AN APPLICATION ON SOCIAL MEDIA ANALYZER

GOOGLE PLAYSTORE ANALYSIS

AIM:

Perform comprehensive analysis on the Google Play Store dataset using Jupyter Notebook, exploring various aspects such as app categories, ratings, reviews, and pricing to gain insights into the app market trends and user preferences.

OBJECTIVES:

Perform in-depth analysis on the Google Play Store dataset in Jupyter Notebook to uncover trends, preferences, and patterns in app categories, ratings, pricing, and user sentiments, enabling data-driven insights for app developers and stakeholders.

TOOLS AND LIBRARIES:

Jupyter Notebook: For interactive data analysis and visualization.

Python: Programming language for data manipulation and analysis.

Pandas: For data manipulation and analysis.

Matplotlib and Seaborn: For data visualization.

PREREQUISITES:

Ensure Python, along with Pandas, Matplotlib, Seaborn, and NLTK libraries, is installed. Obtain the Google Play Store dataset in a structured format like CSV. Familiarize yourself with data cleaning techniques. Develop basic statistical analysis skills. Install Jupyter Notebook for interactive analysis.

1. Data Loading and Inspection:

- Load the Google Play Store dataset into a DataFrame.
- Inspect the data to understand its structure and contents.

2. Data Cleaning:

- Handle missing or duplicate values.
- Convert data types if necessary.
- Address any inconsistencies or errors in the dataset.

3. Exploratory Data Analysis (EDA):

- Analyze the distribution of app categories and their frequencies.
- Investigate the distribution of app ratings and reviews.
- Explore the relationship between app ratings, reviews, and installs.
- Identify popular app categories based on ratings and installs.

4. Pricing Analysis:

- Analyze the distribution of app prices and identify pricing trends.
- Compare the pricing strategies across different app categories.

5. Conclusion and Insights:

- Summarize key findings and insights obtained from the analysis.
- Provide recommendations for app developers and stakeholders based on the analysis results.

PROGRAM:

Step 1: Dataset name

```
In [1]: project_name = "GooglePlayStoreAnalysis"
```

Step 2: Importing libraries

```
In [13]: # Imports
import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings('ignore')
```

Step 3: load the data as dataframe

```
In [14]: googlestore_df =  pd.read_csv("C:/Users/SANTHOSH L/Downloads/googleplaystore.csv")
```

```
In [22]: rows = googlestore_df.shape[0]
column = googlestore_df.shape[1]
```

Step 4: print the number of rows and the column

```
In [23]: print('There are {} Rows and {} Columns in the dataset'.format(rows, column))
         There are 10841 Rows and 13 Columns in the dataset
In [24]: googlestore_df.head(10)
```

Out[24]:

	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	19M	10,000+	Free	0	Everyone	Art & Design	7-Jan-18	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	15-Jan-18	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	1-Aug-18	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	8-Jun-18	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	20-Jun-18	1.1	4.4 and up
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167	5.6M	50,000+	Free	0	Everyone	Art & Design	26-Mar-17	1	2.3 and up
6	Smoke Effect Photo Maker - Smoke Editor	ART_AND_DESIGN	3.8	178	19M	50,000+	Free	0	Everyone	Art & Design	26-Apr-18	1.1	4.0.3 and up
7	Infinite Painter	ART_AND_DESIGN	4.1	36815	29M	1,000,000+	Free	0	Everyone	Art & Design	14-Jun-18	6.1.61.1	4.2 and up
8	Garden Coloring Book	ART_AND_DESIGN	4.4	13791	33M	1,000,000+	Free	0	Everyone	Art & Design	20-Sep-17	2.9.2	3.0 and up
9	Kids Paint Free - Drawing Fun	ART_AND_DESIGN	4.7	121	3.1M	10,000+	Free	0	Everyone	Art & Design;Creativity	3-Jul-18	2.8	4.0.3 and up

```
In [51]: googlestore_df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   App             10841 non-null  object
 1   Category        10841 non-null  object
 2   Rating          9367 non-null   float64
 3   Reviews         10841 non-null  object
 4   Size            10841 non-null  object
 5   Installs        10841 non-null  object
 6   Type            10840 non-null  object
 7   Price           10841 non-null  object
 8   Content Rating  10840 non-null  object
 9   Genres          10841 non-null  object
 10  Last Updated    10841 non-null  object
 11  Current Ver     10833 non-null  object
 12  Android Ver     10838 non-null  object
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

```
In [50]: googlestore_df.describe()
```

Out[50]:

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

```
In [52]: googlestore_df.columns
Out[52]: Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
                'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
                'Android Ver'],
               dtype='object')
```

Step 5: Column Price Converting the Price column from object to numeric

```
In [48]: googlestore_df['Price'].value_counts()
```

```
Out[48]: 0          10040
         $0.99        148
         $2.99        129
         $1.99         73
         $4.99         72
                      ...
         $1.75          1
         $14.00         1
         $4.85          1
         $46.99         1
         $1.04          1
         Name: Price, Length: 93, dtype: int64
```

```
In [97]: googlestore_df.describe()
```

Out[97]:

	Rating	Installs	Price
count	9367.000000	1.084100e+04	10841.000000
mean	4.193338	1.546291e+07	1.027273
std	0.537431	8.502557e+07	15.948971
min	1.000000	0.000000e+00	0.000000
25%	4.000000	1.000000e+03	0.000000
50%	4.300000	1.000000e+05	0.000000
75%	4.500000	5.000000e+06	0.000000
max	19.000000	1.000000e+09	400.000000

Step 6: Exploratory Analysis and Visualization

- Importing matplotlib.pyplot and seaborn

```
In [30]: import seaborn as sns
         import matplotlib
         import matplotlib.pyplot as plt
         %matplotlib inline

         sns.set_style('darkgrid')
         matplotlib.rcParams['font.size'] = 14
         matplotlib.rcParams['figure.figsize'] = (9, 5)
         matplotlib.rcParams['figure.facecolor'] = '#00000000'
```

- Can we see all the categories from the Category column.

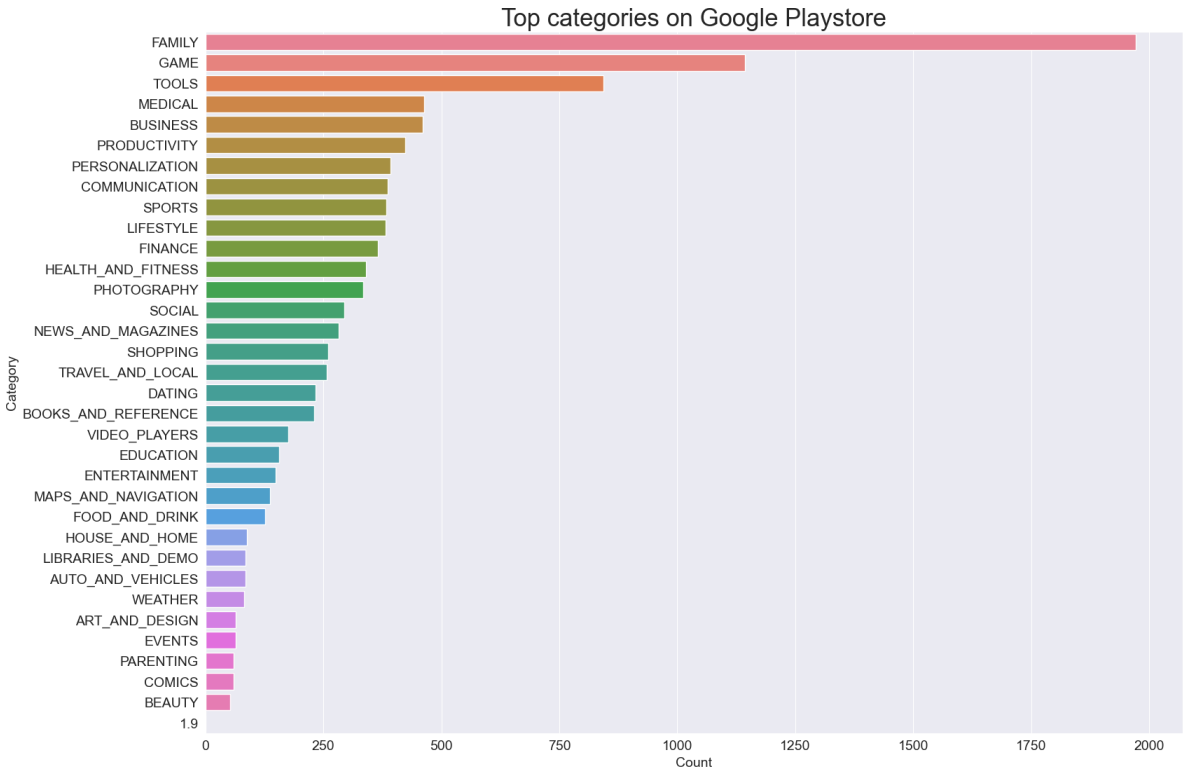
```
In [32]: googlestore_df['Category'].value_counts()
```

```
Out[32]: 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
         Name: Category, dtype: int64
```

- Let's Plot it and have a visual look.

```
In [33]: y = googlestore_df['Category'].value_counts().index
         x = googlestore_df['Category'].value_counts()
         xsis = []
         ysis = []
         for i in range(len(x)):
             xsis.append(x[i])
             ysis.append(y[i])
```

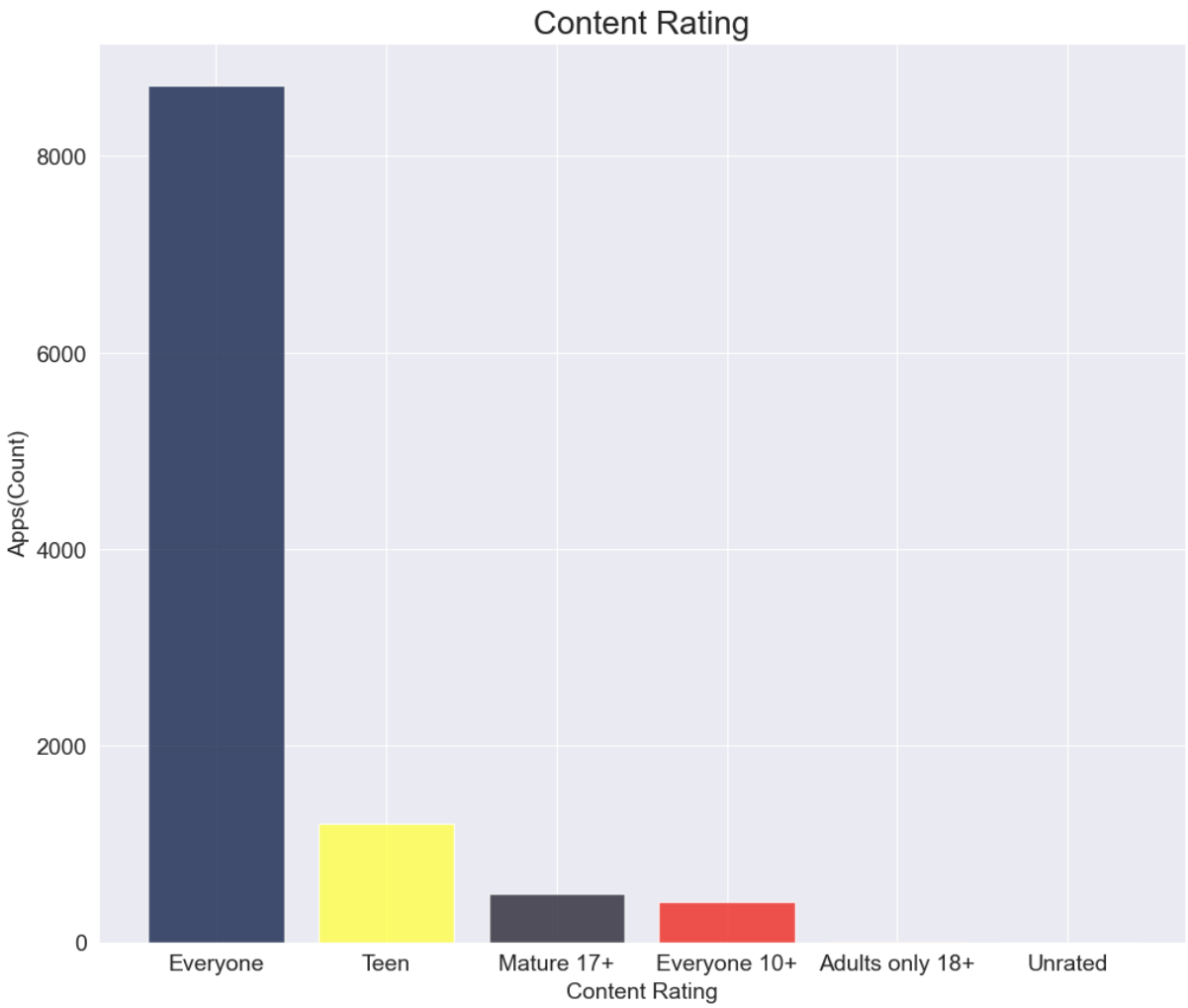
```
In [34]: plt.figure(figsize=(18,13))
         plt.xlabel("Count")
         plt.ylabel("Category")

         graph = sns.barplot(x = xsis, y = ysis, palette= "husl")
         graph.set_title("Top categories on Google Playstore", fontsize = 25);
```



- Which category of Apps from the Content Rating column are found more on playstore?

In [35]:
```
x2 = googlestore_df['Content Rating'].value_counts().index
y2 = googlestore_df['Content Rating'].value_counts()

x2sis = []
y2sis = []
for i in range(len(x2)):
    x2sis.append(x2[i])
    y2sis.append(y2[i])
```

In [36]:
```
plt.figure(figsize=(12,10))
plt.bar(x2sis,y2sis,width=0.8,color=['#15244C','#FFFF48','#292734','#EF2920','#CD202D','#ECC5F2'], alpha=0.8);
plt.title('Content Rating',size = 20);
plt.ylabel('Apps(Count)');
plt.xlabel('Content Rating');
```

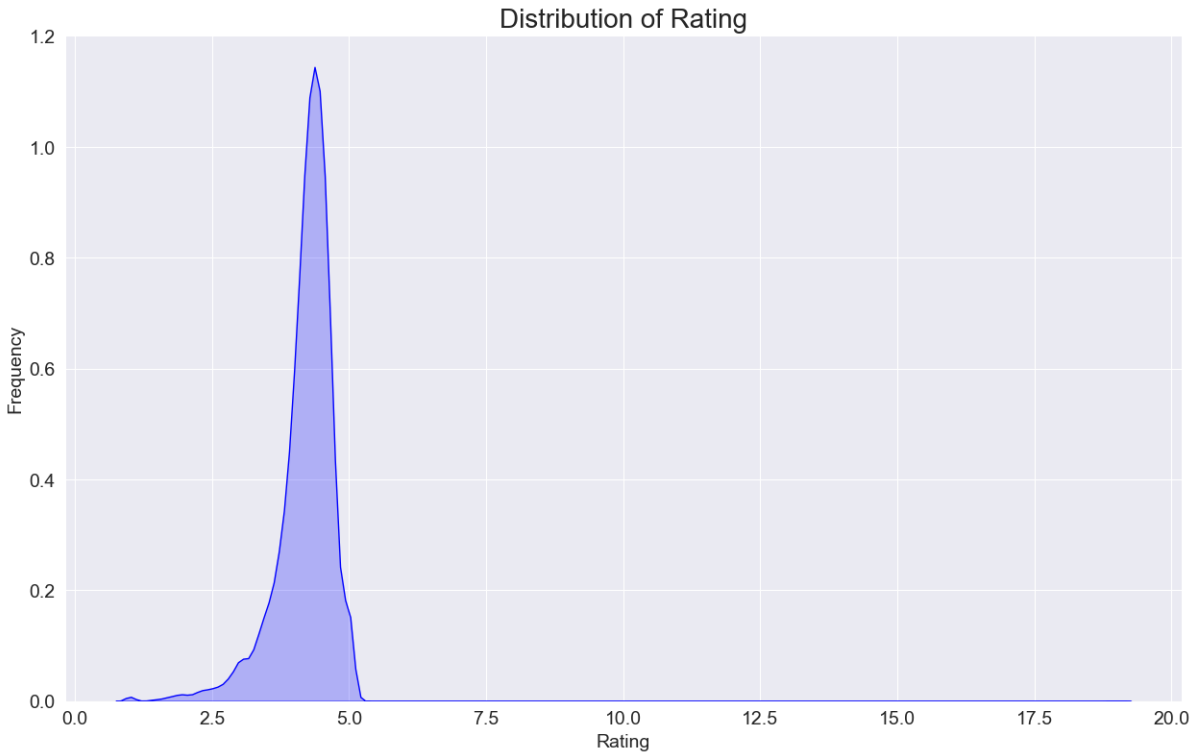


➢ distribution of the ratings the dataframe.

```
In [43]: googlestore_df['Rating'].describe()
```

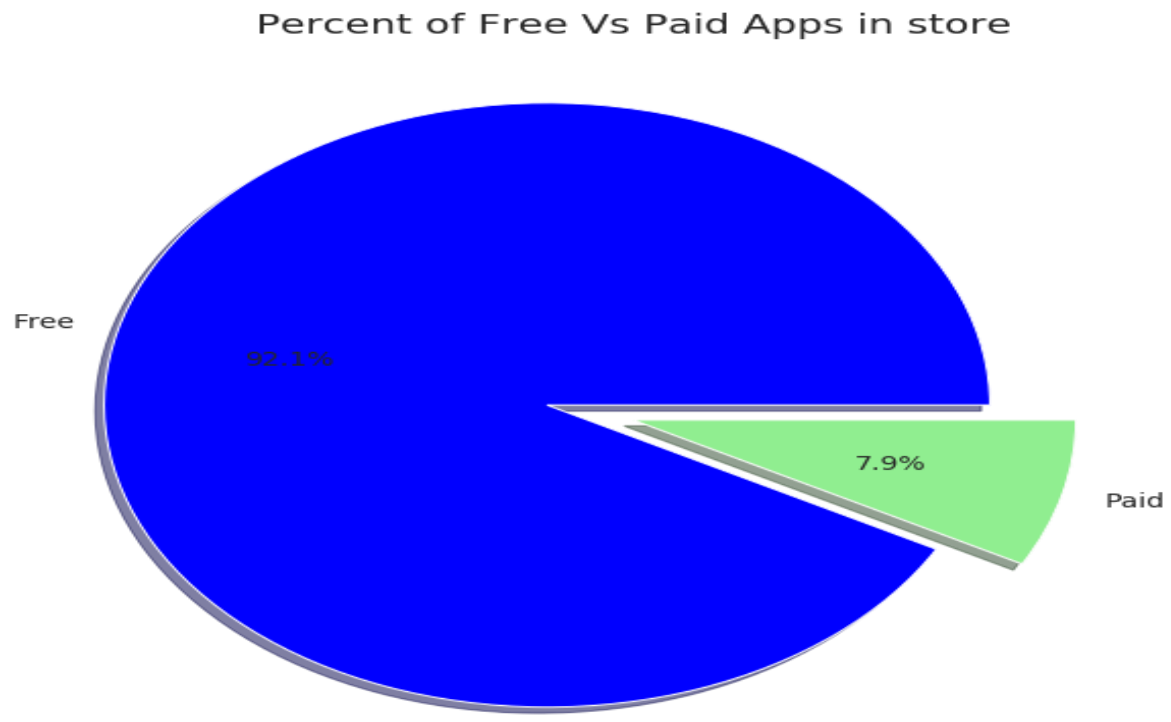
```
Out[43]: count    9367.000000
         mean        4.193338
         std         0.537431
         min         1.000000
         25%         4.000000
         50%         4.300000
         75%         4.500000
         max        19.000000
         Name: Rating, dtype: float64
```

```
In [44]: plt.figure(figsize=(15,9))
         plt.xlabel("Rating")
         plt.ylabel("Frequency")
         graph = sns.kdeplot(googlestore_df.Rating, color="Blue", shade = True)
         plt.title('Distribution of Rating',size = 20);
```



- Graph to view what portion of the apps in playstore are paid and free

```
In [54]: plt.figure(figsize=(10,10))
labels = googlestore_df['Type'].value_counts(sort = True).index
sizes = googlestore_df['Type'].value_counts(sort = True)
colors = ["blue","lightgreen"]
explode = (0.2,0)
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=0)
plt.title('Percent of Free Vs Paid Apps in store',size = 20)
plt.show()
```



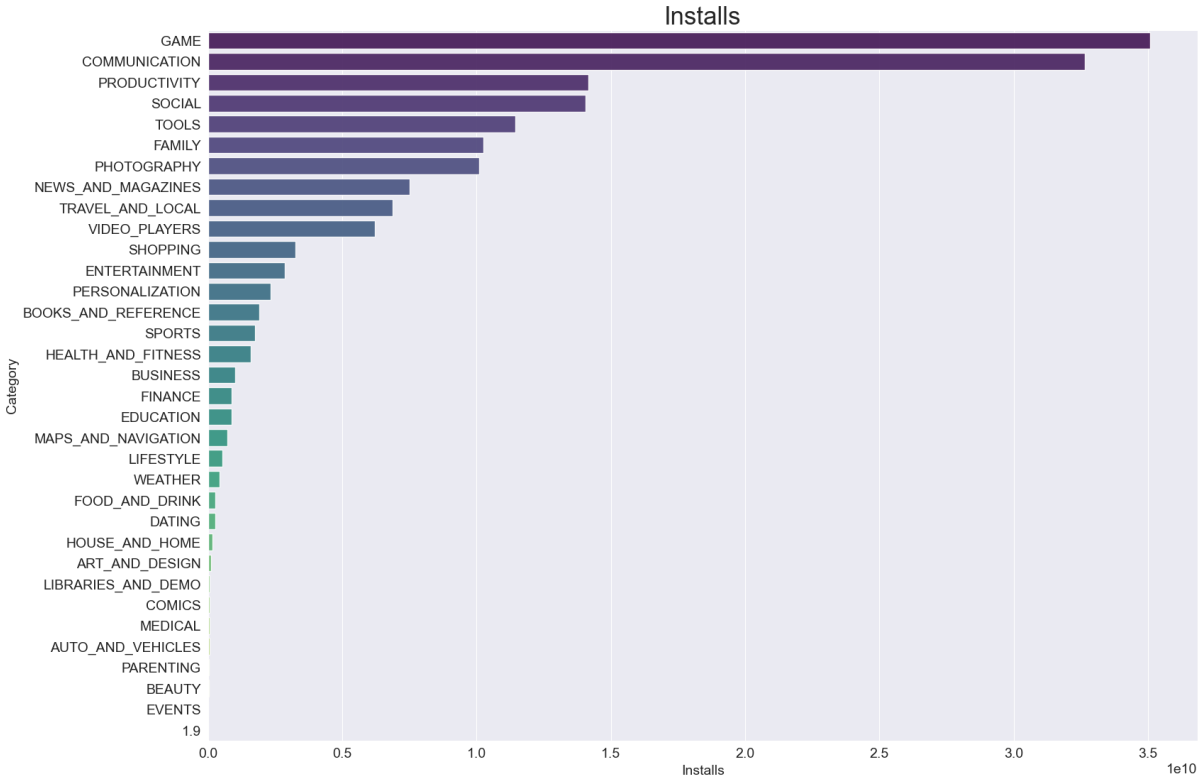
➢ Which category App's have most number of installs?

```
In [99]: highest_Installs_df = googlestore_df.groupby('Category')[['Installs']].sum().sort_values(by='Installs', ascending=False)

In [100]: highest_Installs_df.head()
```

Out[100]:

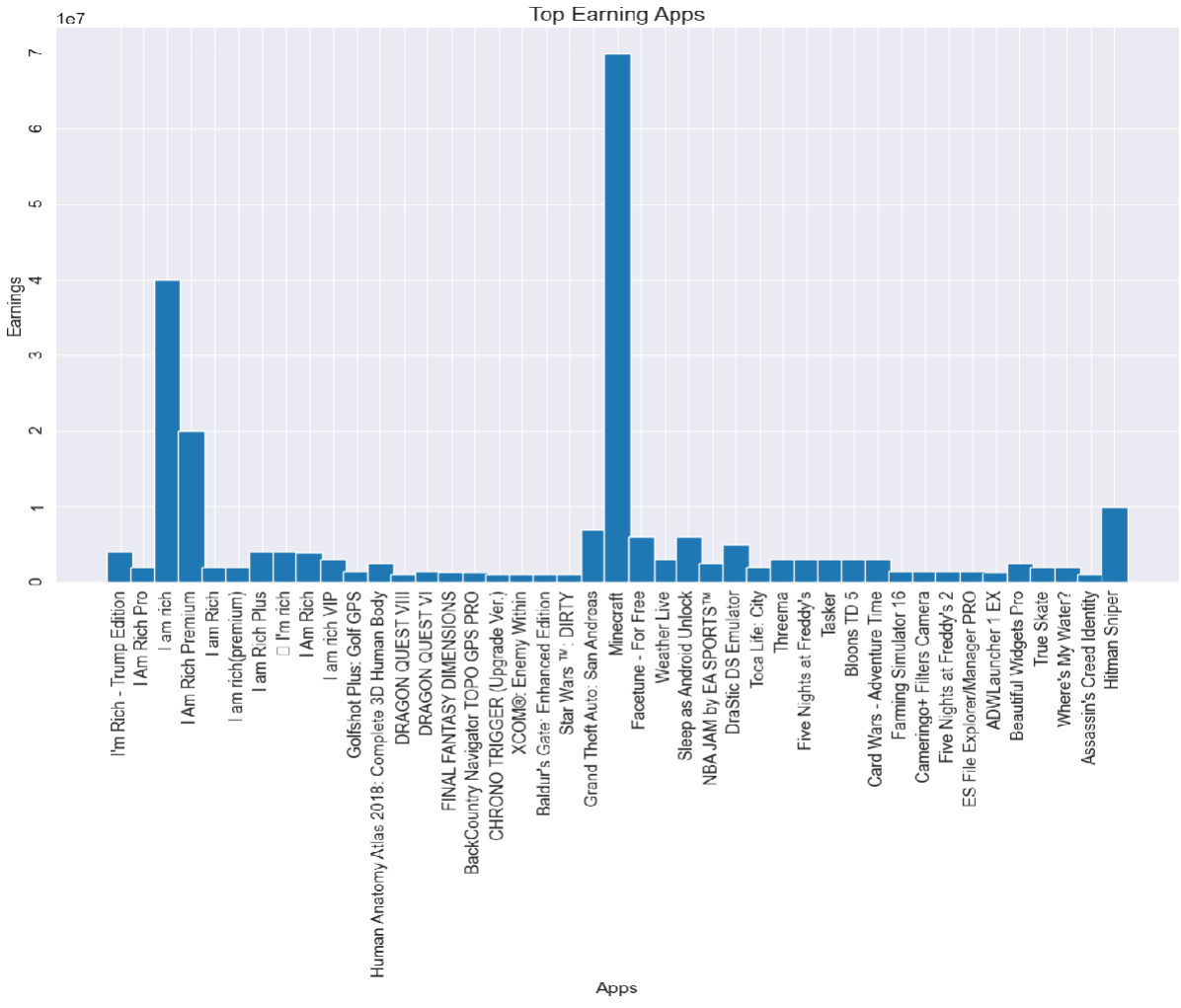
Category	Installs
GAME	35086024415
COMMUNICATION	32647276251
PRODUCTIVITY	14176091369
SOCIAL	14069867902
TOOLS	11452771915

```
In [101]: x2sis = []
          y2sis = []

          for i in range(len(highest_Installs_df)):
              x2sis.append(highest_Installs_df.Installs[i])
              y2sis.append(highest_Installs_df.index[i])

          plt.figure(figsize=(18,13))

          plt.xlabel("Installs")
          plt.ylabel("Category")
          graph = sns.barplot(x = x2sis, y = y2sis, alpha =0.9, palette= "viridis")
          graph.set_title("Installs", fontsize = 25);
```



➢ Which are the apps that have made the highest earning?

```
In [102]:
Paid_Apps_df = googlestore_df[googlestore_df['Type'] == 'Paid']
earning_df = Paid_Apps_df[['App', 'Installs', 'Price']]
earning_df['Earnings'] = earning_df['Installs'] * earning_df['Price'];
earning_df_sorted_by_Earnings = earning_df.sort_values(by='Earnings', ascending=False).head(50)
earning_df_sorted_by_Price = earning_df_sorted_by_Earnings.sort_values(by='Price', ascending=False)
# PLot a bar chart of earning at y and app names at x
plt.figure(figsize=(15,9))
plt.bar(earning_df_sorted_by_Price.App, earning_df_sorted_by_Price.Earnings, width=1.1, label=earning_df_sorted_by_Price.Ear
plt.xlabel("Apps")
plt.ylabel("Earnings")
plt.tick_params(rotation=90)
plt.title("Top Earning Apps");
```



RESULT:

Thus the social media analyzer on google playstore analysis is executed successfully in jupyter.