Data Manipulation and Visualization

A project by **Francesco Genna**

Full code notebook

GitHub

Google Drive

start2impact UNIVERSITY

Introduction

Data-driven analysis of Google Play apps.

01 The project

Providing support to stakeholders during the development and launch of an app

02 Data source

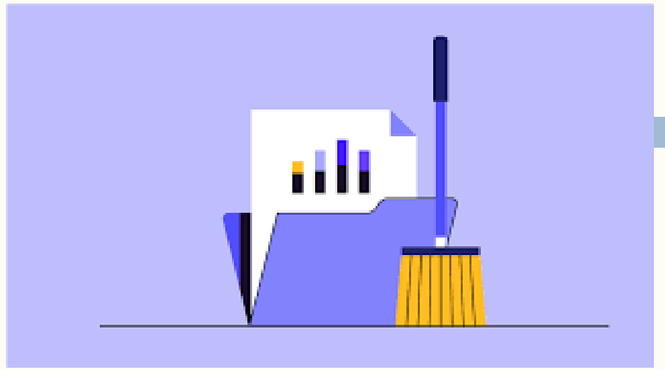
The datasets used in this project were imported from Kaggle and are available at the following link: https://www.kaggle.com/datasets/lava18/google-play-store-apps.

03 Python

The project was developed using Python, along with key data analysis and visualization libraries such as **Pandas**, **NumPy,** **Matplotlib,** and **Seaborn**

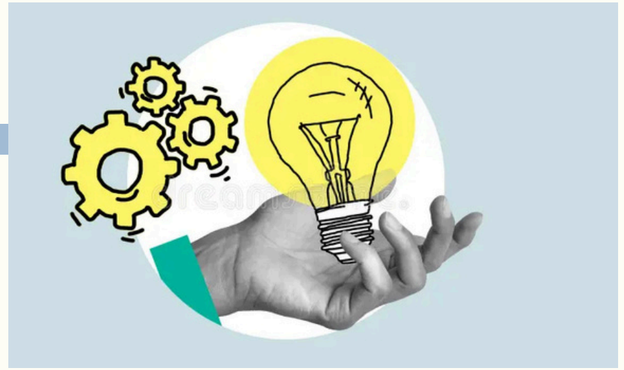
The app market is rapidly growing, with high competition in every sector.
A data-driven approach using ***Python*** helps identify promising niches
and understand the dynamics to develop a sustainable app.

Data Cleaning



Exploratory Analysis

Strategic Conclusion



📥 Importing Datasets and Initial Exploration

[169]:
```
import pandas as pd
import numpy as np

df = pd.read_csv("googleplaystore.csv")
Gr = pd.read_csv("googleplaystore_user_reviews.csv")
```

Preliminary exploration of the two dataframes

[171]:
```
df.head()
```

[171]:

	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	January 7, 2018	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	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide Apps	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	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	June 8, 2018	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	June 20, 2018	1.1	4.4 and up

[43]:
```
Gr.head()
```

[43]:

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
0	10 Best Foods for You	I like eat delicious food. That's I'm cooking ...	Positive	1.00	0.533333
1	10 Best Foods for You	This help eating healthy exercise regular basis	Positive	0.25	0.288462
2	10 Best Foods for You	NaN	NaN	NaN	NaN
3	10 Best Foods for You	Works great especially going grocery store	Positive	0.40	0.875000
4	10 Best Foods for You	Best idea us	Positive	1.00	0.300000

Data cleaning

Pre-processing

```
df.dtypes
```

```
App                object
Category           object
Rating            float64
Reviews            object
Size               object
Installs           object
Type               object
Price              object
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

Some **data types** are incorrect:
"Reviews" should be an integer
"Last Updated" a datetime
"Installs" an integer.

```
df.nunique()
```

```
App                9660
Category             34
Rating               40
Reviews            6002
Size                462
Installs             22
Type                  3
Price                93
Content Rating        6
Genres              120
Last Updated       1378
Current Ver        2832
Android Ver          33
dtype: int64
```

There are 3 unique values in the "Type" column.
Since "Type" should only be "Free" or "Paid," this indicates an error.

```
# Calculate total number of rows
total_rows = df.shape[0]

# Create boolean mask of duplicate rows (marks all but the first occurrence)
dup_mask = df.duplicated()

# Absolute count of duplicates
num_duplicates = dup_mask.sum()

# Percentage of duplicates
pct_duplicates = num_duplicates / total_rows * 100

print(f"Total rows: {total_rows}")
print(f"Duplicate rows: {num_duplicates}")
print(f"Duplicates (%): {pct_duplicates:.2f}%")
```

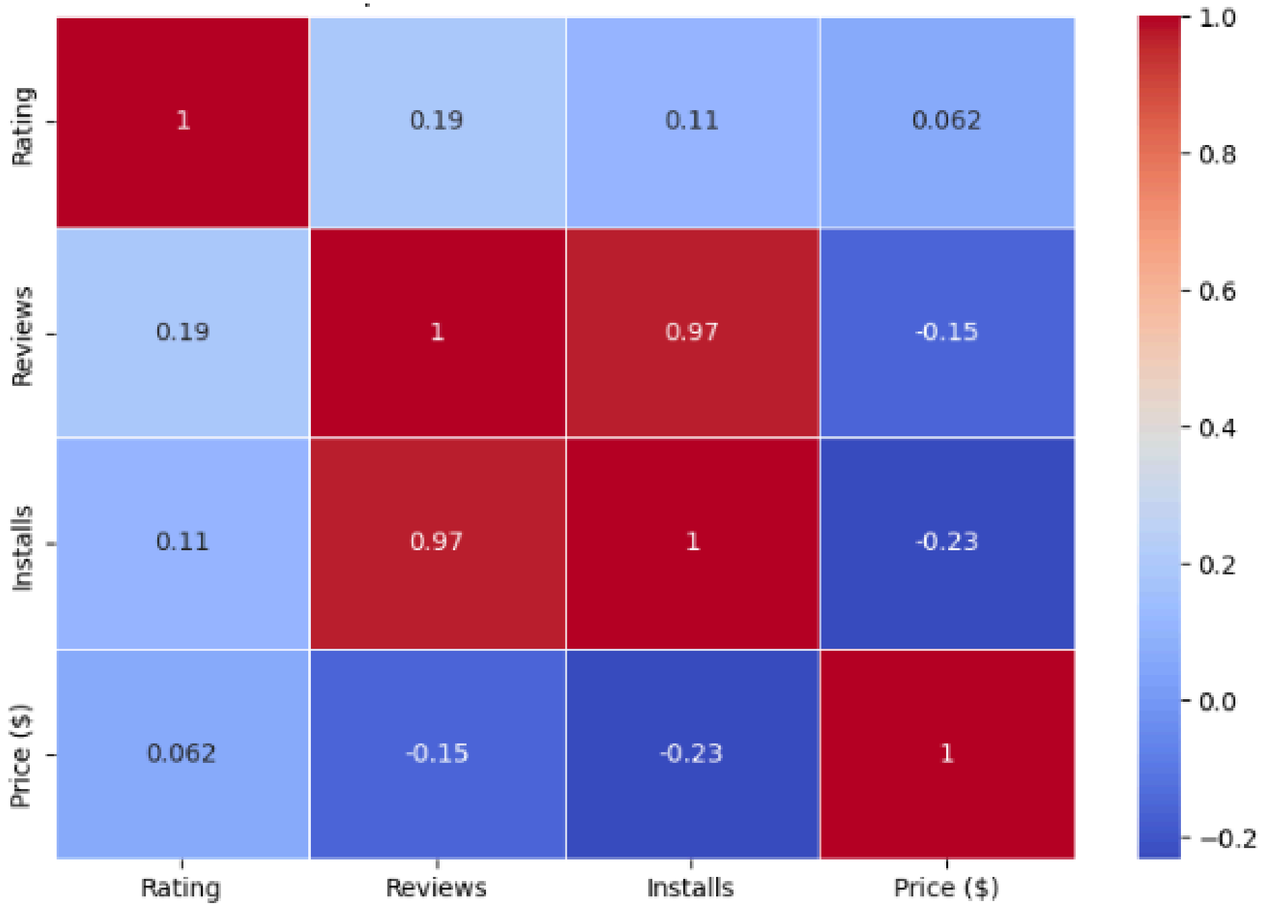
```
Total rows: 10841
Duplicate rows: 483
Duplicates (%): 4.46%
```

483 rows (4,46%) are duplicates

Note: Full data processing and cleaning is detailed in the Jupyter notebook 📊📄.

Exploratory Analysis 🔍

Spearman Correlation Matrix

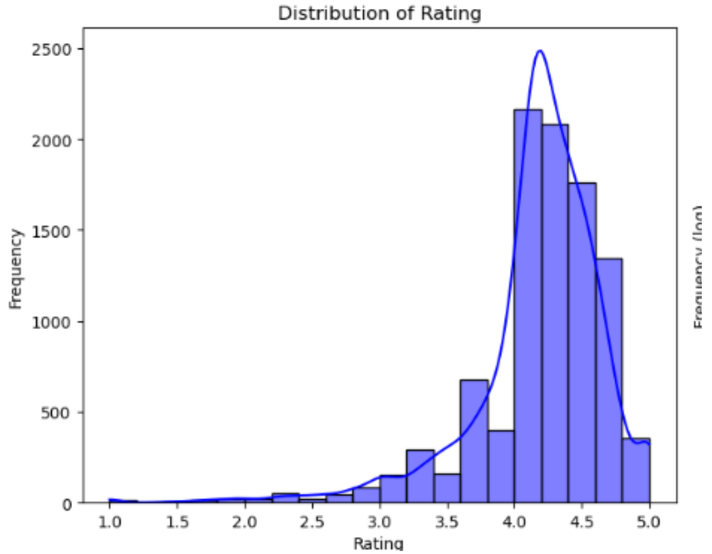


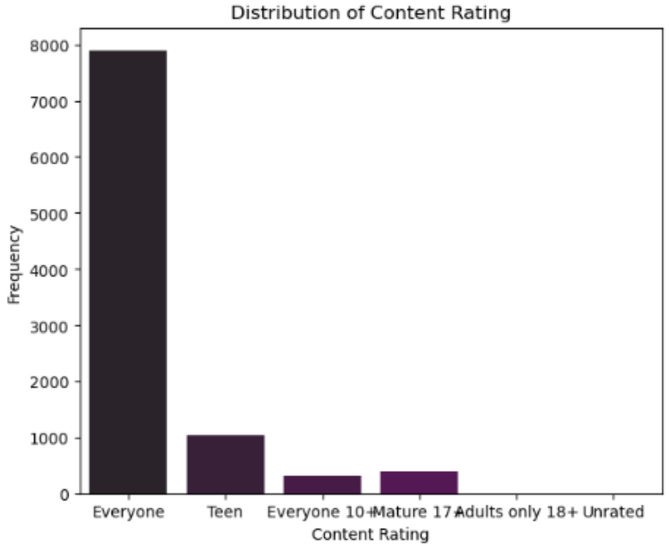
In addition to the correlation between *Number of Installs and Number of Reviews,* no significant correlations were observed.
A weak negative correlation (ρ = - 0.2) was found between Price and Number of Installs, suggesting that as the price increases, the number of installs tends to decrease slightly. However, this relationship is weak and should be interpreted with caution.

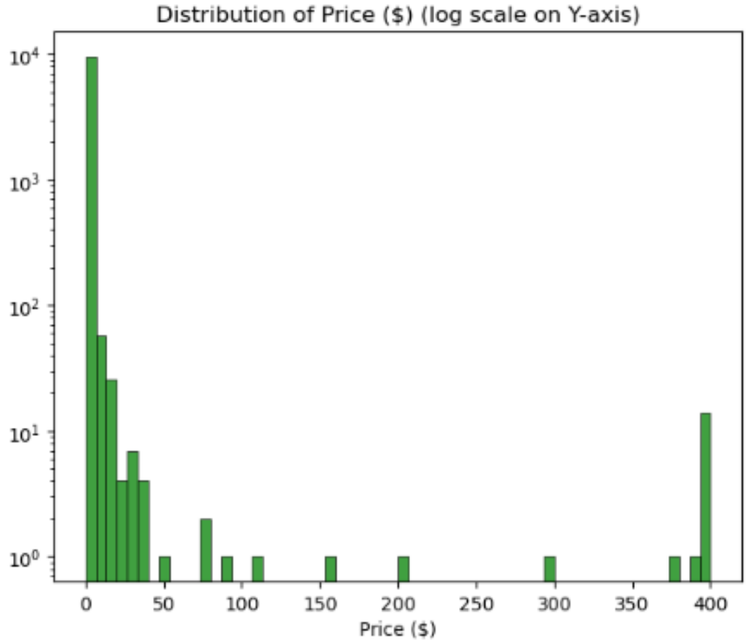
It is important to note that `the correlation value obtained represents a relationship between the two variables and does not necessarily imply causality.`
This means that while the two variables move together, it does not indicate that one causes the other.

Exploratory visualization 📊

```python
plt.figure(figsize=(12, 10))

# Histogram of Rating distribution
plt.subplot(2, 2, 1)
sns.histplot(df['Rating'], kde=True, bins=20, color='blue')
plt.title('Distribution of Rating')
plt.xlabel('Rating')
plt.ylabel('Frequency')

# Histogram of Price ($) distribution with log-scaled Y-axis
plt.subplot(2, 2, 2)
sns.histplot(df['Price ($)'], kde=False, bins=60, color='green')
plt.yscale('log')  # Logarithmic scale for Y-axis
plt.title('Distribution of Price ($) (log scale on Y-axis)')
plt.xlabel('Price ($)')
plt.ylabel('Frequency (log)')

# Bar plot of Content Rating distribution
plt.subplot(2, 2, 3)
sns.countplot(x='Content Rating', data=df, palette='dark:purple'
plt.title('Distribution of Content Rating')
plt.xlabel('Content Rating')
plt.ylabel('Frequency')

# Optimize layout
plt.tight_layout()
plt.show()
```







Data visualization

Market Analysis

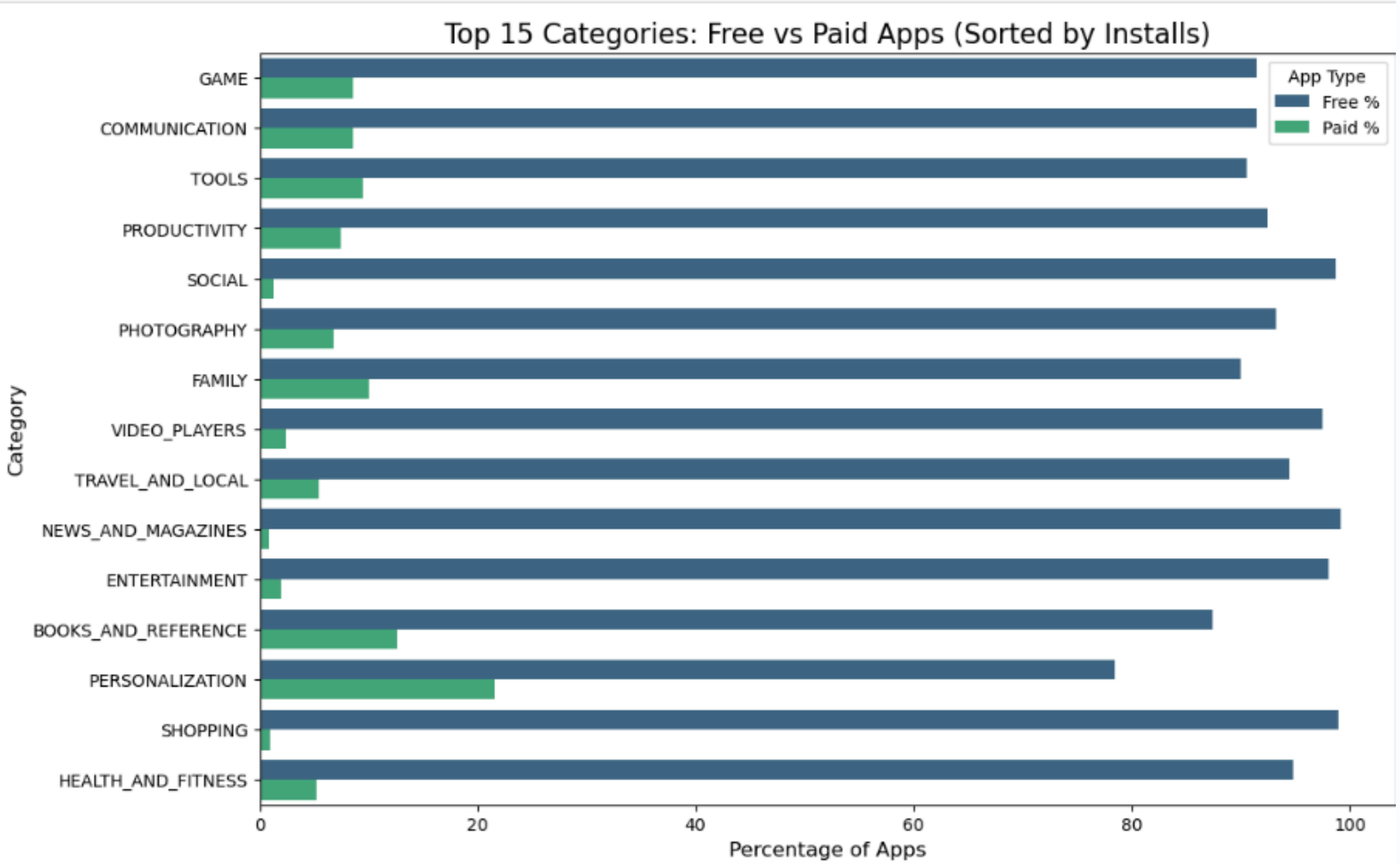
A focused overview of the app ecosystem through two key perspectives: top-performing categories and genres, and the sentiment polarity associated with different categories

Free App VS Paid App.

Distinction between free and paid applications, focusing on their potential influence on user ratings and sentiment.

Data visualization

Categories - Top 15 for number of installs



This bar plot displays the top 15 categories ordered by the number of installations.

For each category, the plot shows the percentage of paid and free apps.

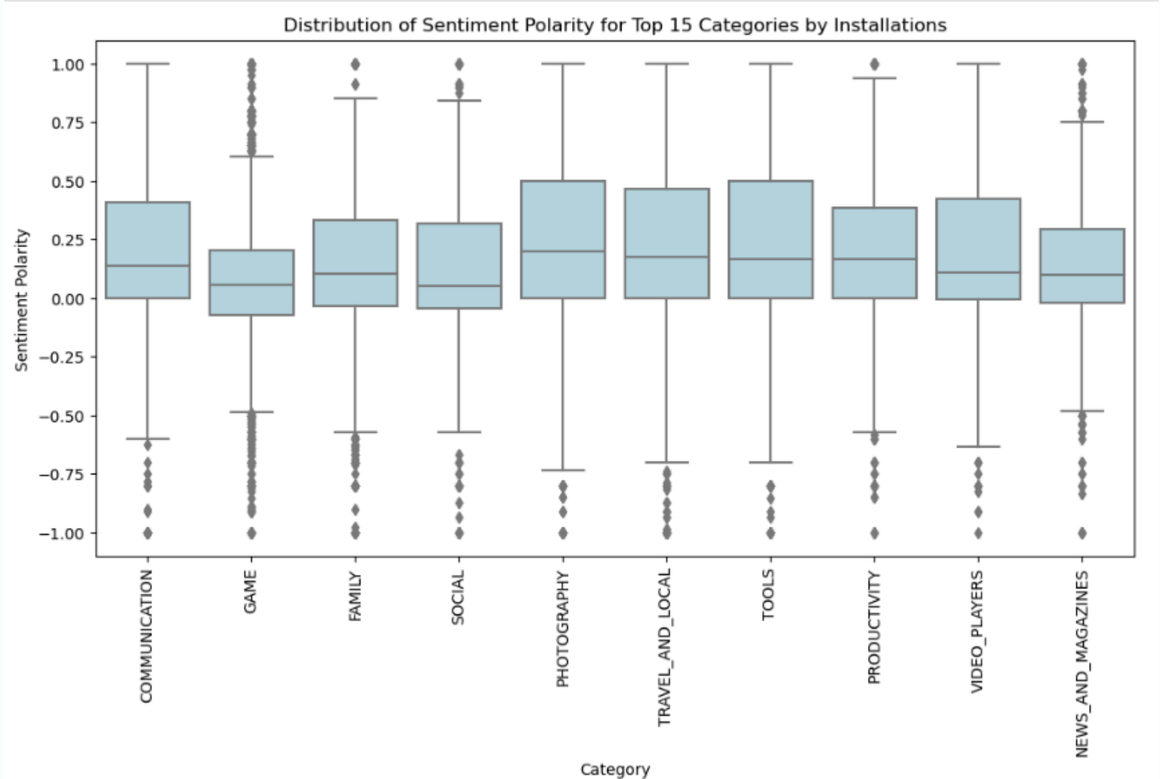
Paid apps are significantly fewer

For the visualization of all categories, refer to the jupiter notebook.

Data visualization

Top 15 Categories - Distribution of sentiment polarity

```
# Top 15 categories by total number of installations
top_15_categories = df.groupby('Category')['Installs'].sum().nlargest(10).index

# Merge the dataframes on the 'App' column
merged_df = pd.merge(df, Gr, on='App')

# Filter for categories with the most installations
filtered_df = merged_df[merged_df['Category'].isin(top_15_categories)]

# Create the boxplot
plt.figure(figsize=(12, 6))
sns.boxplot(data=filtered_df, x='Category', y='Sentiment_Polarity', color='lightblue')
plt.xticks(rotation=90)
plt.title('Distribution of Sentiment Polarity for Top 15 Categories by Installations')
plt.xlabel('Category')
plt.ylabel('Sentiment Polarity')
plt.show()
```



The boxplots of various app categories indicate that the median Sentiment Polarity is similar across categories, suggesting that `**the average sentiment of reviews does not differ substantially among them.**

However, the GAME category exhibits a higher number of outliers, both above and below the median.
These outliers suggest that within this category, some apps receive reviews with extreme sentiments, both highly positive and highly negative.
This can be attributed to:

Polarization of Opinions : Games tend to evoke strong emotions, leading to enthusiastic positive reviews and highly critical negative reviews.

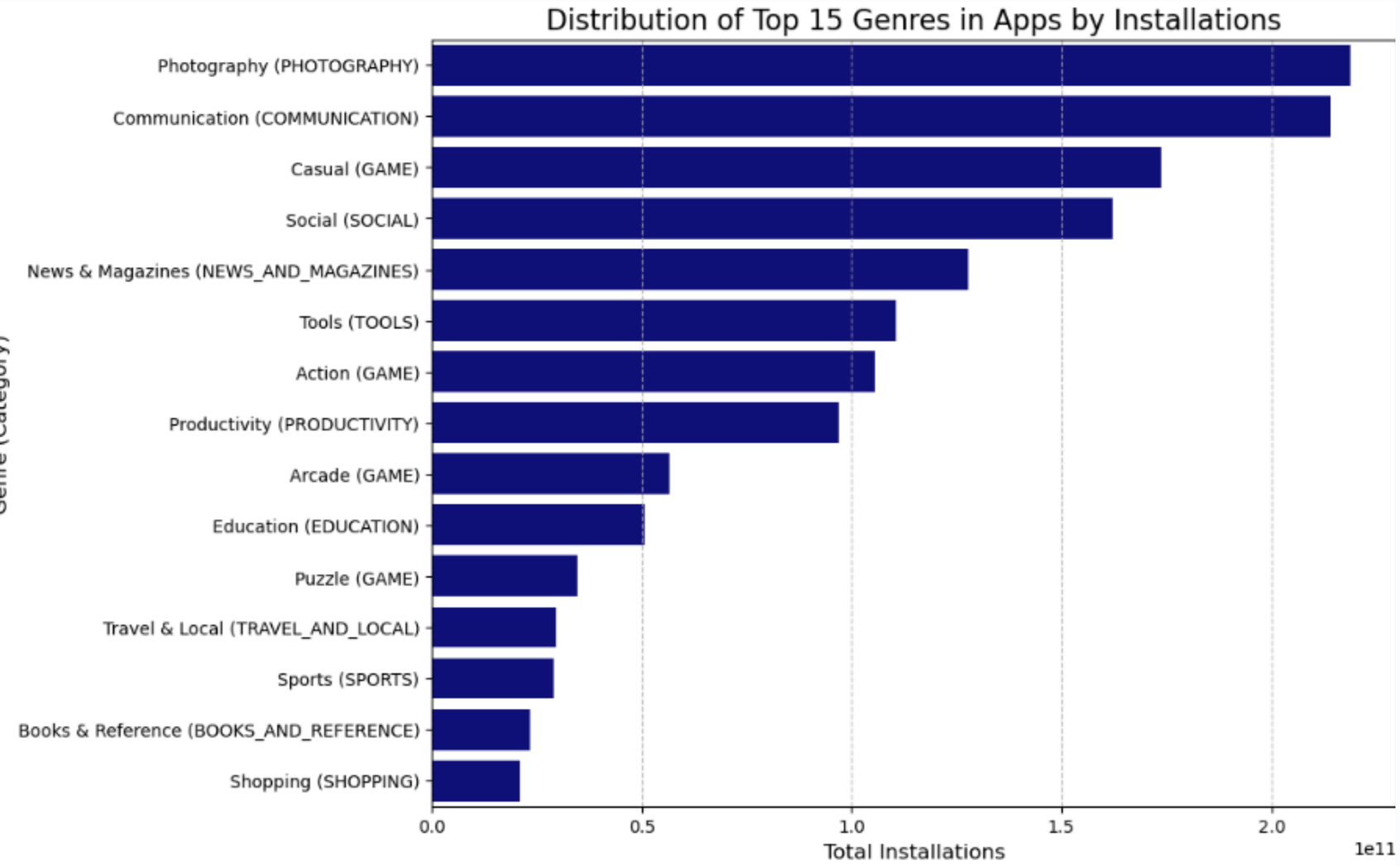
Varied User Experiences : The quality of games can vary significantly, with some functioning perfectly and others having technical issues, resulting in extreme reviews.

High Expectations : Game users often have higher expectations. If a game fails to meet these, it may receive very negative reviews; conversely, highly engaging games can generate very positive reviews

In summary, the GAME category elicits more extreme reactions compared to others, with users being either very satisfied or very disappointed. To a lesser extent, this pattern is also observed in the News and Magazine category.

Data visualization

Top 15 Genre

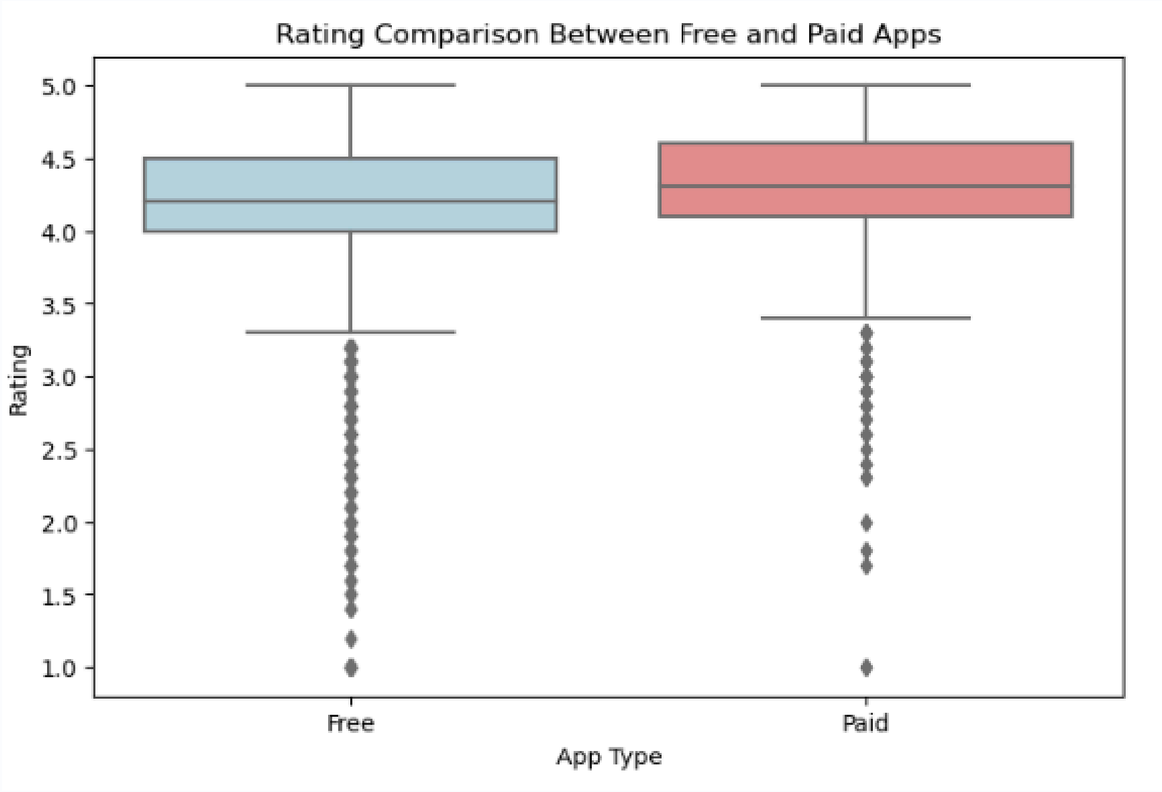


The 'Game' category appears to be the most popular, as evidenced by the frequent occurrence of the string 'Game' in parentheses.
However, in terms of genre, the most popular applications belong to the `photography and communication categories.`

Data visualization

Free App VS Paid App

```
# Boxplot to compare app ratings by type
plt.figure(figsize=(8, 5))
sns.boxplot(data=df, x='Type', y='Rating', palette=['lightblue', 'lightcoral'])

plt.title('Rating Comparison Between Free and Paid Apps')
plt.xlabel('App Type')
plt.ylabel('Rating')
plt.show()
```



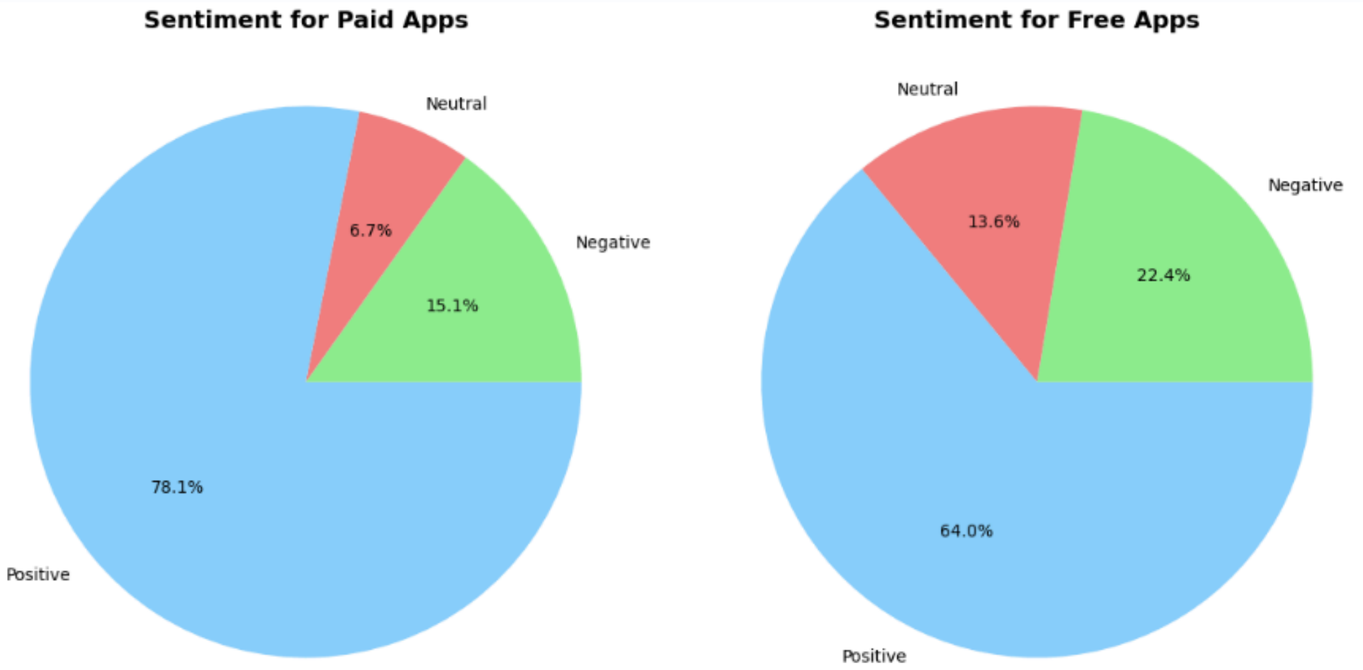
The variability of the rating distribution is similar for both categories (Free and Paid).

Paid apps have slightly higher ratings than Free apps.
The dense presence of outliers in Free apps means they have lower ratings than Paid apps.

Data visualization

Free App VS Paid App (Sentiment)

```
sentiment_percentage = merged_df.groupby(['Type', 'Sentiment']).size().unstack(fill_value=0)
sentiment_percentage = sentiment_percentage.apply(lambda x: x / x.sum() * 100, axis=1)

# Split data for Paid and Free apps
paid_sentiment = sentiment_percentage.loc['Paid']
free_sentiment = sentiment_percentage.loc['Free']

# Create pie charts
fig, axes = plt.subplots(1, 2, figsize=(12, 6))

# Pie chart for Paid apps
axes[0].pie(paid_sentiment, labels=paid_sentiment.index, autopct='%1.1f%%',
            colors=['lightgreen', 'lightcoral', 'lightskyblue'])
axes[0].set_title('Sentiment for Paid Apps', fontsize=14, fontweight='bold')
axes[0].set_ylabel('', fontsize=12, fontweight='bold')

# Pie chart for Free apps
axes[1].pie(free_sentiment, labels=free_sentiment.index, autopct='%1.1f%%',
            colors=['lightgreen', 'lightcoral', 'lightskyblue'])
axes[1].set_title('Sentiment for Free Apps', fontsize=14, fontweight='bold')
axes[1].set_ylabel('', fontsize=12, fontweight='bold')

# Bold axis labels
for ax in axes:
    for label in ax.get_xticklabels() + ax.get_yticklabels():
        label.set_fontweight('bold')

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



The paid apps tend to have more positive sentiment

Textual analysis

Reviews

```python
# Set Pandas to display the full content of the 'Translated_Review' column
pd.set_option('display.max_colwidth', None)

# Assume 'top_15_genres_with_category' is a list of genres
top_15_genres_with_category = [
    'Photography', 'Communication', 'Casual', 'Social', 'News & Magazines',
    'Tools', 'Action', 'Productivity', 'Arcade', 'Education',
    'Travel', 'Shopping', 'Entertainment', 'Health & Fitness', 'Lifestyle'
]

# Filter the dataframe Gr to get only the apps that belong to the specified genres
app_in_top_genres = df[df['Genres'].isin(top_15_genres_with_category)]['App']
gr_filtrato = Gr[Gr['App'].isin(app_in_top_genres)].copy()  # Create a copy to avoid the warning

# Add a column that determines if the comment is positive, negative, or neutral
gr_filtrato['Sentiment_Type'] = gr_filtrato['Sentiment_Polarity'].apply(
    lambda x: 'positive' if x > 0 else 'negative' if x < 0 else 'neutral'
)

# Dictionary to store comments for each genre
commenti_per_genere = {}

# Extract 2 positive and 2 negative comments for each genre
for genere in top_15_genres_with_category:
    # Filter apps belonging to the specific genre
    apps_in_genere = df[df['Genres'] == genere]['App']
    gr_genere = gr_filtrato[gr_filtrato['App'].isin(apps_in_genere)]

    # Separate positive and negative comments
    commenti_positivi = gr_genere[gr_genere['Sentiment_Type'] == 'positive'].head(2)
    commenti_negativi = gr_genere[gr_genere['Sentiment_Type'] == 'negative'].head(2)

    # Save comments in the dictionary
    commenti_per_genere[genere] = {
        'Positive_Comments': commenti_positivi[['App', 'Translated_Review']],
        'Negative_Comments': commenti_negativi[['App', 'Translated_Review']],
    }

# Display the results
for genere, commenti in commenti_per_genere.items():
    print(f"Genre: {genere}")
    print("Positive:")
    print(commenti['Positive_Comments'])
    print("Negative:")
    print(commenti['Negative_Comments'])
    print("\n")
```

The complete output of this code is available in the notebook.

What penalises the evaluation of an app?

- Continuous crashes and slow execution
- Problems with updates
- Difficulties in use and unclear interface
- Invasive Ads
- Lack of functionality

What improves evaluation?

- Free features
- Regular updates
- Error handilng and performance management
- User support:

🎯 Strategy

To build a successful app, pricing alone is not decisive.
Our analysis shows that install numbers and user ratings vary widely by category and genre, but price has little influence on sentiment.
Instead, **users value stability, usability, and ongoing support.**
Key success factors include:

- Performance and Stability
- Fast Issue Resolution
- Intuitive Interface
- Regular Updates
- Active Review Monitoring
- Good Value in Paid Apps
- Cross-Device Compatibility
- Balanced Monetization in Free Apps
- Valuable Free Features to Promote Upgrades
- Effective Customer Support
- Customization Options

The choice of category also plays a strategic role:

- Entering a highly competitive category (e.g. Games, Communication) offers greater visibility and user volume, but also requires more investment to stand out.
- Targeting a niche or less saturated category may reduce exposure but increases the chances of dominating a specific segment.

💡***Ultimately, user experience remains the strongest driver of ratings and retention, regardless of app type or price***.