True, color ='r')
```

```
    plt.xlabel(numeric_features[i])
    plt.tight_layout()
plt.show()
```



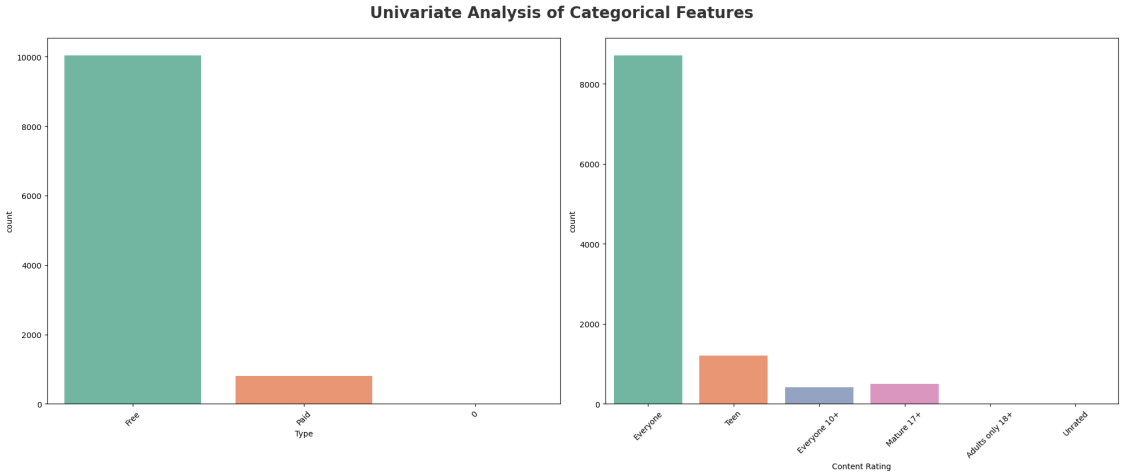
2.2 Observations

- Rating and Year is left skewed while Reviews, Install, Size and Price are right skewed

[190]:
```
# Categorical Columns
plt.figure(figsize=(20,15))
plt.suptitle('Univariate Analysis of Categorical Features',fontsize = 20,␣
 ↪fontweight = 'bold', alpha=0.8, y=1)
category = ['Type','Content Rating']
for i in range(0,len(category)):
    plt.subplot(2,2,i+1)
    sns.countplot(x=df[category[i]],palette='Set2')
    plt.xlabel(category[i])
    plt.xticks(rotation = 45)
    plt.tight_layout()
plt.show()
```

```
<Figure size 2000x1500 with 0 Axes>
```

```
<Figure size 2000x1500 with 0 Axes>
```

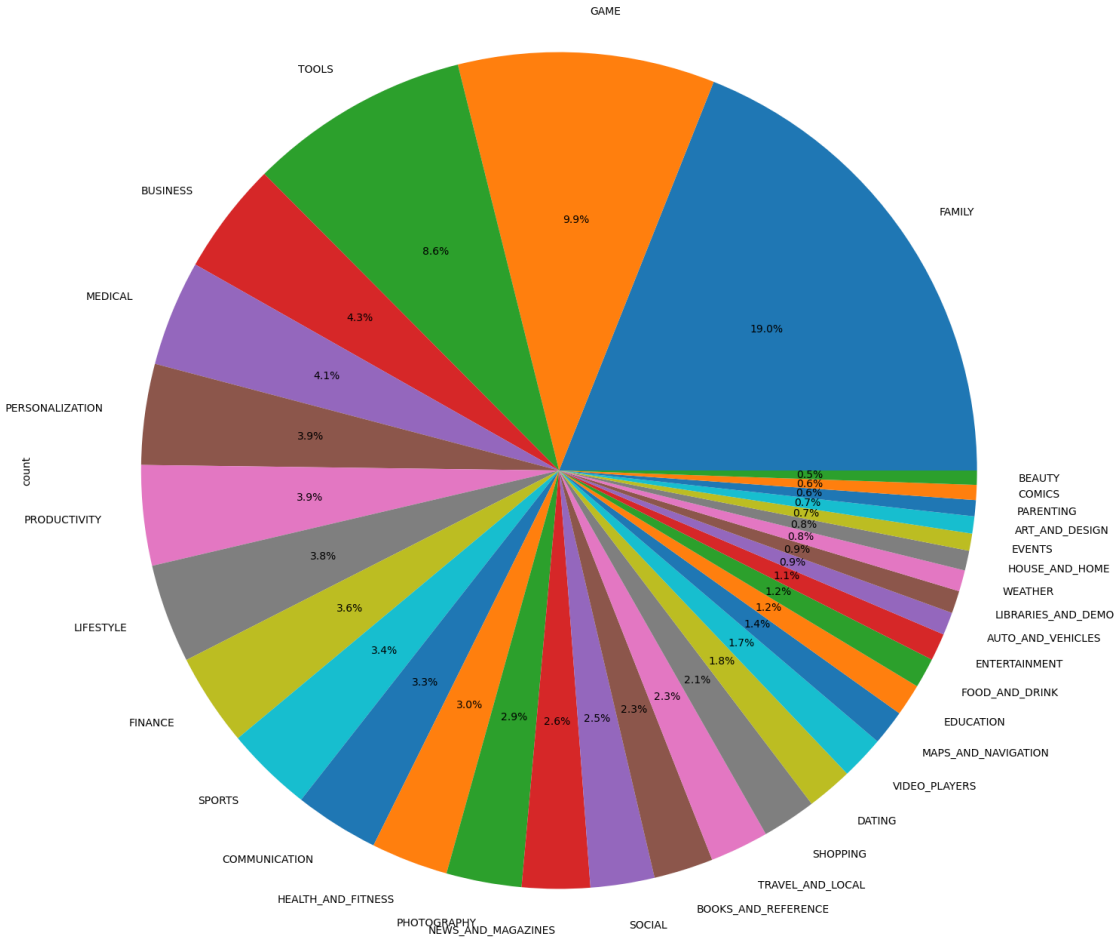


2.3 Which is the most popular app category?

[162]:
```
df_copy.head(2)
```

[162]:
```
                                                 App        Category  Rating  \
0  Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                             Coloring book moana  ART_AND_DESIGN     3.9

   Reviews     Size  Installs  Type  Price Content Rating  \
0      159  19000.0     10000  Free    0.0       Everyone
1      967  14000.0    500000  Free    0.0       Everyone

                     Genres Current Ver   Android Ver  Day  Month  Year
0              Art & Design       1.0.0  4.0.3 and up    7      1  2018
1  Art & Design;Pretend Play       2.0.0  4.0.3 and up   15      1  2018
```

[184]:
```
df_copy['Category'].value_counts().plot.pie(y=df['Category'],figsize = (20,18),␣
↪autopct='%1.1f%%')
plt.show()
```



2.4 Observations

1. There are more kinds of apps in playstore which are under category of family, games & tools
2. Beatuty,comics,arts and weather kinds of apps are very less in playstore

[219]:
```
## Top 10 App Categories
category = pd.DataFrame(df_copy['Category'].value_counts())          #Dataframe␣
 ↪of apps on the basis of category
category.rename(columns = {'Category':'Count'},inplace=True)
```

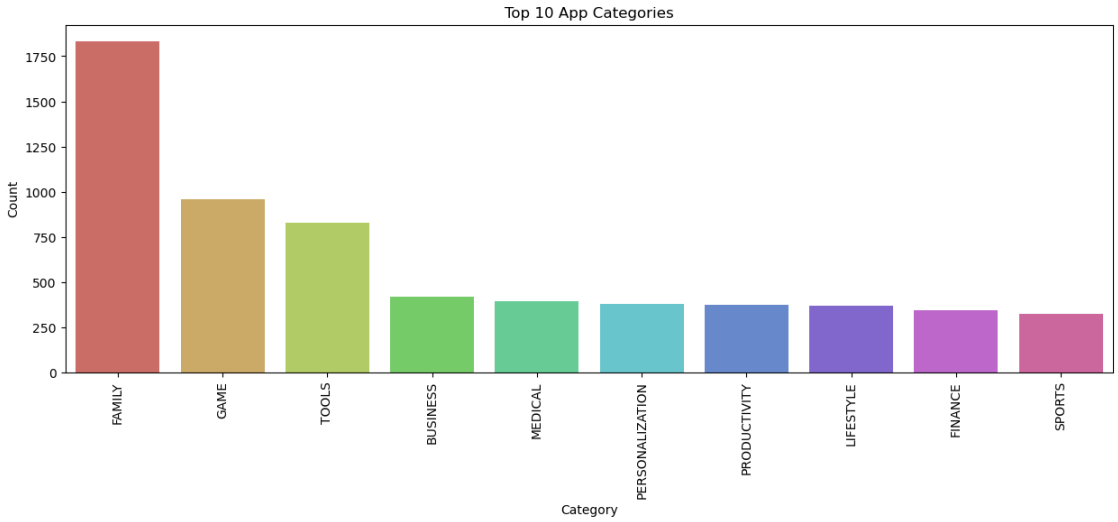
[221]:
```
category
```

[221]:

Category	count
FAMILY	1832

```
GAME                 959
TOOLS                827
BUSINESS             420
MEDICAL              395
PERSONALIZATION      376
PRODUCTIVITY         374
LIFESTYLE            369
FINANCE              345
SPORTS               325
COMMUNICATION        315
HEALTH_AND_FITNESS   288
PHOTOGRAPHY          281
NEWS_AND_MAGAZINES   254
SOCIAL               239
BOOKS_AND_REFERENCE  222
TRAVEL_AND_LOCAL     219
SHOPPING             202
DATING               171
VIDEO_PLAYERS        163
MAPS_AND_NAVIGATION  131
EDUCATION            119
FOOD_AND_DRINK       112
ENTERTAINMENT        102
AUTO_AND_VEHICLES     85
LIBRARIES_AND_DEMO    84
WEATHER               79
HOUSE_AND_HOME        74
EVENTS                64
ART_AND_DESIGN        64
PARENTING             60
COMICS                56
BEAUTY                53
```

[229]:
```
plt.figure(figsize=(15,5))

# Convert 'category' Series to DataFrame
category_df = category[:10].reset_index()  # Reset index to include category␣
 ↪names
category_df.columns = ['Category', 'Count']  # Rename columns

sns.barplot(x='Category', y='Count', data=category_df, palette='hls')

plt.title('Top 10 App Categories')
plt.xticks(rotation=90)
plt.show()
```



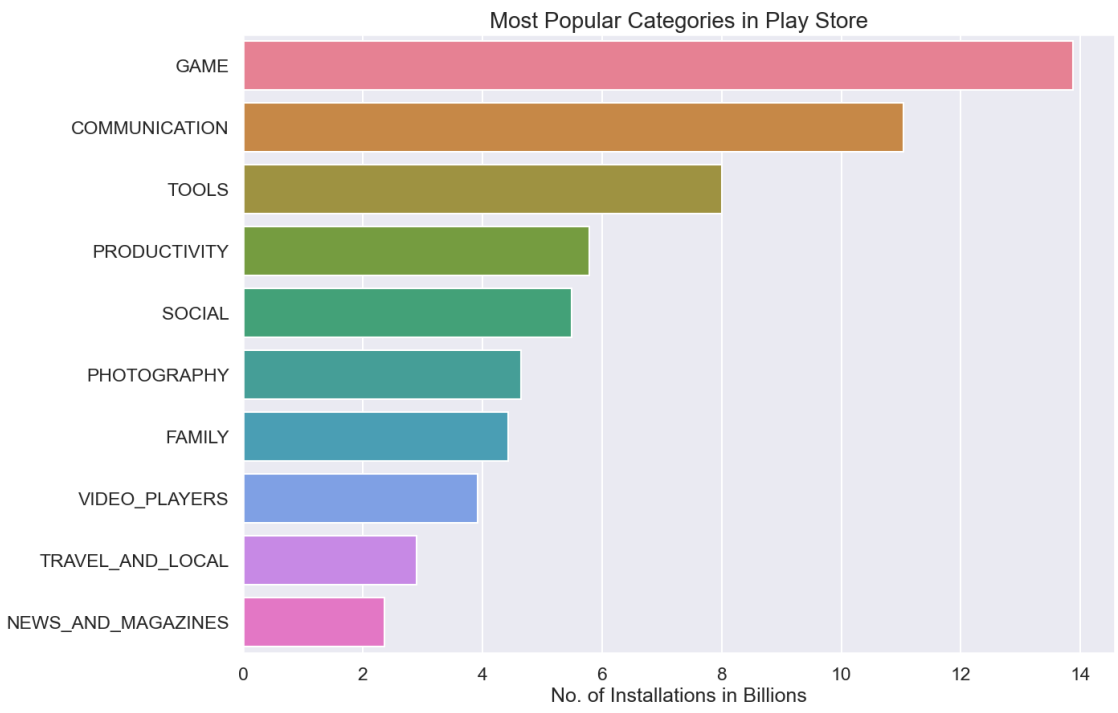
2.5 Insights

1. Family category has the most number of apps with 18% of apps belonging to it, followed by Games category which has 11% of the apps.
2. Least number of apps belong to the Beauty category with less than 1% of the total apps belonging to it.

2.6 Which Category has largest number of installations??

[243]:
```
import seaborn as sns
import matplotlib.pyplot as plt

df_cat_installs = df_copy.groupby(['Category'])['Installs'].sum().
 ↪sort_values(ascending=False).reset_index()
df_cat_installs['Installs'] = df_cat_installs['Installs'] / 1e9  # Convert to␣
 ↪billions
df2 = df_cat_installs.head(10)

plt.figure(figsize=(14,10))
sns.set_context("talk")
sns.set_style("darkgrid")

# Use different colors for each bar
ax = sns.barplot(x='Installs', y='Category', data=df2, palette=sns.
 ↪color_palette("husl", len(df2)))

ax.set_xlabel('No. of Installations in Billions')
ax.set_ylabel('')
ax.set_title("Most Popular Categories in Play Store", size=20)
```

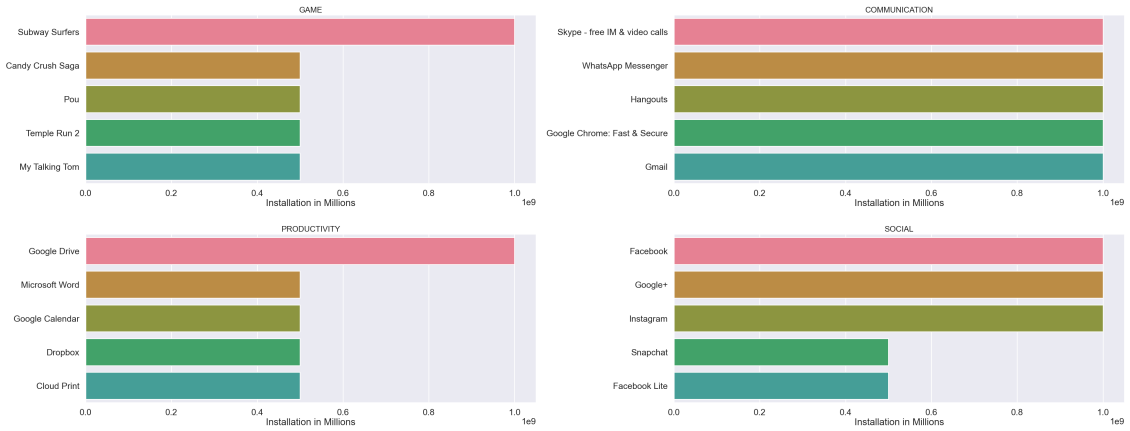
```
plt.show()
```



2.7 What are the top 5 most installed Apps in each popular categories?

[285]:
```
import seaborn as sns
import matplotlib.pyplot as plt

dfa = df_copy.groupby(['Category', 'App'])['Installs'].sum().reset_index()
dfa = dfa.sort_values('Installs', ascending=False)

apps = ['GAME', 'COMMUNICATION', 'PRODUCTIVITY', 'SOCIAL']
sns.set_context("poster")
sns.set_style("darkgrid")

plt.figure(figsize=(40, 30))

# Define different color palettes
palettes = ['Blues', 'Reds', 'Greens', 'Purples']  # You can add more color maps

for i, app in enumerate(apps):
    df2 = dfa[dfa.Category == app]
    df3 = df2.head(5)
```

```
    plt.subplot(4, 2, i + 1)
    sns.barplot(data=df3, x='Installs', y='App', palette=sns.
 ↪color_palette('husl',8))

    plt.xlabel('Installation in Millions')
    plt.ylabel('')
    plt.title(app, size=20)

plt.tight_layout()
plt.subplots_adjust(hspace=0.3)
plt.show()
```



2.8 Insights

- Most popular game is Subway Surfers.
- Most popular communication app is Hangouts.
- Most popular productivity app is Google Drive.
- Most popular social app is Instagram.

2.9 How many apps are there on Google Play Store which get 5 ratings??

[291]:
```
rating = df_copy.groupby(['Category','Installs', 'App'])['Rating'].sum().
 ↪sort_values(ascending = False).reset_index()

toprating_apps = rating[rating.Rating == 5.0]
print("Number of 5 rated apps",toprating_apps.shape[0])
toprating_apps.head(10)
```

```
Number of 5 rated apps 271
```

[291]:

	Category	Installs	App	Rating
0	FAMILY	1000	CS & IT Interview Questions	5.0

```
1   DATING       100                          Online Girls Chat Group     5.0
2   FAMILY        10                                    Chronolink DX     5.0
3   DATING       500                            Spine- The dating app     5.0
4  MEDICAL         5                                Clinic Doctor EHr     5.0
5  MEDICAL         5  Anatomy & Physiology Vocabulary Exam Review App     5.0
6  MEDICAL         1                               KBA-EZ Health Guide     5.0
7   FAMILY        10                                     DN Employee     5.0
8   FAMILY      1000                                   Safe Santa Fe     5.0
9   DATING       100                 Speeding Joyride & Car Meet App     5.0
```

2.10 Result

- There are 271 five rated apps on Google Play store
- Top most is 'CT Brain Interpretation' from 'Family' Category

[ ]: