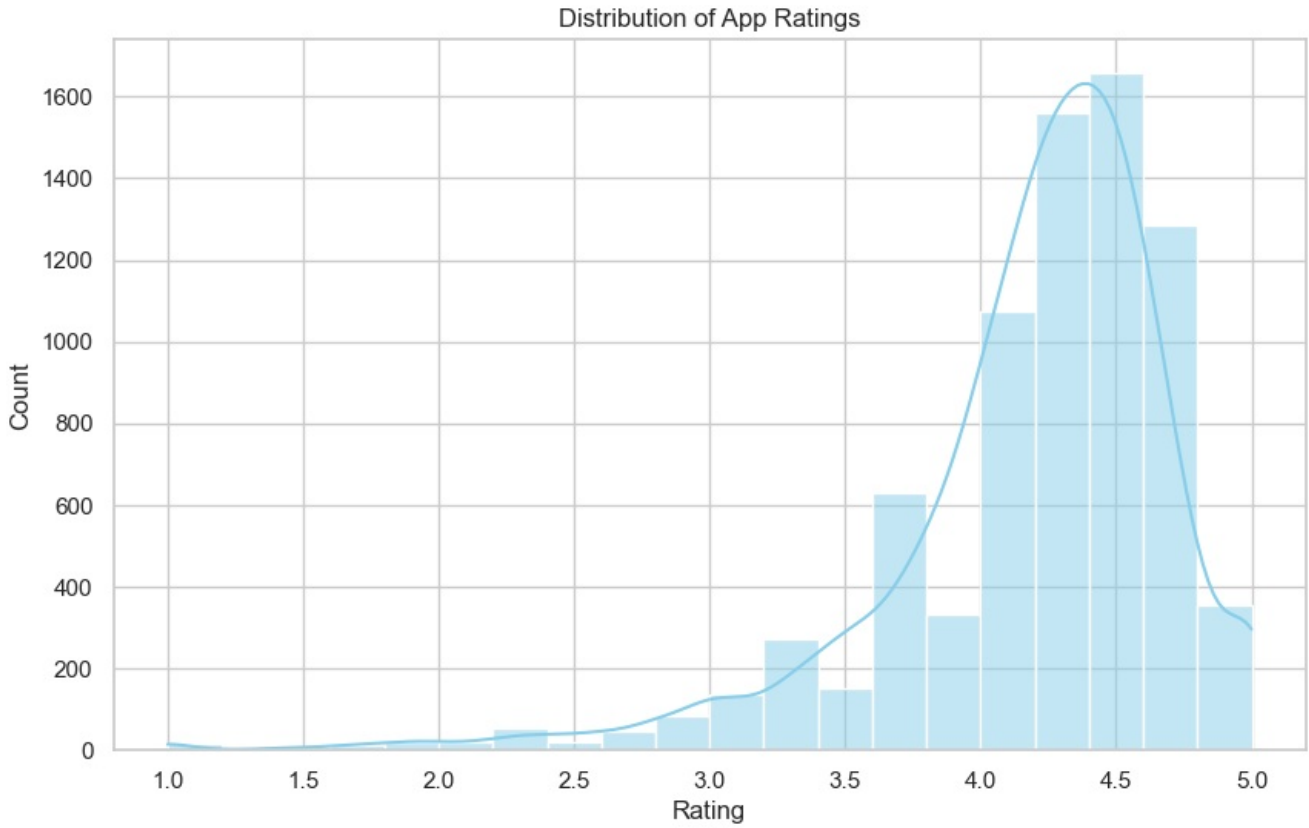
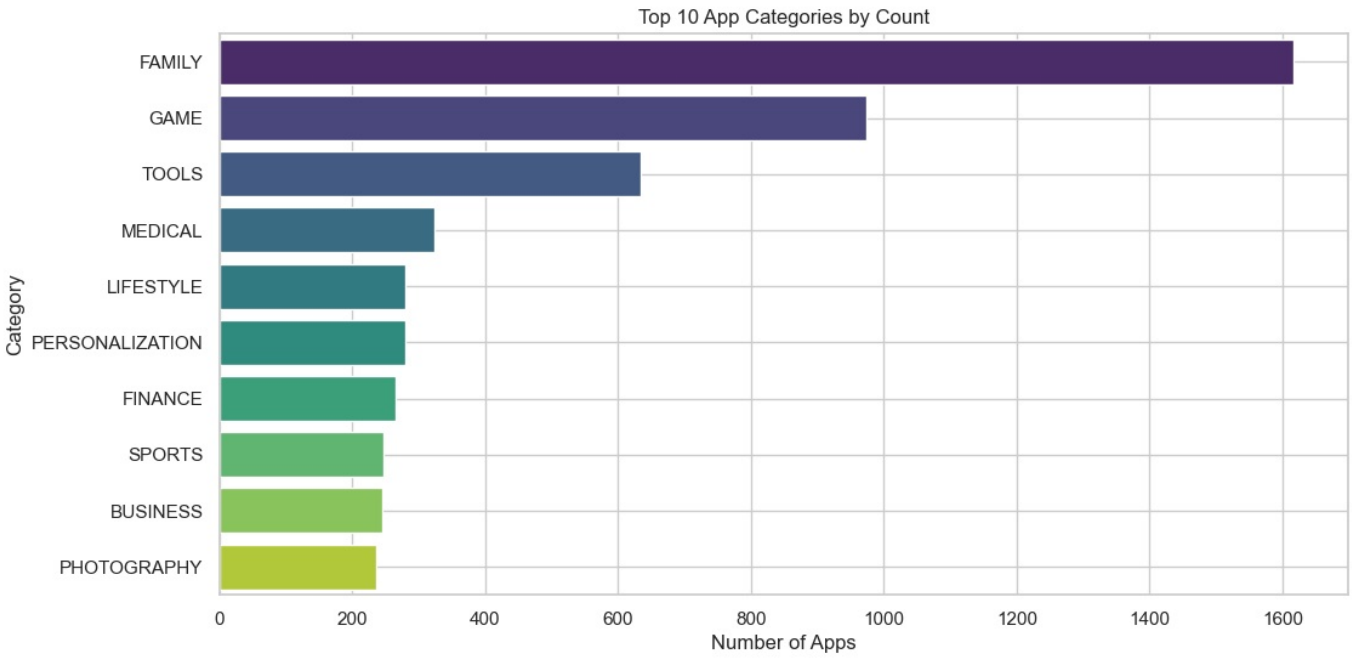
In [5]:
```
# Google Play Store Apps Analysis and Sentiment Project

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from textblob import TextBlob
import warnings
warnings.filterwarnings("ignore")
sns.set(style="whitegrid")
```

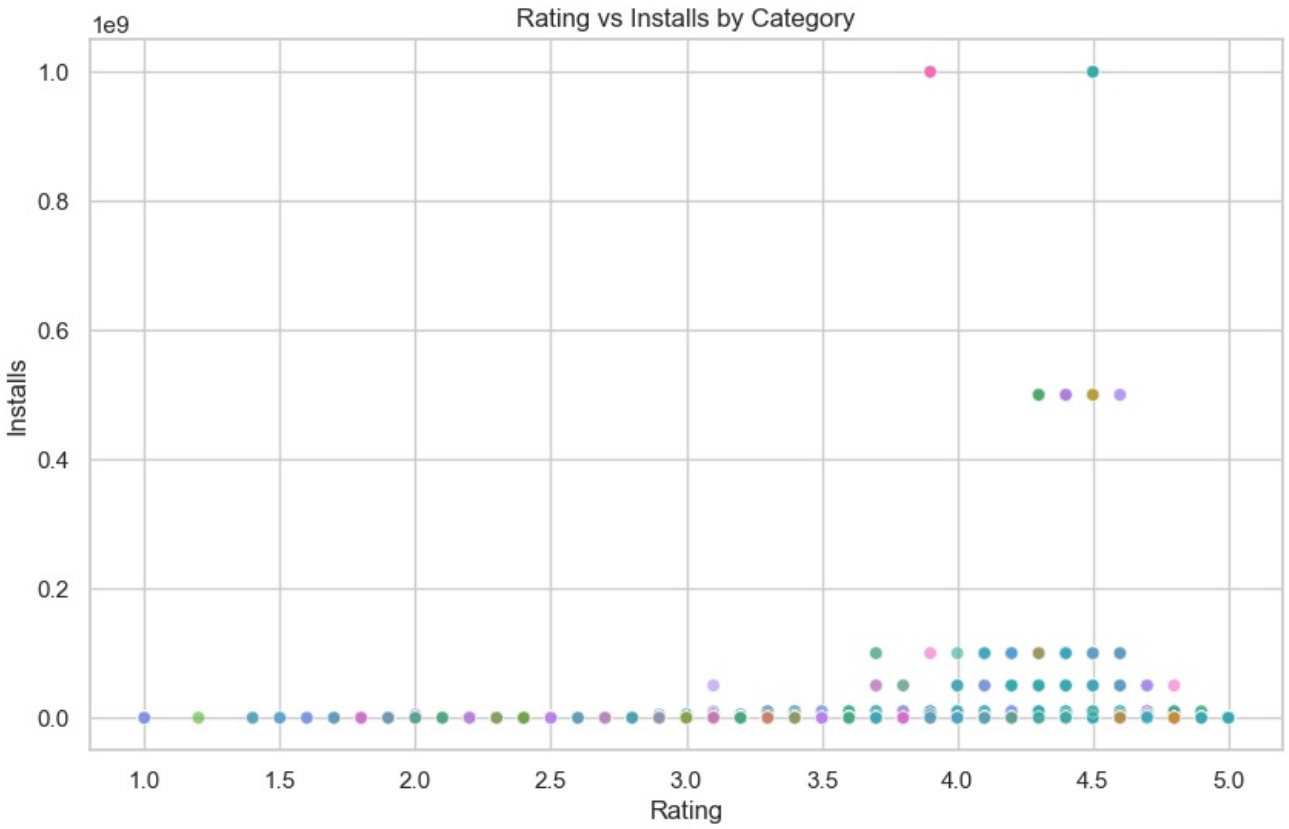
In [8]:
```
apps_df = pd.read_csv("C:\\Users\\HP\\Downloads\\googleplaystore.csv")
reviews_df = pd.read_csv("C:\\Users\\HP\\Downloads\\googleplaystore_user_reviews.csv")

#Data Cleaning for apps_df
apps_df.drop(index=10472, inplace=True)
apps_df['Reviews'] = apps_df['Reviews'].astype(str).str.replace('M', '').str.replace('True', '1').str.replace('
apps_df['Reviews'] = pd.to_numeric(apps_df['Reviews'], errors='coerce')
apps_df['Installs'] = apps_df['Installs'].str.replace('+', '').str.replace(',', '')
apps_df['Installs'] = pd.to_numeric(apps_df['Installs'], errors='coerce')
apps_df['Price'] = apps_df['Price'].str.replace('$', '')
apps_df['Price'] = pd.to_numeric(apps_df['Price'], errors='coerce')

# Convert Size to float (in MB)
def convert_size(size):
    if 'M' in size:
        return float(size.replace('M', ''))
    elif 'k' in size:
        return float(size.replace('k', '')) / 1024
    else:
        return np.nan

apps_df['Size'] = apps_df['Size'].astype(str).apply(convert_size)
apps_df.dropna(subset=['Rating', 'Reviews', 'Installs', 'Price', 'Size'], inplace=True)
```

In [10]:
```
# Cleaning reviews_df and Sentiment Analysis
reviews_df.dropna(subset=['Translated_Review'], inplace=True)
reviews_df['Sentiment_Polarity'] = reviews_df['Translated_Review'].apply(lambda x: TextBlob(x).sentiment.polari
reviews_df['Sentiment_Label'] = reviews_df['Sentiment_Polarity'].apply(lambda x: 'Positive' if x > 0 else ('Nega
```

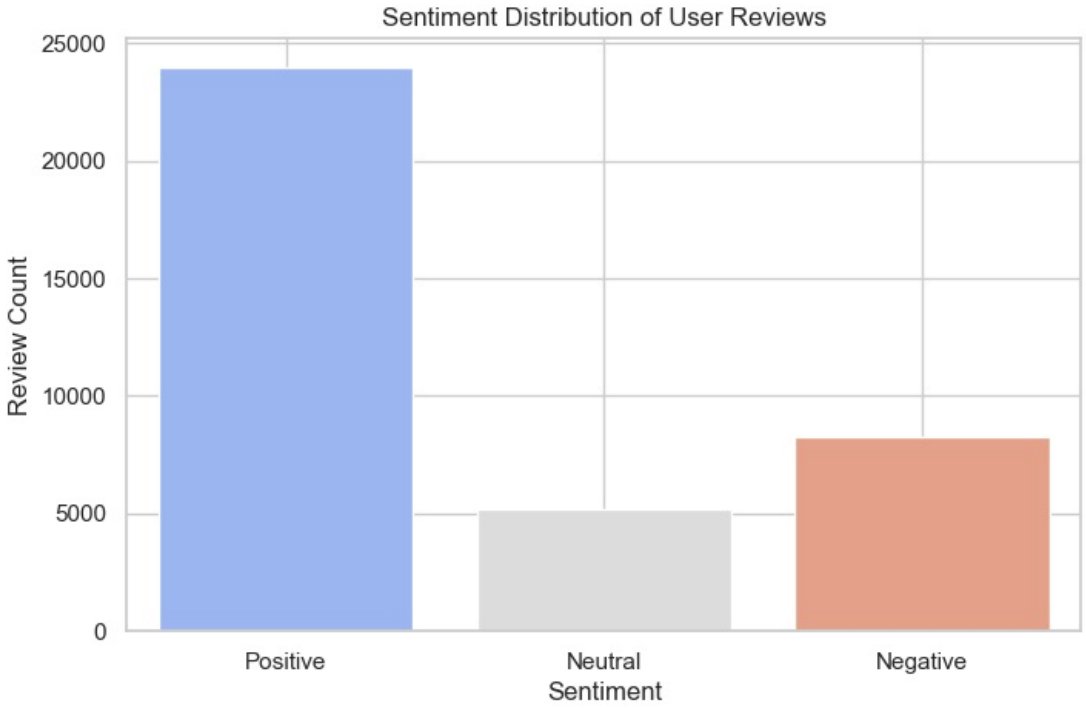
In [11]:
```
#EDA Plots
plt.figure(figsize=(10, 6))
sns.histplot(apps_df['Rating'], bins=20, kde=True, color='skyblue')
plt.title('Distribution of App Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.grid(True)
plt.show()

plt.figure(figsize=(12, 6))
top_categories = apps_df['Category'].value_counts().head(10)
sns.barplot(x=top_categories.values, y=top_categories.index, palette='viridis')
plt.title('Top 10 App Categories by Count')
plt.xlabel('Number of Apps')
plt.ylabel('Category')
plt.grid(True)
plt.show()

plt.figure(figsize=(10, 6))
sns.scatterplot(data=apps_df, x='Rating', y='Installs', hue='Category', alpha=0.6, legend=False)
plt.title('Rating vs Installs by Category')
plt.xlabel('Rating')
plt.ylabel('Installs')
plt.grid(True)
plt.show()
```


Distribution of App Ratings
Count
Rating
0
200
400
600
800
1000
1200
1400
1600
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


Top 10 App Categories by Count
Category
FAMILY
GAME
TOOLS
MEDICAL
LIFESTYLE
PERSONALIZATION
FINANCE
SPORTS
BUSINESS
PHOTOGRAPHY
0
200
400
600
800
1000
1200
1400
1600
Number of Apps



In [12]:
```
# Sentiment Distribution Plot
plt.figure(figsize=(8, 5))
sns.countplot(x='Sentiment_Label', data=reviews_df, palette='coolwarm')
plt.title("Sentiment Distribution of User Reviews")
plt.xlabel("Sentiment")
plt.ylabel("Review Count")
plt.grid(True)
plt.show()
```



In [13]:
```
# Summary Stats
print("\nTop Categories by Number of Apps:\n", apps_df['Category'].value_counts().head())
print("\nTop Categories by Total Installs:\n", apps_df.groupby('Category')['Installs'].sum().sort_values(ascend
print("\nTop Categories by Rating:\n", apps_df.groupby('Category')['Rating'].mean().sort_values(ascending=False
print("\nSentiment Counts:\n", reviews_df['Sentiment_Label'].value_counts())
```

```
Top Categories by Number of Apps:
 Category
FAMILY       1617
GAME          974
TOOLS         634
MEDICAL       324
LIFESTYLE     280
Name: count, dtype: int64

Top Categories by Total Installs:
 Category
GAME                   29874452717
FAMILY                  6798433580
COMMUNICATION           4941915530
NEWS_AND_MAGAZINES      4251900550
TOOLS                   3526053500
Name: Installs, dtype: int64

Top Categories by Rating:
 Category
EVENTS             4.478947
EDUCATION          4.387273
ART_AND_DESIGN     4.361017
PARENTING          4.347727
PERSONALIZATION    4.324286
Name: Rating, dtype: float64

Sentiment Counts:
 Sentiment_Label
Positive    23997
Negative     8272
Neutral      5158
Name: count, dtype: int64
```

In [ ]: