EDA&FE Google Playstore

February 20, 2024

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

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

warnings.filterwarnings("ignore")

%matplotlib inline
```

[2]:
```
df=pd.read_csv('https://raw.githubusercontent.com/krishnaik06/playstore-Dataset/
↪main/googleplaystore.csv')
df.head()
```

[2]:

	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
```

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

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

[4]:
```
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]:
```
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 Value
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 Observations

The dataset has missing 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
```

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

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

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

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File /opt/conda/lib/python3.10/site-packages/pandas/core/indexes/base.py:3803,␣
 ↪in Index.get_loc(self, key, method, tolerance)
   3802 try:
-> 3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:

File /opt/conda/lib/python3.10/site-packages/pandas/_libs/index.pyx:138, in␣
 ↪pandas._libs.index.IndexEngine.get_loc()

File /opt/conda/lib/python3.10/site-packages/pandas/_libs/index.pyx:165, in␣
 ↪pandas._libs.index.IndexEngine.get_loc()

File pandas/_libs/hashtable_class_helper.pxi:5745, in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

File pandas/_libs/hashtable_class_helper.pxi:5753, in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

KeyError: '/Reviews'

The above exception was the direct cause of the following exception:

KeyError                                  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/frame.py:3805, in␣
 ↪DataFrame.__getitem__(self, key)
   3803 if self.columns.nlevels > 1:
   3804     return self._getitem_multilevel(key)
-> 3805 indexer = self.columns.get_loc(key)
   3806 if is_integer(indexer):
   3807     indexer = [indexer]

File /opt/conda/lib/python3.10/site-packages/pandas/core/indexes/base.py:3805,␣
 ↪in Index.get_loc(self, key, method, tolerance)
   3803     return self._engine.get_loc(casted_key)
   3804 except KeyError as err:
-> 3805     raise KeyError(key) from err
   3806 except TypeError:
   3807     # If we have a listlike key, _check_indexing_error will raise
```

```
   3808     #  InvalidIndexError. Otherwise we fall through and re-raise
   3809     #  the TypeError.
   3810     self._check_indexing_error(key)

KeyError: '/Reviews'
```

[10]:
```
df.shape
```

[10]:
```
(10841, 13)
```

[11]:
```
df['Reviews'].str.isnumeric().sum()
```

[11]:
```
10840
```

[12]:
```
df[~df['Reviews'].str.isnumeric()]
```

[12]:
```
                                            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
```

[13]:
```
df_copy=df.copy()
```

[14]:
```
df_copy=df_copy.drop(df_copy.index[10472])
```

[15]:
```
df_copy[~df_copy['Reviews'].str.isnumeric()]
```

[15]:
```
Empty DataFrame
Columns: [App, Category, Rating, Reviews, Size, Installs, Type, Price, Content
Rating, Genres, Last Updated, Current Ver, Android Ver]
Index: []
```

[16]:
```
df_copy['Reviews']=df_copy['Reviews'].astype(int)
```

[17]:
```
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
```

[18]:
```
df_copy['Size'].unique()
```

[18]:
```
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)
```

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

```
  Cell In[19], line 1
    19M=19000
     ^
SyntaxError: invalid decimal literal
```

[20]:
```
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)
```

[21]:
```
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  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(2), int64(1), object(10)
memory usage: 1.2+ MB
```

[22]:
```
df_copy['Installs'].unique()
```

[22]:
```
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)
```

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

[25]:
```
df_copy['Price'].unique()
```

[25]:
```
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)
```

[26]:
```
df_copy['Installs'].unique()
```

[26]:
```
array(['10000', '500000', '5000000', '50000000', '100000', '50000',
       '1000000', '10000000', '5000', '100000000', '1000000000', '1000',
       '500000000', '50', '100', '500', '10', '1', '5', '0'], dtype=object)
```

[27]:
```
df_copy['Installs']=df_copy['Installs'].astype('int')
df_copy['Price']=df_copy['Price'].astype('float')
```

[28]:
```
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
```

[29]:
```
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
```

[30]:
```
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
```

[31]:
```
df_copy['Content Rating'].value_counts()
```

[31]:
```
Everyone           8714
Teen               1208
Mature 17+          499
Everyone 10+        414
Adults only 18+       3
Unrated               2
Name: Content Rating, dtype: int64
```

0.3 EDA

[32]:
```
df_copy.head(2)
```

[32]:
```
                                                 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
```

[33]:
```
df_copy[df_copy.duplicated('App')].shape
```

[33]:
```
(1181, 16)
```

0.4 Observations

the dataset has duplicate records

[34]:
```
df_copy=df_copy.drop_duplicates(subset=['App'],keep='first')
```

[35]:
```
df_copy[df_copy.duplicated('App')].shape
```

[35]:
```
(0, 16)
```

[36]:
```
## Lets go ahead and explore more data
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))
```

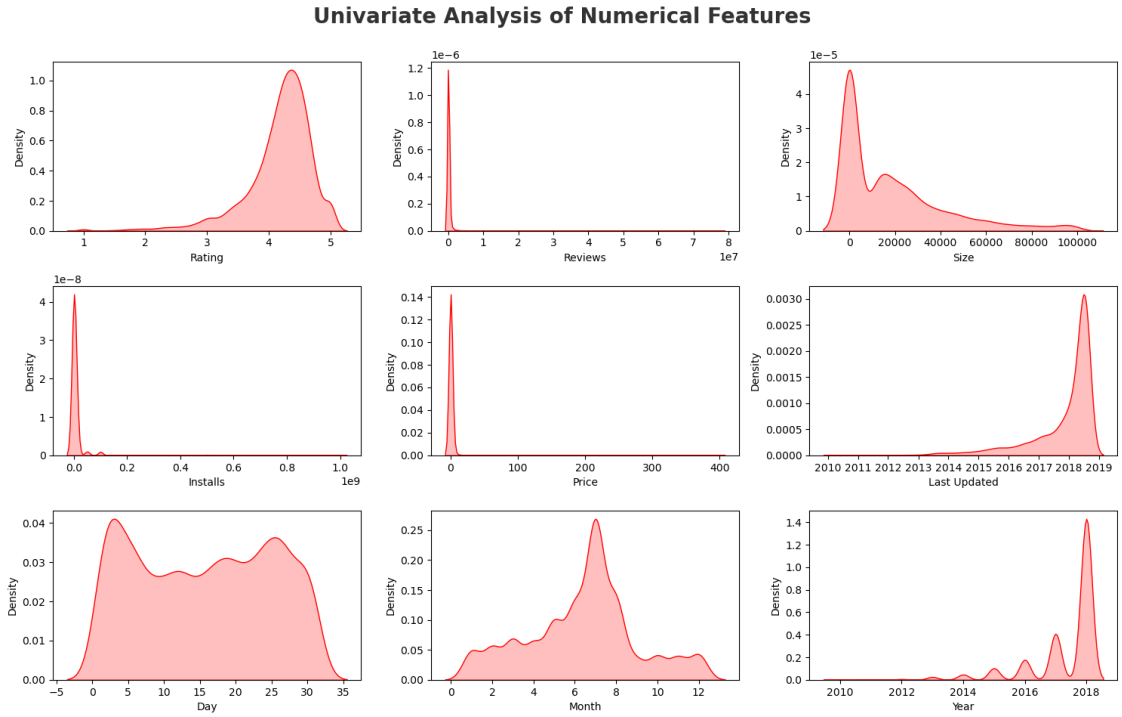
```
We have 9 numerical features : ['Rating', 'Reviews', 'Size', 'Installs',
'Price', 'Last Updated', 'Day', 'Month', 'Year']

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

[37]:
```
## Visualization Diagrams

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()
```

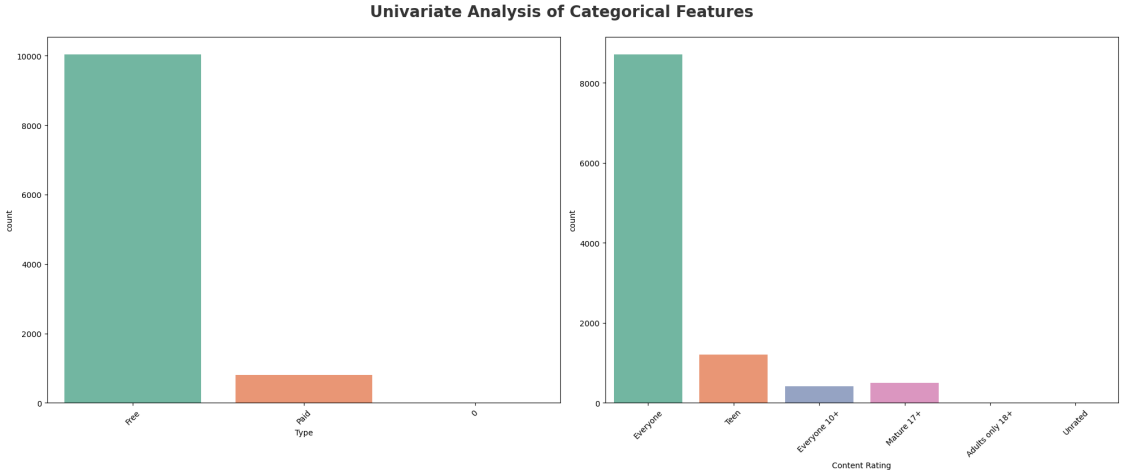


0.5 Observations

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

[38]:
```
# 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()
```



0.6 which is the most popular app category?

Indented block

[39]:
```
df_copy.head()
```

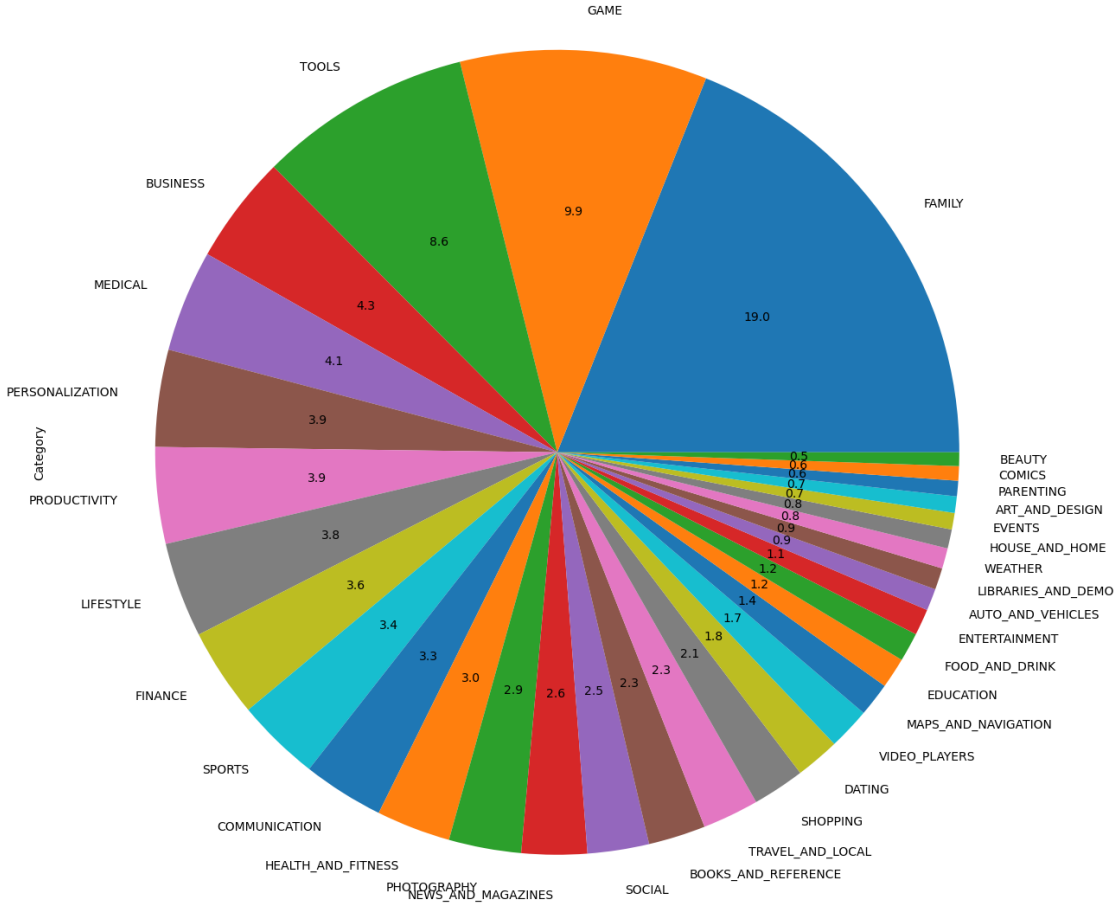
[39]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19000.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14000.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide …	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25000.0	50000000	Free	0.0	Teen	Art & Design	2018-06-08	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	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

[40]:
```
df_copy['Category'].value_counts().plot.
↪pie(y=df_copy['Category'],figsize=(15,16),autopct='%1.1f')
```

[40]: <AxesSubplot: ylabel='Category'>



0.7 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

[41]:
```
## 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)
```

[42]:
```
category
```

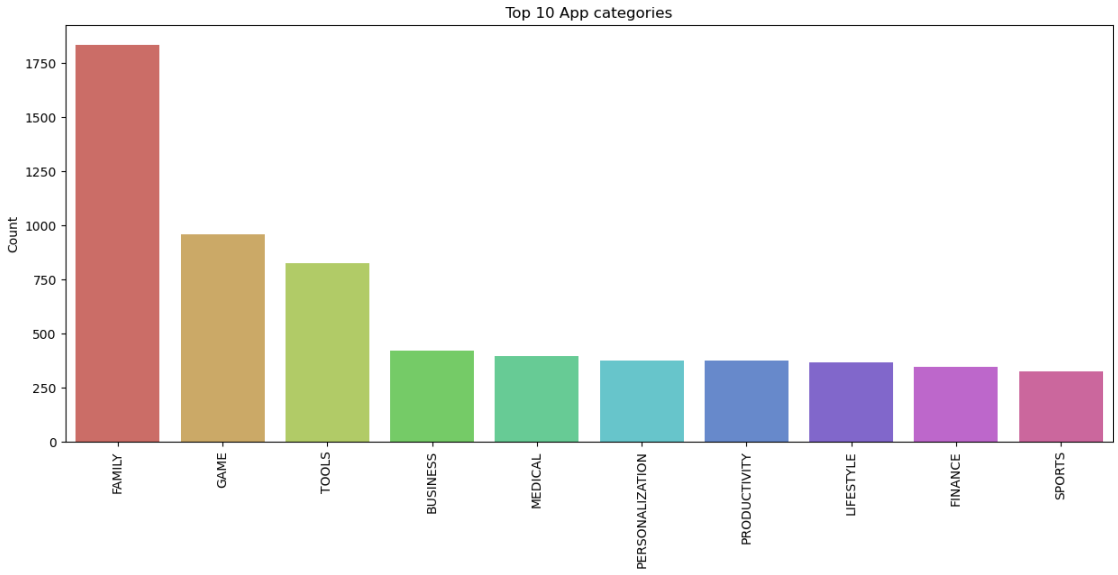
[42]:

	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

[ ]:
```
```

[43]:
```
## 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()
```



[44]:
```
category.index[:10]
```

[44]:
```
Index(['FAMILY', 'GAME', 'TOOLS', 'BUSINESS', 'MEDICAL', 'PERSONALIZATION',
       'PRODUCTIVITY', 'LIFESTYLE', 'FINANCE', 'SPORTS'],
      dtype='object')
```

0.8 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.

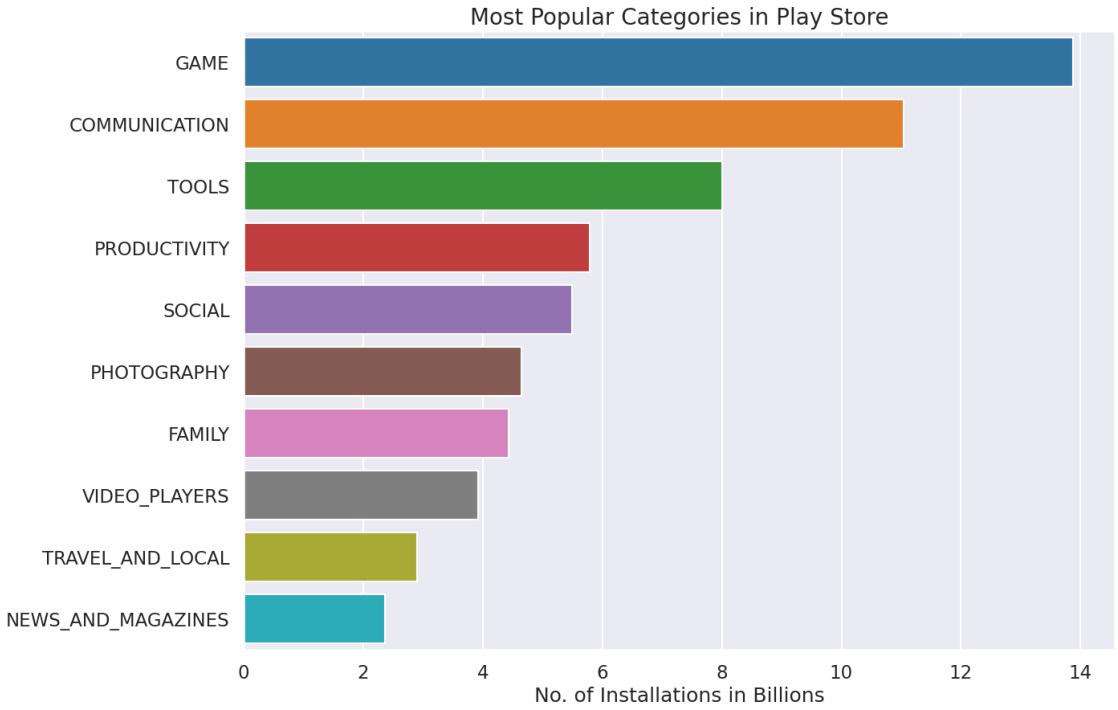
0.9 Internal Assignments

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??

0.10 Which Category has largest number of installations??

[45]:
```
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)
```

[45]:
```
Text(0.5, 1.0, 'Most Popular Categories in Play Store')
```



[ ]:

[ ]: