6.3-EDA+And+FE+Google+Playstore

July 17, 2025

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 20 column and 10841 rows.

0.1.1 Steps We Are Going to Follow

1. Data Clearning
2. Exploratory Data Analysis
3. Featur eEngineering

[1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

warnings.filterwarnings("ignore")

%matplotlib inline
```

[3]:
```python
df.shape
```

[3]:
```
(10841, 13)
```

[4]:
```python
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
```

[5]:
```
##summary of the dataset
df.describe()
```

[5]:
```
            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
```

[6]:
```
##Missing Values
df.isnull().sum()
```

[6]:
```
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
```

0.2 Insights and observation

The dataset has msising values

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

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

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

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

0.3 Data Cleaning

```
[8]: df['Reviews'].unique()
```

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

```
[9]: df['Reviews'].astype(int)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[9], line 1
----> 1 df['Reviews'].astype(int)

File /opt/conda/lib/python3.10/site-packages/pandas/core/generic.py:6240, in␣
 ↪NDFrame.astype(self, dtype, copy, errors)
   6233     results = [
   6234         self.iloc[:, i].astype(dtype, copy=copy)
   6235         for i in range(len(self.columns))
   6236     ]
   6238 else:
   6239     # else, only a single dtype is given
-> 6240     new_data = self._mgr.astype(dtype=dtype, copy=copy, errors=errors)
   6241     return self._constructor(new_data).__finalize__(self,␣
 ↪method="astype")
   6243 # GH 33113: handle empty frame or series

File /opt/conda/lib/python3.10/site-packages/pandas/core/internals/managers.py:
 ↪450, in BaseBlockManager.astype(self, dtype, copy, errors)
    449 def astype(self: T, dtype, copy: bool = False, errors: str = "raise") -
 ↪T:
--> 450     return self.apply("astype", dtype=dtype, copy=copy, errors=errors)
```

```
File /opt/conda/lib/python3.10/site-packages/pandas/core/internals/managers.py:
 ↪352, in BaseBlockManager.apply(self, f, align_keys, ignore_failures, **kwargs
    350         applied = b.apply(f, **kwargs)
    351     else:
--> 352         applied = getattr(b, f)(**kwargs)
    353 except (TypeError, NotImplementedError):
    354     if not ignore_failures:

File /opt/conda/lib/python3.10/site-packages/pandas/core/internals/blocks.py:
 ↪526, in Block.astype(self, dtype, copy, errors)
    508 """
    509 Coerce to the new dtype.
    510
   (…)
    522 Block
    523 """
    524 values = self.values
--> 526 new_values = astype_array_safe(values, dtype, copy=copy, errors=errors)
    528 new_values = maybe_coerce_values(new_values)
    529 newb = self.make_block(new_values)

File /opt/conda/lib/python3.10/site-packages/pandas/core/dtypes/astype.py:299,
 ↪in astype_array_safe(values, dtype, copy, errors)
    296     return values.copy()
    298 try:
--> 299     new_values = astype_array(values, dtype, copy=copy)
    300 except (ValueError, TypeError):
    301     # e.g. astype_nansafe can fail on object-dtype of strings
    302     #  trying to convert to float
    303     if errors == "ignore":

File /opt/conda/lib/python3.10/site-packages/pandas/core/dtypes/astype.py:230,
 ↪in astype_array(values, dtype, copy)
    227     values = values.astype(dtype, copy=copy)
    229 else:
--> 230     values = astype_nansafe(values, dtype, copy=copy)
    232 # in pandas we don't store numpy str dtypes, so convert to object
    233 if isinstance(dtype, np.dtype) and issubclass(values.dtype.type, str):

File /opt/conda/lib/python3.10/site-packages/pandas/core/dtypes/astype.py:170,
 ↪in astype_nansafe(arr, dtype, copy, skipna)
    166     raise ValueError(msg)
    168 if copy or is_object_dtype(arr.dtype) or is_object_dtype(dtype):
    169     # Explicit copy, or required since NumPy can't view from / to objec .
--> 170     return arr.astype(dtype, copy=True)
    172 return arr.astype(dtype, copy=copy)
```

```
ValueError: invalid literal for int() with base 10: '3.0M'
```

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

[10]:
```
df[~df['Reviews'].str.isnumeric()]
```

[10]:
```
                                          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
```

[11]:
```
df_copy=df.copy()
```

[12]:
```
df_copy=df_copy.drop(df_copy.index[10472])
```

[13]:
```
df_copy[~df_copy['Reviews'].str.isnumeric()]
```

[13]:
```
Empty DataFrame
Columns: [App, Category, Rating, Reviews, Size, Installs, Type, Price, Content
Rating, Genres, Last Updated, Current Ver, Android Ver]
Index: []
```

[14]:
```
## Convert Review Datatype to int
df_copy['Reviews']=df_copy['Reviews'].astype(int)
```

[15]:
```
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  int64
 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), int64(1), object(11)
memory usage: 1.2+ MB
```

[16]:
```
df_copy['Size'].unique()
```

[16]:
```
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)
```

[ ]:
```
19000K==19M
```

[17]:
```
df_copy['Size'].isnull().sum()
```

[17]:
```
0
```

[18]:
```
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)
```

[19]:
```
df_copy['Size']
```

[19]:
```
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
```

[20]:
```
df_copy['Installs'].unique()
```

[20]:
```
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)
```

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

```
[21]: 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)
```

```
[22]: 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,'')
```

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

```
[23]: 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)
```

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

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

[25]:
```
df_copy['Installs']=df_copy['Installs'].astype('int')
df_copy['Price']=df_copy['Price'].astype('float')
```

[26]:
```
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  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  Last Updated    10840 non-null  object
 11  Current Ver     10832 non-null  object
 12  Android Ver     10838 non-null  object
dtypes: float64(3), int64(2), object(8)
memory usage: 1.2+ MB
```

[27]:
```
## Handlling Last update feature
df_copy['Last Updated'].unique()
```

[27]:
```
array(['January 7, 2018', 'January 15, 2018', 'August 1, 2018', …,
       'January 20, 2014', 'February 16, 2014', 'March 23, 2014'],
      dtype=object)
```

[28]:
```
df_copy['Last Updated']=pd.to_datetime(df_copy['Last Updated'])
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
```

[29]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10840 entries, 0 to 10840
Data columns (total 16 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  Last Updated    10840 non-null  datetime64[ns]
 11  Current Ver     10832 non-null  object
 12  Android Ver     10838 non-null  object
 13  Day             10840 non-null  int64
 14  Month           10840 non-null  int64
 15  Year            10840 non-null  int64
dtypes: datetime64[ns](1), float64(3), int64(5), object(7)
memory usage: 1.4+ MB
```

[30]:
```
df_copy.head()
```

[30]:
```
                                                 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 Last Updated         Current Ver   Android Ver  \
0               Art & Design   2018-01-07               1.0.0  4.0.3 and up
1  Art & Design;Pretend Play   2018-01-15               2.0.0  4.0.3 and up
2               Art & Design   2018-08-01               1.2.4  4.0.3 and up
3               Art & Design   2018-06-08  Varies with device    4.2 and up
4    Art & Design;Creativity   2018-06-20                 1.1    4.4 and up

   Day  Month  Year
0    7      1  2018
1   15      1  2018
2    1      8  2018
3    8      6  2018
4   20      6  2018
```

[32]:
```
df_copy.to_csv('data/google_cleaned.csv')
```

0.4 EDA

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

```
[33]:                                                 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 Last Updated         Current Ver   Android Ver  \
      0               Art & Design   2018-01-07               1.0.0  4.0.3 and up
      1  Art & Design;Pretend Play   2018-01-15               2.0.0  4.0.3 and up
      2               Art & Design   2018-08-01               1.2.4  4.0.3 and up
      3               Art & Design   2018-06-08  Varies with device    4.2 and up
      4    Art & Design;Creativity   2018-06-20                 1.1    4.4 and up

        Day  Month  Year
      0   7      1  2018
      1  15      1  2018
      2   1      8  2018
      3   8      6  2018
      4  20      6  2018
```

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

```
[38]: (1181, 16)
```

0.5 Observation

The dataset has duplicate records

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

```
[40]: df_copy.shape
```

```
[40]: (9659, 16)
```

0.6 Explore Data

```
[42]: 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 columns
      print('We have {} numerical features : {}'.format(len(numeric_features),␣
        ↪numeric_features))
      print('\nWe have {} categorical features : {}'.
        ↪format(len(categorical_features), categorical_features))
```

```
We have 9 numerical features : ['Rating', 'Reviews', 'Size', 'Installs',
'Price', 'Last Updated', 'Day', 'Month', 'Year']

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

0.7 3.2 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

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

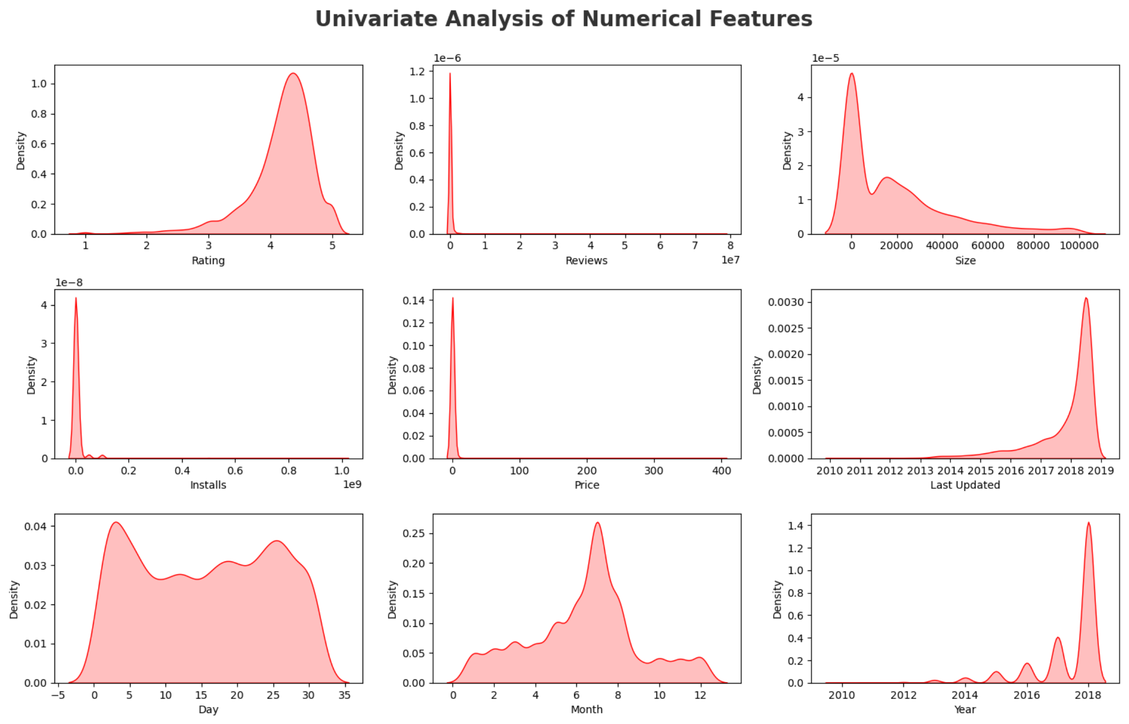
```
ROBLOX                                            0.083018
CBS Sports App - Scores, News, Stats & Watch Live 0.073794
ESPN                                              0.064570
Duolingo: Learn Languages Free                    0.064570
Candy Crush Saga                                  0.064570
                                                    …
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: App, Length: 9660, dtype: float64
---------------------------
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: Category, dtype: float64
---------------------------
Free    92.610701
Paid     7.380074
0        0.009225
Name: Type, dtype: float64
---------------------------
Everyone           80.387454
Teen               11.143911
Mature 17+          4.603321
```

```
Everyone 10+        3.819188
Adults only 18+     0.027675
Unrated             0.018450
Name: Content Rating, dtype: float64
---------------------------
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: Genres, Length: 120, dtype: float64
---------------------------
Varies with device    13.468107
1.0                    7.467922
1.1                    2.436998
1.2                    1.643127
2.0                    1.393889
                         …
1.0.17.3905            0.009231
15.1.2                 0.009231
4.94.19                0.009231
1.1.11.11              0.009231
2.0.148.0              0.009231
Name: Current Ver, Length: 2832, dtype: float64
---------------------------
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: Android Ver, dtype: float64
---------------------------
```

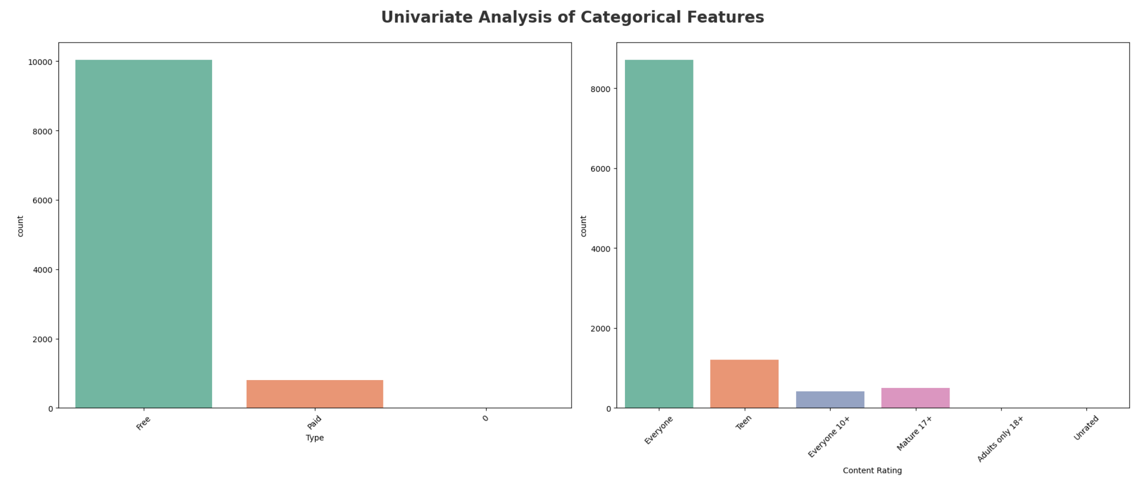
[46]:
```
## Proportion of count data on numerical columns
plt.figure(figsize=(15, 15))
plt.suptitle('Univariate Analysis of Numerical Features', fontsize=20,␣
 ↪fontweight='bold', alpha=0.8, y=1.)

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()
```



0.8 Observations

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

[47]:
```
# 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()
```



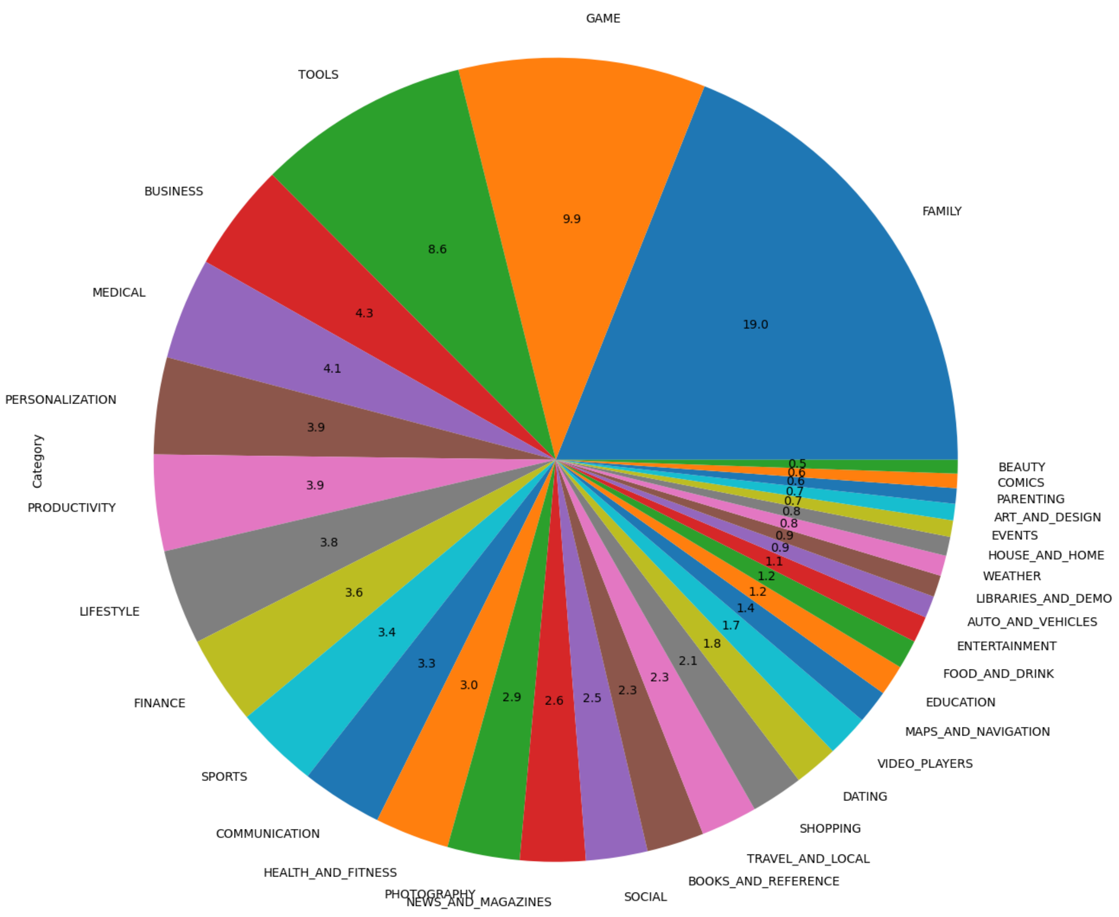
0.9 Which is the most popular app category?

[48]:
```
df_copy.head(2)
```

[48]:
```
                                                 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 Last Updated Current Ver   Android Ver  Day  \
0              Art & Design   2018-01-07       1.0.0  4.0.3 and up    7
1  Art & Design;Pretend Play   2018-01-15       2.0.0  4.0.3 and up   15

   Month  Year
0      1  2018
1      1  2018
```

[57]:
```
df_copy['Category'].value_counts().plot.
 ↪pie(y=df_copy['Category'],figsize=(15,16),autopct='%1.1f')
```

[57]:
```
<AxesSubplot: ylabel='Category'>
```



0.10 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

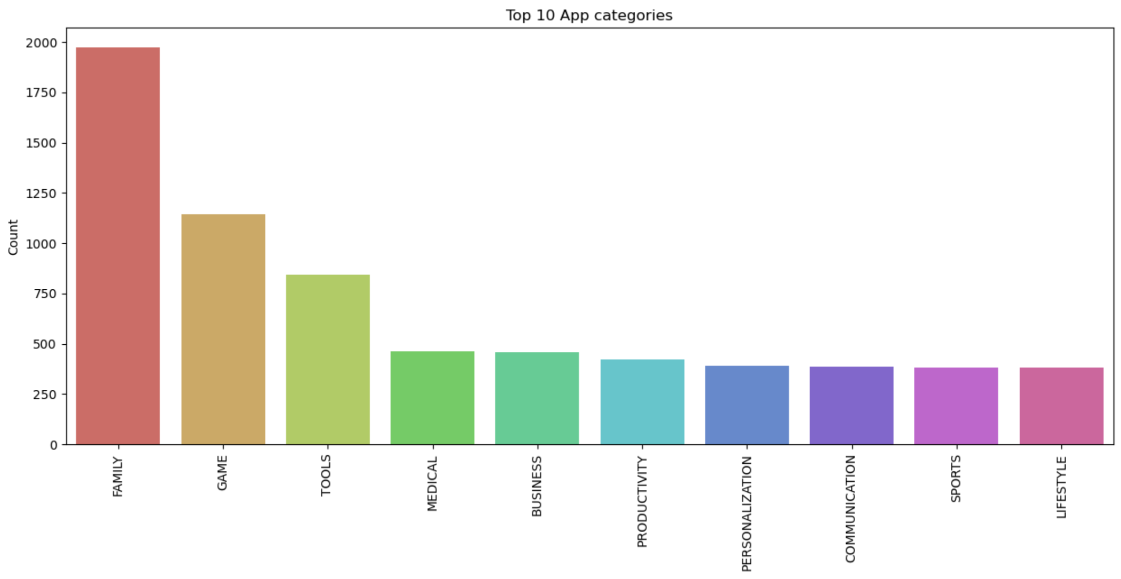
```
[53]: ## Top 10 App Categories
      category = pd.DataFrame(df_copy['Category'].value_counts())        #Dataframe␣
       ↪of apps on the basis of category
      category.rename(columns = {'Category':'Count'},inplace=True)
```

```
[55]: category
```

```
[55]:                     Count
      FAMILY               1972
      GAME                 1144
      TOOLS                 843
```

```
MEDICAL                  463
BUSINESS                 460
PRODUCTIVITY             424
PERSONALIZATION          392
COMMUNICATION            387
SPORTS                   384
LIFESTYLE                382
FINANCE                  366
HEALTH_AND_FITNESS       341
PHOTOGRAPHY              335
SOCIAL                   295
NEWS_AND_MAGAZINES       283
SHOPPING                 260
TRAVEL_AND_LOCAL         258
DATING                   234
BOOKS_AND_REFERENCE      231
VIDEO_PLAYERS            175
EDUCATION                156
ENTERTAINMENT            149
MAPS_AND_NAVIGATION      137
FOOD_AND_DRINK           127
HOUSE_AND_HOME            88
LIBRARIES_AND_DEMO        85
AUTO_AND_VEHICLES         85
WEATHER                   82
ART_AND_DESIGN            65
EVENTS                    64
PARENTING                 60
COMICS                    60
BEAUTY                    53
1.9                        1
```

[56]:
```
## top 10 app
plt.figure(figsize=(15,6))
sns.barplot(x=category.index[:10], y ='Count',data = category[:
 ↪10],palette='hls')
plt.title('Top 10 App categories')
plt.xticks(rotation=90)
plt.show()
```



0.11 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.

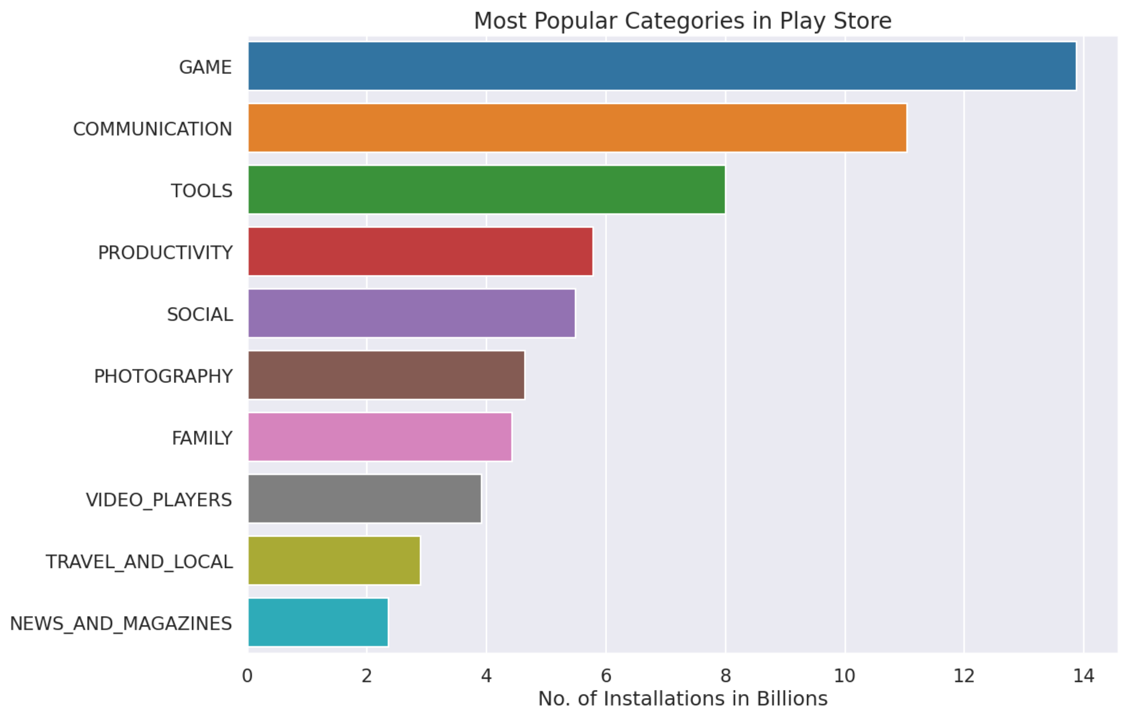
0.12 Internal Assignments

1. Which Category has largest number of installations??
2. What are the Top 5 most installed Apps in Each popular Categories ??
3. How many apps are there on Google Play Store which get 5 ratings??

0.13 Which Category has largest number of installations??

[58]:
```
df_cat_installs = df_copy.groupby(['Category'])['Installs'].sum().
 ↪sort_values(ascending = False).reset_index()
df_cat_installs.Installs = df_cat_installs.Installs/1000000000# converting into␣
 ↪billions
df2 = df_cat_installs.head(10)
plt.figure(figsize = (14,10))
sns.set_context("talk")
sns.set_style("darkgrid")

ax = sns.barplot(x = 'Installs' , y = 'Category' , data = df2 )
ax.set_xlabel('No. of Installations in Billions')
ax.set_ylabel('')
ax.set_title("Most Popular Categories in Play Store", size = 20)
```

```
[58]: Text(0.5, 1.0, 'Most Popular Categories in Play Store')
```

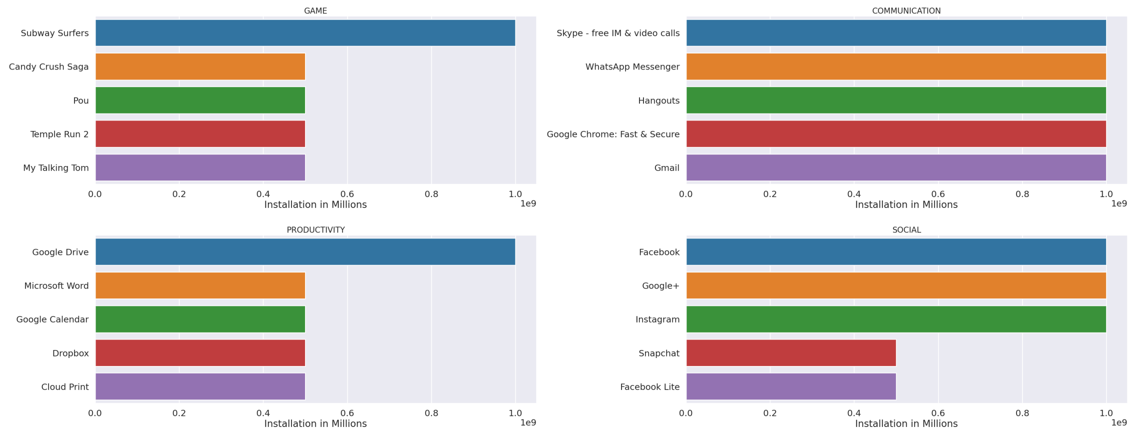


0.14 Insights

1. Out of all the categories “GAME” has the most number of Installations.
2. With almost 35 Billion Installations GAME is the most popular Category in Google App store

0.15 What are the Top 5 most installed Apps in Each popular Categories ??

```
[59]: 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))

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

plt.tight_layout()
plt.subplots_adjust(hspace= .3)
plt.show()
```



0.16 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.

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

```
[60]: 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(1)
```

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

```
[60]:   Category  Installs                          App  Rating
      0   FAMILY      1000  CS & IT Interview Questions     5.0
```

0.18 Result

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

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

```
[61]:                                                  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 Last Updated         Current Ver   Android Ver  \
      0                Art & Design   2018-01-07               1.0.0  4.0.3 and up
      1  Art & Design;Pretend Play   2018-01-15               2.0.0  4.0.3 and up
      2                Art & Design   2018-08-01               1.2.4  4.0.3 and up
      3                Art & Design   2018-06-08  Varies with device    4.2 and up
      4    Art & Design;Creativity   2018-06-20                 1.1    4.4 and up

         Day  Month  Year
      0    7      1  2018
      1   15      1  2018
      2    1      8  2018
      3    8      6  2018
      4   20      6  2018
```

```
[ ]:
```