1 Unveiling the Android App Market

1.1 Google Play Store Data Analysis

This project aims to analyze the **Google Play Store** app data to identify key trends, understand user preferences, and explore market dynamics. The analysis focuses on the following objectives:

- Correct data types, handle missing values, and eliminate duplicates to ensure data accuracy.
- Examine the distribution of app categories and investigate key metrics such as ratings, size, popularity, and pricing.
- Analyze user reviews to gauge sentiment, focusing on identifying positive, negative, and neutral feedback.
- Generate both static and interactive plots to communicate insights effectively and make the analysis more intuitive.
- Based on the analysis, provide data-driven suggestions for optimizing app features, pricing strategies, and user engagement.

Dataset Link: The dataset used for this analysis can be downloaded from Kaggle:

https://www.kaggle.com/datasets/utshabkumarghosh/android-app-market-on-google-play

1.1.1 Dataset Loading & Preparation

[2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
```

[3]:
```
apps = pd.read_csv("apps.csv")
```

[4]:
```
review = pd.read_csv("user_reviews.csv")
```

1.1.2 Data Exploration

[6]:
```
review.head(3)
```

[6]:
```
                  App                                 Translated_Review  \
0  10 Best Foods for You  I like eat delicious food. That's I'm cooking …
1  10 Best Foods for You    This help eating healthy exercise regular basis
2  10 Best Foods for You                                              NaN

  Sentiment  Sentiment_Polarity  Sentiment_Subjectivity
0  Positive                1.00                0.533333
1  Positive                0.25                0.288462
2       NaN                 NaN                     NaN
```

[7]:
```
apps.head(3)
```

[7]:
```
   Unnamed: 0                                                App  \
0           0     Photo Editor & Candy Camera & Grid & ScrapBook
1           1                                 Coloring book moana
2           2  U Launcher Lite - FREE Live Cool Themes, Hide …

         Category  Rating  Reviews  Size    Installs  Type Price  \
0  ART_AND_DESIGN     4.1      159  19.0     10,000+  Free     0
1  ART_AND_DESIGN     3.9      967  14.0    500,000+  Free     0
2  ART_AND_DESIGN     4.7    87510   8.7  5,000,000+  Free     0

  Content Rating                     Genres      Last Updated Current Ver  \
0       Everyone               Art & Design   January 7, 2018       1.0.0
1       Everyone  Art & Design;Pretend Play  January 15, 2018       2.0.0
2       Everyone               Art & Design    August 1, 2018       1.2.4

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
```

[8]:
```
apps.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9659 entries, 0 to 9658
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Unnamed: 0      9659 non-null   int64
 1   App             9659 non-null   object
 2   Category        9659 non-null   object
 3   Rating          8196 non-null   float64
 4   Reviews         9659 non-null   int64
 5   Size            8432 non-null   float64
 6   Installs        9659 non-null   object
 7   Type            9659 non-null   object
```

```
 8   Price           9659 non-null   object
 9   Content Rating  9659 non-null   object
 10  Genres          9659 non-null   object
 11  Last Updated    9659 non-null   object
 12  Current Ver     9651 non-null   object
 13  Android Ver     9657 non-null   object
dtypes: float64(2), int64(2), object(10)
memory usage: 1.0+ MB
```

[9]:
```
apps.shape
```

[9]:
```
(9659, 14)
```

[10]:
```
apps.duplicated().sum()
```

[10]:
```
0
```

[11]:
```
apps.isna().sum()
```

[11]:
```
Unnamed: 0          0
App                 0
Category            0
Rating           1463
Reviews             0
Size             1227
Installs            0
Type                0
Price               0
Content Rating      0
Genres              0
Last Updated        0
Current Ver         8
Android Ver         2
dtype: int64
```

[12]:
```
apps.describe()
```

[12]:

	Unnamed: 0	Rating	Reviews	Size
count	9659.000000	8196.000000	9.659000e+03	8432.000000
mean	5666.172896	4.173243	2.165926e+05	20.395327
std	3102.362863	0.536625	1.831320e+06	21.827509
min	0.000000	1.000000	0.000000e+00	0.000000
25%	3111.500000	4.000000	2.500000e+01	4.600000
50%	5814.000000	4.300000	9.670000e+02	12.000000
75%	8327.500000	4.500000	2.940100e+04	28.000000
max	10840.000000	5.000000	7.815831e+07	100.000000

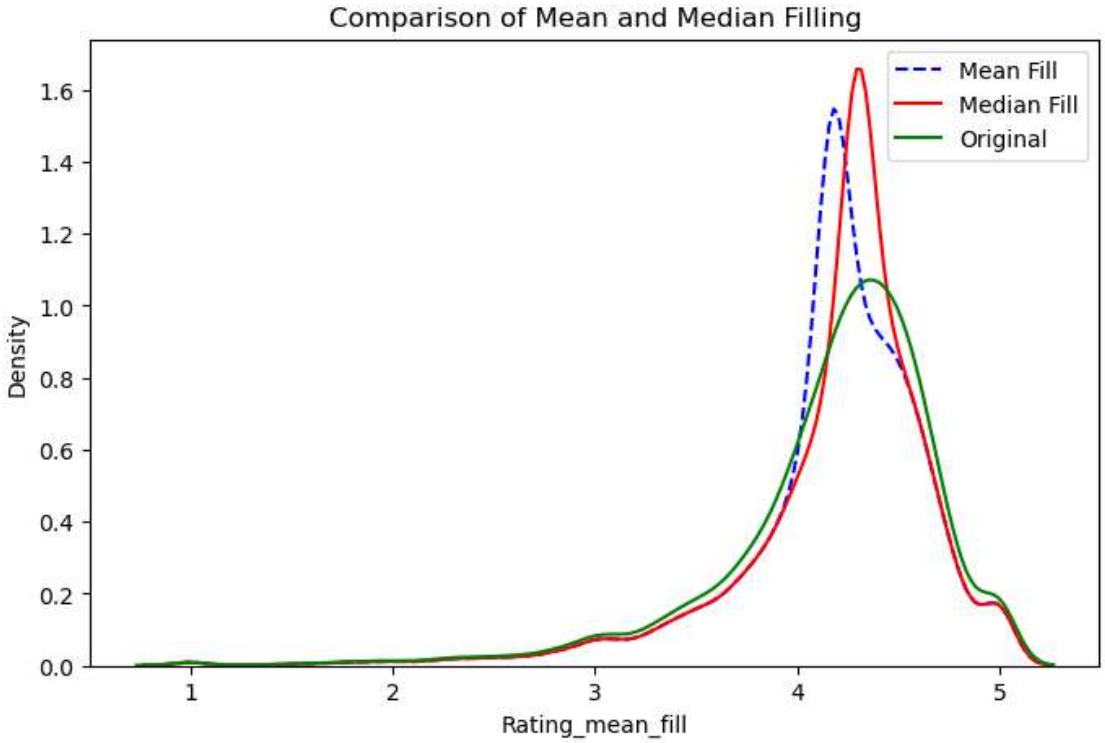
1.1.3 Data Cleaning

[14]:
```
replace_char = ['+', ',', '$']
Clean_Columns = ['Installs','Price']
for col in Clean_Columns:
    for char in replace_char:
        apps[col]=  apps[col].astype(str).str.replace(char,'')
    apps[col] = pd.to_numeric(apps[col])
```

[15]:
```
apps['Last Updated'] = pd.to_datetime(apps['Last Updated'])
```

[17]:
```
# Fill with mean
apps['Rating_mean_fill'] = apps['Rating'].fillna(apps['Rating'].mean())
# Fill with median
apps['Rating_median_fill'] = apps['Rating'].fillna(apps['Rating'].median())
```

[19]:
```
# Comparison of Mean and Median Filling
plt.figure(figsize=(8, 5))
sns.kdeplot(apps['Rating_mean_fill'], label='Mean Fill', color='blue',␣
 ↪linestyle='--')
sns.kdeplot(apps['Rating_median_fill'], label='Median Fill', color='red')
sns.kdeplot(apps['Rating'], label='Original', color='green')
plt.title('Comparison of Mean and Median Filling')
plt.legend()
plt.show()
```



[20]:
```
apps['Rating'] = apps['Rating'].fillna(apps['Rating'].median())
```

[25]:
```
apps.rename(columns={'Unnamed: 0':'Index_id'},inplace=True)
```

[32]:
```
apps.drop('Rating_mean_fill',axis=1,inplace=True)
apps.drop('Rating_median_fill',axis=1,inplace=True)
```

[34]:
```
apps.describe()
```

[34]:

	Index_id	Rating	Reviews	Size	Installs
count	9659.000000	9659.000000	9.659000e+03	8432.000000	9.659000e+03
mean	5666.172896	4.192442	2.165926e+05	20.395327	7.777507e+06
min	0.000000	1.000000	0.000000e+00	0.000000	0.000000e+00
25%	3111.500000	4.000000	2.500000e+01	4.600000	1.000000e+03
50%	5814.000000	4.300000	9.670000e+02	12.000000	1.000000e+05
75%	8327.500000	4.500000	2.940100e+04	28.000000	1.000000e+06
max	10840.000000	5.000000	7.815831e+07	100.000000	1.000000e+09
std	3102.362863	0.496397	1.831320e+06	21.827509	5.375828e+07

	Price	Last Updated
count	9659.000000	9659
mean	1.099299	2017-10-30 19:34:02.074748928
min	0.000000	2010-05-21 00:00:00
25%	0.000000	2017-08-05 12:00:00
50%	0.000000	2018-05-04 00:00:00
75%	0.000000	2018-07-17 00:00:00
max	400.000000	2018-08-08 00:00:00
std	16.852152	NaN

1.1.4 Exploratory Data Analysis

[36]:
```
apps.head(2)
```

[36]:

	Index_id	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19.0	10000	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
1	1	Coloring book moana	ART_AND_DESIGN	3.9	967	14.0	500000	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4.0.3 and up

[39]:
```
# Create a revenue estimate (for paid apps)
apps['Revenue'] = apps['Installs'] * apps['Price']
```

[42]:
```
# Top 10 most expensive apps
apps[['App', 'Category', 'Rating', 'Size', 'Price','Installs','Revenue']].
 ↪nlargest(10, 'Price')
```

[42]:

	App	Category	Rating	Size	Price	Installs	Revenue
3469	I'm Rich - Trump Edition	LIFESTYLE	3.6	7.3	400.00	10000	4000000.0
3327	most expensive app (H)	FAMILY	4.3	1.5	399.99	100	39999.0
3465	I'm rich	LIFESTYLE	3.8	26.0	399.99	10000	3999900.0
4396	I am rich	LIFESTYLE	3.8	1.8	399.99	100000	39999000.0
4398	I am Rich Plus	FAMILY	4.0	8.7	399.99	10000	3999900.0
4400	I Am Rich Premium	FINANCE	4.1	4.7	399.99	50000	19999500.0
4402	I am Rich!	FINANCE	3.8	22.0	399.99	1000	399990.0
4403	I am rich(premium)	FINANCE	3.5	1.0	399.99	5000	1999950.0
4406	I Am Rich Pro	FAMILY	4.4	2.7	399.99	5000	1999950.0
4408	I am rich (Most expensive app)	FINANCE	4.1	2.7	399.99	1000	399990.0

[46]:
```
# Apps with extreme installs
apps[['App','Installs']].max()
```

[46]:
```
App           Football Wallpapers 4K | Full HD Backgrounds
Installs                                        1000000000
dtype: object
```

[49]:
```
# Top 5 categories
apps.groupby('Category')['Category'].value_counts().nlargest(5)
```

[49]:
```
Category
FAMILY      1832
GAME         959
TOOLS        827
BUSINESS     420
MEDICAL      395
```

```
Name: count, dtype: int64
```

[51]:
```
# the maximum frequency rating
apps['Rating'].mode()[0]
```

[51]:
```
4.3
```

[53]:
```
# Compare average ratings of free vs. paid apps
print(apps.groupby('Type')['Rating'].agg(['mean', 'median', 'count']))
```

```
          mean  median  count
Type
Free  4.186050     4.3   8903
Paid  4.267725     4.3    756
```

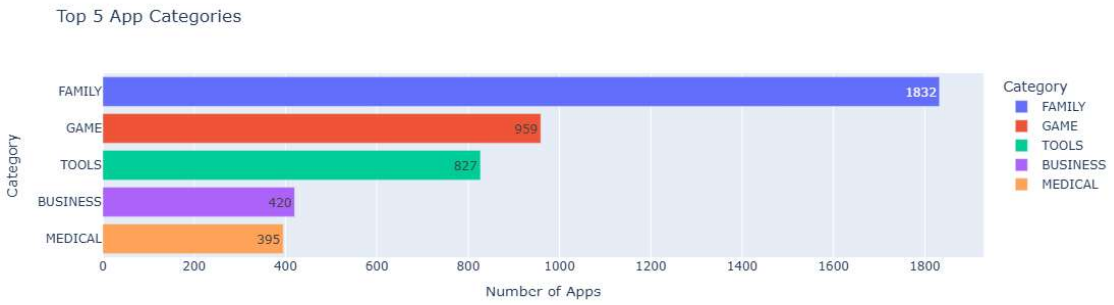
[55]:
```
# Correlation matrix
apps[['Rating', 'Reviews','Size','Installs', 'Price']].corr()
```

[55]:

	Rating	Reviews	Size	Installs	Price
Rating	1.000000	0.050207	0.045546	0.034307	-0.018662
Reviews	0.050207	1.000000	0.179321	0.625165	-0.007598
Size	0.045546	0.179321	1.000000	0.134291	-0.022434
Installs	0.034307	0.625165	0.134291	1.000000	-0.009405
Price	-0.018662	-0.007598	-0.022434	-0.009405	1.000000

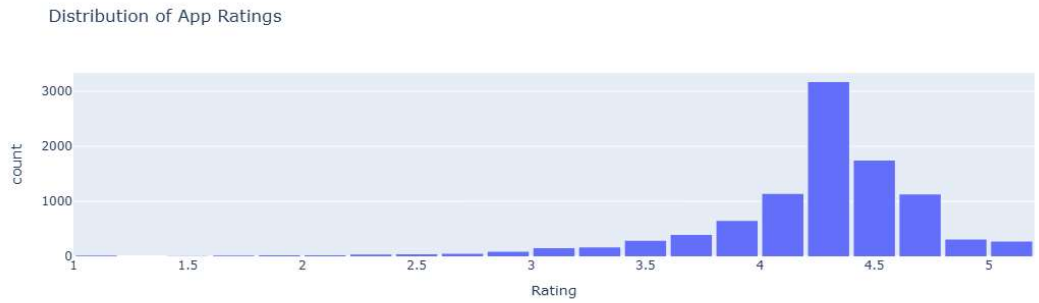
1.1.5 Data Visualization

[58]:
```
# App Distribution by Category
fig = px.bar(apps['Category'].value_counts().head(5).reset_index(),
             x='count', y='Category',
             title='Top 5 App Categories',
             labels={'count': 'Number of Apps', 'Category': 'Category'},
             color='Category', text_auto=True)
fig.show()
```



[59]:
```
# Free vs. Paid App Distribution
fig = px.pie(
    apps, names='Type',
    title='Free vs. Paid Apps',
    hole=0.3,
    color='Type',
    color_discrete_map={'Free': '#00CC96', 'Paid': '#EF553B'}
)
fig.update_traces(textinfo='percent+label')
fig.show()
```

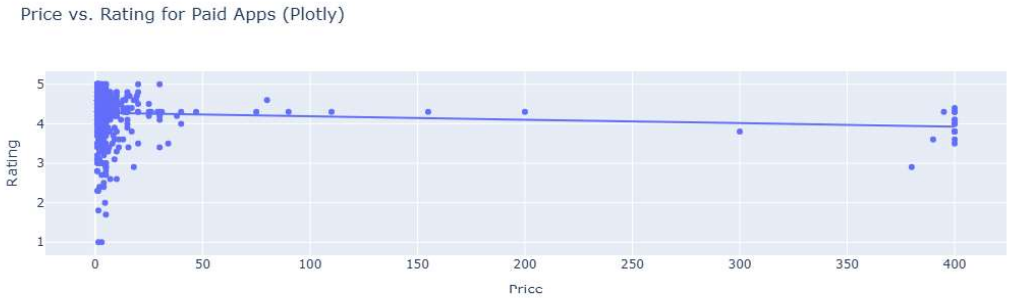
[60]:
```
# Distribution of App Ratings
fig = px.histogram(apps, x='Rating', nbins=30,title='Distribution of App␣
↪Ratings')
fig.update_layout(bargap=0.1)
fig.show()
```



[61]:
```
# Which App Categories Have the Highest Competition (Oversaturation)?
fig = px.scatter(
```

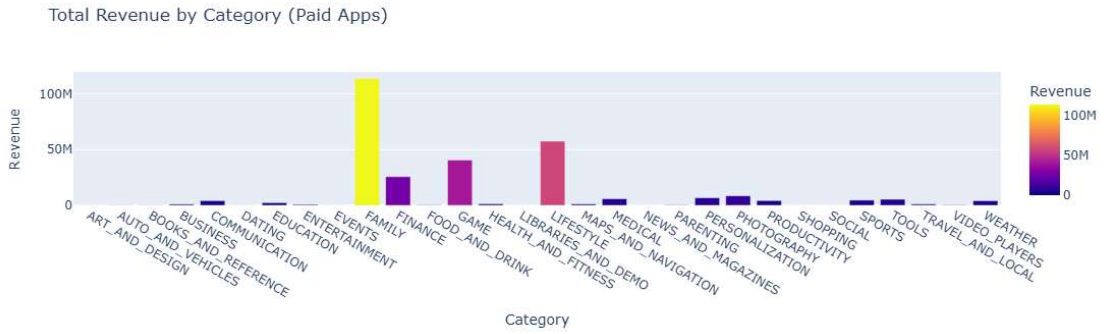
```
    apps.groupby('Category').agg({'App': 'count', 'Rating': 'mean'}).
 ↪reset_index(),
    x='App', y='Rating', size='App', color='Category',
    title='Category Competition vs. Average Rating',
    labels={'App': 'Number of Apps', 'Rating': 'Avg Rating'}
)
fig.show()
```

[62]:
```
# Are Higher-Priced Apps Rated Better (Quality Perception)?
fig = px.scatter(
    apps[apps['Price'] > 0],
    x='Price', y='Rating', trendline="ols",
    title='Price vs. Rating for Paid Apps (Plotly)'
)
fig.show()
```

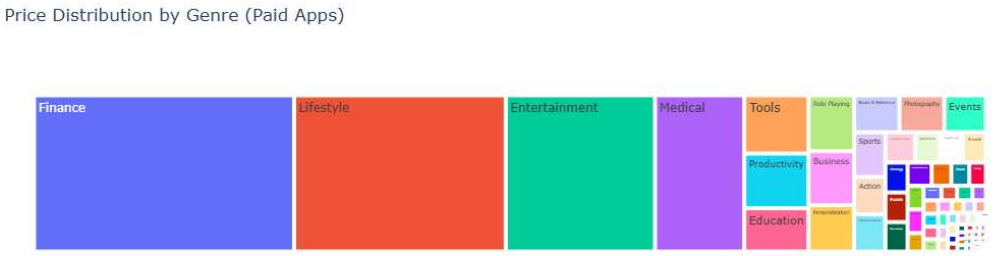


[63]:
```
# Which Categories Generate the Most Revenue?
fig = px.bar(
    apps[apps['Type'] == 'Paid'].groupby('Category')['Revenue'].sum().
 ↪reset_index(),
```

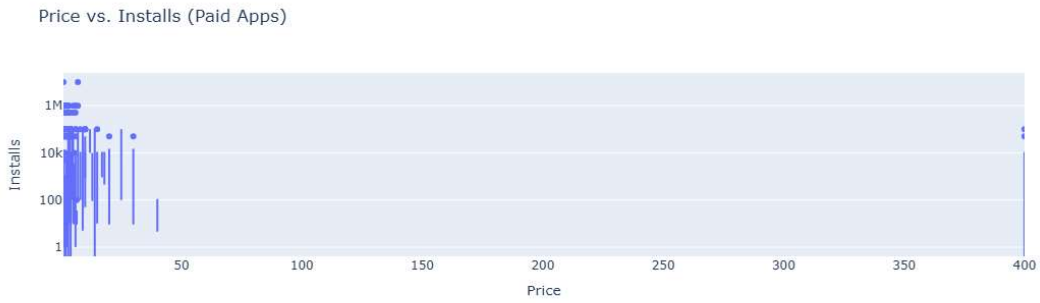
```
    x='Category', y='Revenue', color='Revenue',
    title='Total Revenue by Category (Paid Apps)'
)
fig.show()
```



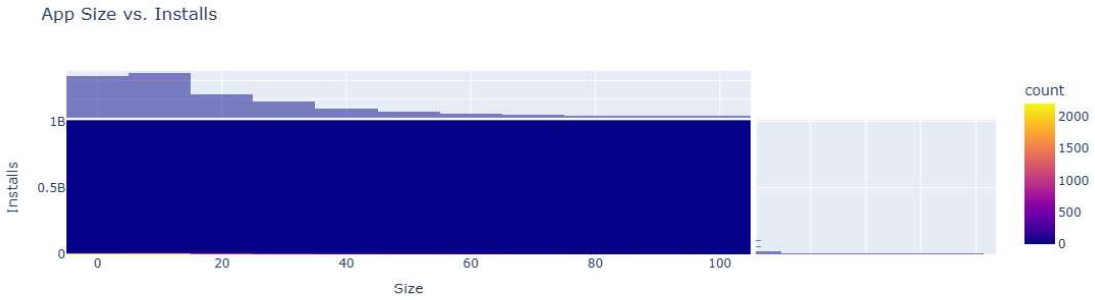
[64]:
```
#  Do Certain Genres Have Higher Price Potential?
fig = px.treemap(
    apps[apps['Type'] == 'Paid'],
    path=['Genres'], values='Price',
    title='Price Distribution by Genre (Paid Apps)'
)
fig.show()
```



[65]:
```
# Price Impact on Installs
paid_apps = apps[apps['Price'] > 0]

fig = px.box(paid_apps, x='Price', y='Installs', log_y=True, title='Price vs.␣
 ↪Installs (Paid Apps)')
fig.show()
```



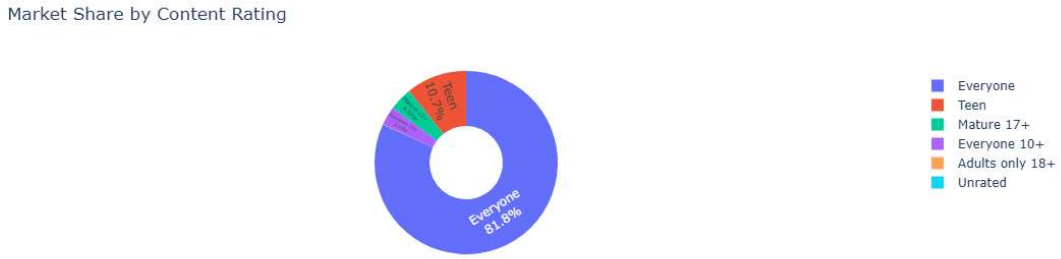
[66]:
```
# App Size Impact Installs
fig = px.density_heatmap(
    apps, x='Size', y='Installs', nbinsx=20,
    title='App Size vs. Installs',
    marginal_x="histogram", marginal_y="histogram"
)
fig.show()
```



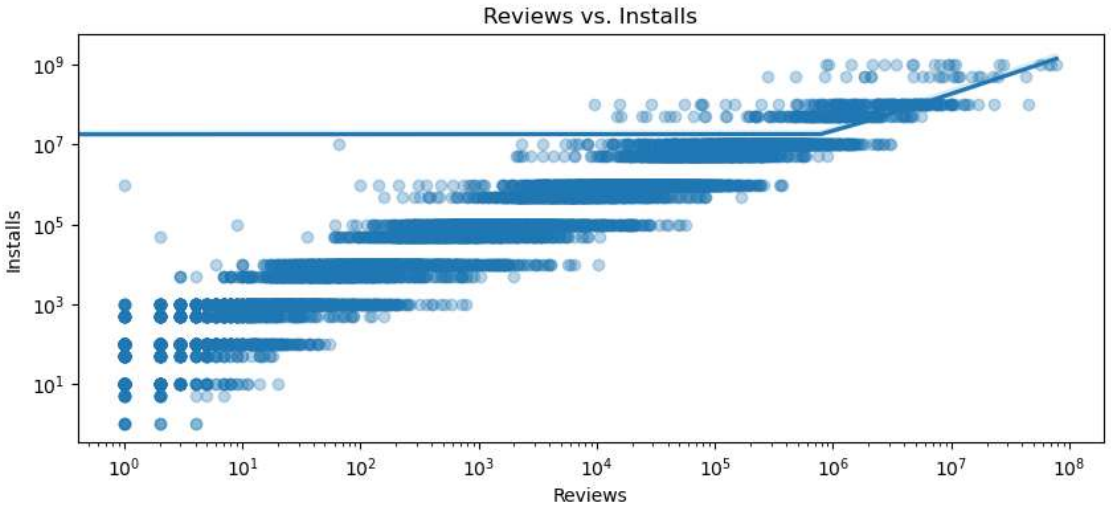
[67]:
```
# Are Frequent Updates Correlated with Higher Ratings?
apps['Days Since Update'] = (pd.Timestamp.now() - apps['Last Updated']).dt.days

fig = px.scatter(
    apps, x='Days Since Update', y='Rating',
    trendline="lowess", title='Update Freshness vs. Rating'
)
fig.show()
```

[68]:
```
# Are There Seasonal Trends in App Launches/Updates
apps['Month Updated'] = apps['Last Updated'].dt.month
fig = px.line(
    apps['Month Updated'].value_counts().sort_index().reset_index(),
    x='Month Updated', y='count',
    title='App Updates by Month (Seasonality)'
)
fig.show()
```

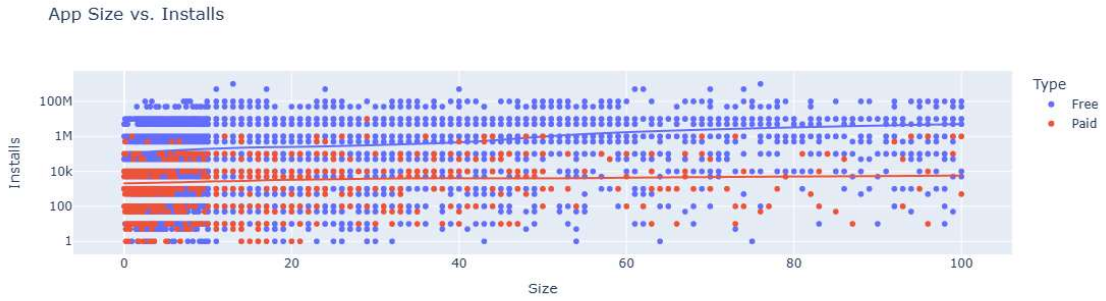
[70]:
```
# Target underserved age groups
fig = px.pie(
    apps, names='Content Rating',
    title='Market Share by Content Rating',
    hole=0.4
)
fig.update_traces(textposition='inside', textinfo='percent+label')
fig.show()
```



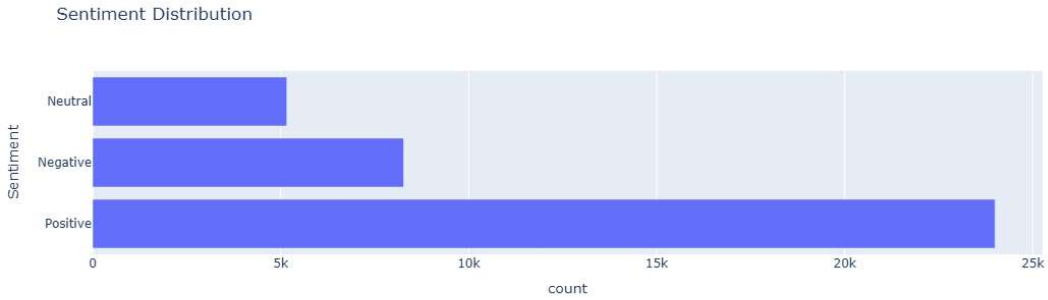
[98]:
```
plt.figure(figsize=(10, 4))
sns.regplot(data=apps, x='Reviews', y='Installs', scatter_kws={'alpha':0.3})
plt.xscale('log')
plt.yscale('log')
plt.title('Reviews vs. Installs')
plt.show()
```



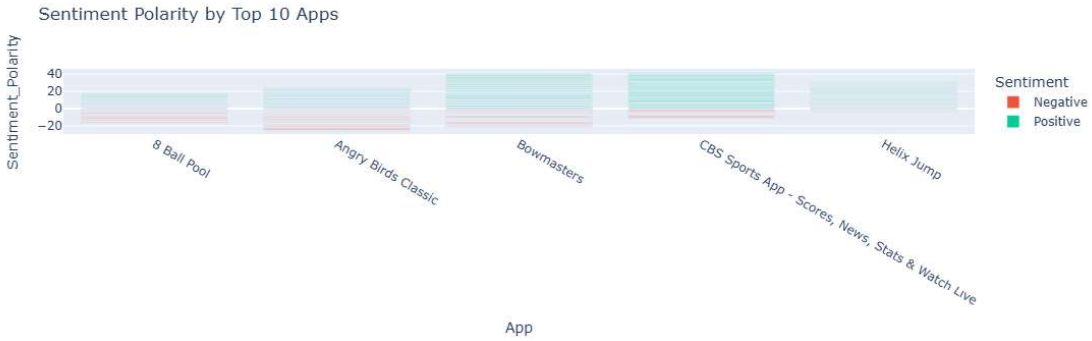
[75]:
```
# App Size vs. Installs
fig = px.scatter(
    apps, x='Size', y='Installs',
    title='App Size vs. Installs',
    trendline='lowess',
    color='Type',
    log_y=True
)
fig.show()
```



[76]:
```
# Sentiment Distribution
fig1 = px.bar(
    review['Sentiment'].value_counts().reset_index(),
    x='count', y='Sentiment',
    title='Sentiment Distribution'
)
fig1.show()
```



[81]:
```
# Get the top 5 apps based on the number of reviews
top_5_apps = review['App'].value_counts().nlargest(5).index
filtered_df = review[review['App'].isin(top_5_apps)]

# Sentiment Polarity by Top 10 Apps Plot
fig2 = px.bar(
    filtered_df, x='App', y='Sentiment_Polarity', color='Sentiment',
    title='Sentiment Polarity by Top 10 Apps')
fig2.show()
```


Sentiment Polarity by Top 10 Apps
Sentiment_Polarity
40
20
0
−20
8 Ball Pool
Angry Birds Classic
Bowmasters
CBS Sports App - Scores, News, Stats & Watch Live
Helix Jump
App
Sentiment
Negative
Positive