ABSTRACT

Now a days, it is becoming difficult for app developers to predict the number of installs of their app or to analyze what is the reach of their apps and what is the current market demand.

The goal of this project is to provide insights that will enable developers better understand what user wants and thus promote the applications. The dataset is chosen from Kaggle. It is of 10k Play Store apps for analyzing the Android market. This dataset provides information on several applications as well as user feedback.

TABLE OF CONTENTS

INTRODUCTION

The purpose of our research is to anticipate the amount of app installs based on app information and reviews. We hope that this project will assist app creators in predicting their number of installs, as well as investors looking for the next big thing. Companies may conduct beta focus groups, or app developers may receive feedback from beta testers and receive a set number of reviews. Developers and company managers might benefit from knowing the number of installs because it allows them to forecast earnings. The outcome of this study may demonstrate the relevance of app reviews in the market, as they may be one of the determining factors for the number of installs.

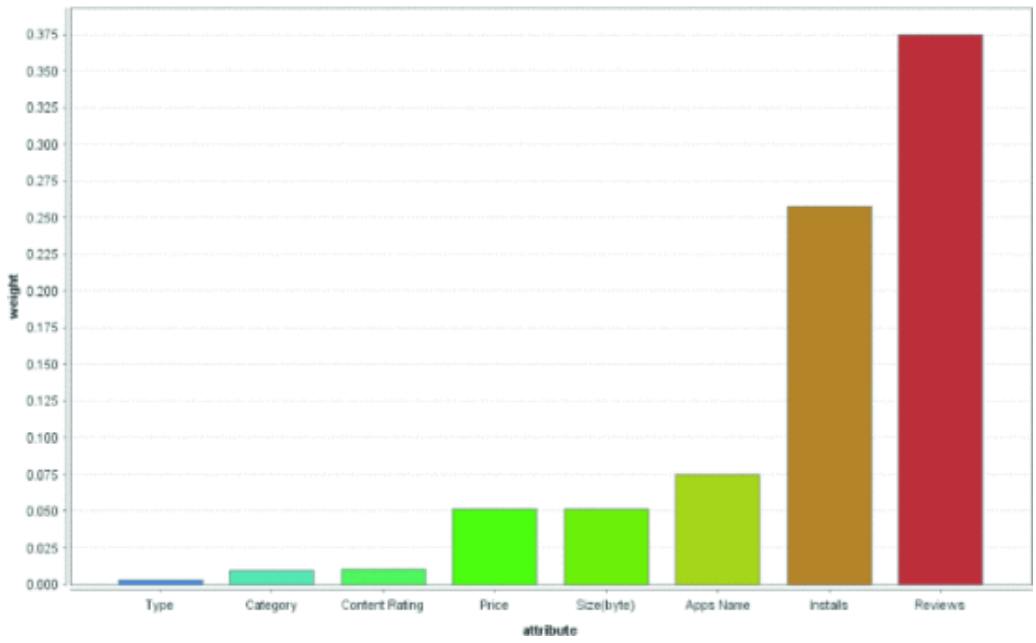
Motivation Behind the Project

While many public datasets give Apple App Store data, there are very few Google Play Store app datasets available anywhere on the web. Further investigation revealed that the iTunes App Store website has a beautifully indexed appendix-like structure to facilitate web scraping. Google Play Store, on the other hand, employs advanced modern-day techniques (such as dynamic page loading) that make scraping more difficult. Hence, we decided to take this dataset and tried to make it less complex in an organized manner accompanied by data visualization.

Previous Works

Paper 1: Apps Rating Classification on Play Store Using Gradient Boost Algorithm

In this paper they have used bar graph data visualization to predict which criteria is most influential in ranking the applications.

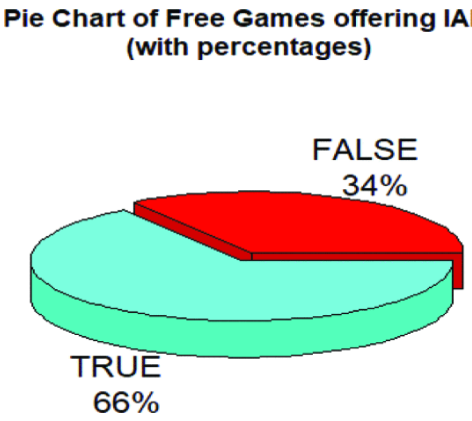


Paper 2: Data Scraping from Google Play Store and Visualization of its Content for Analytics

1. Free Games InApplicationPurchases (IAP)

In Google play store many games contain such items, products, credits etc. which get by performing actions which are not relevant to the game.

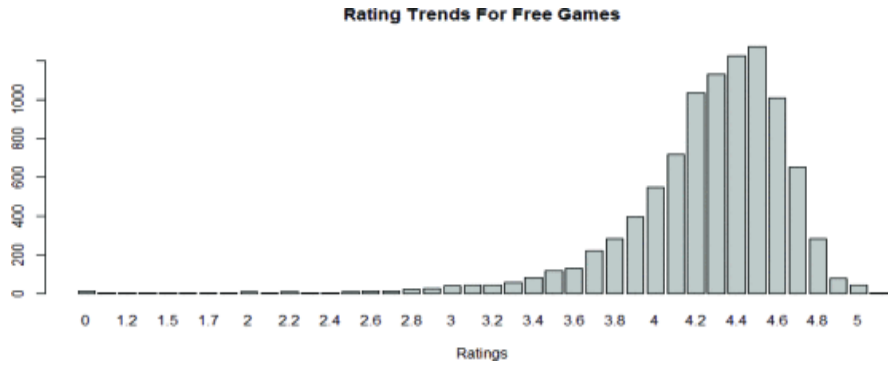
The pie3D chart shows that 66% of free games offer IAP and 34% free games did not offer IAP as shown.



2. Free Games Rating Value

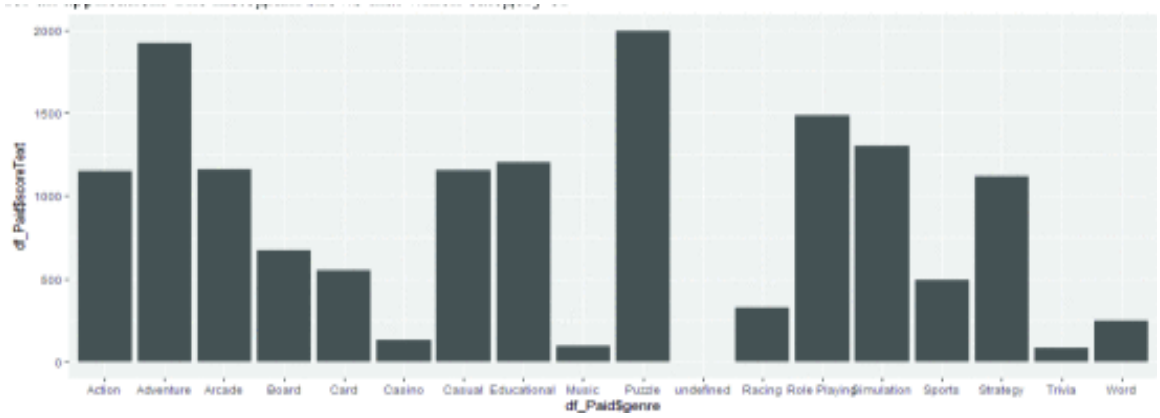
In Google play store users give ratings to the application according to there experience about that application.

It can be clearly visualized in the histogram that most people give 4.5 ratings to the free game as shown.



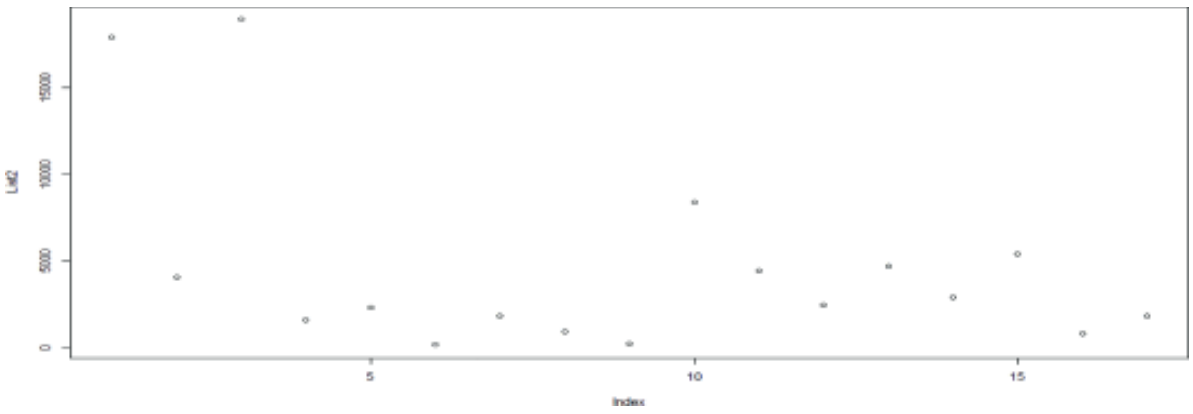
3. All Categories of Paid Games Ratings

In Google play store ratings matter a lot in decision making for an application. It can be clearly visualized by a bar graph that most people give ratings to the puzzle category as shown.



4. All Categories of Paid Games Installs

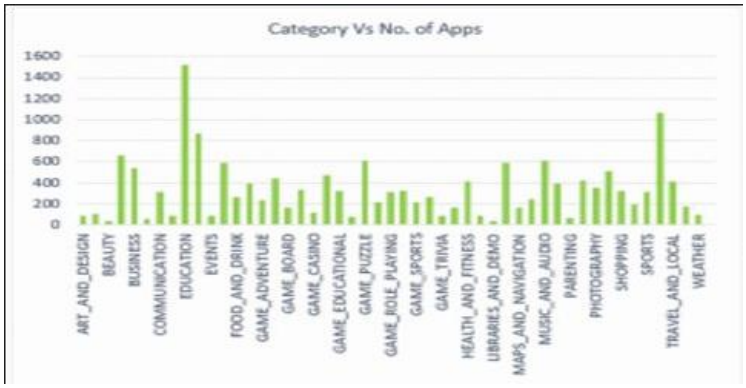
If in Google play store find category which installs is most, guide developers to try their skills in that category. As Arcade category have a greater number of installs with respect to other categories as shown.



Paper 3: Android Apps Success Prediction Before Uploading on Google Play Store

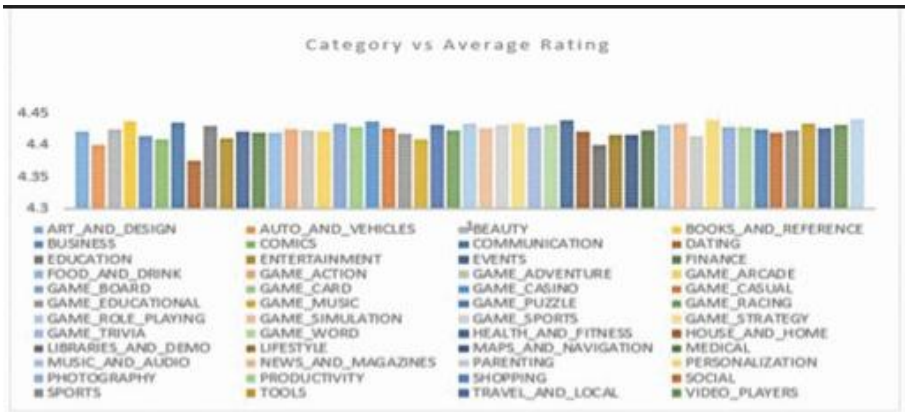
1. Category Vs App

The Category column in the dataset used has 34 different types of categories. Here they have shown the number of different types of apps against the categories. Figure shows the Bar Chart of the number of categories.



2. Category Vs Average Rating

The average rating of apps are presented with a Bar Chart. It shows the Bar Chart of all category against the Average Rating.



PROPOSED APPROACH

Discussion of Google play store dataset will involve various steps such as:

- Loading the data into data frame
- Cleaning the data
- Extracting statistics from the dataset
- Exploratory analysis and visualizations
- Questions that can be asked from the dataset
- Conclusion

IMPLEMENTATION OF PROPOSED APPROACH

○ Loading Data

It is done by reading the CSV file and doing initial statistical analysis. Though the dataset may seem to have the correct datatypes for each column, we need to check it.

▾ Taking the datasets as input

```
[ ] data1 = pd.read_csv("/content/drive/MyDrive/Data Visualization Project/googleplaystore.csv")
    data2 = pd.read_csv("/content/drive/MyDrive/Data Visualization Project/googleplaystore_user_reviews.csv")
```

```
[ ] data1.head()
```

	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 ...	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

Data Preparation

The process of cleaning and transforming raw data prior to processing and analysis is known as data preparation. It's a crucial step before processing that often entails reformatting data, making data corrections, and combining data sets to enrich data.

As there are many missing values in the dataset. They need to be removed, as they may create problems in the visualization.

Taking care of missing values

```
[ ]  data1.isna().sum()
```

```
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
```

```
[ ]  data1_1 = data1.dropna()
```

Finally, the dataset has no missing values now.

Now, the dataset has no missing values

```
[ ]  data1_1.isna().sum()
```

```
App             0
Category        0
Rating          0
Reviews         0
Size            0
Installs        0
Type            0
Price           0
Content Rating  0
Genres          0
Last Updated    0
Current Ver     0
Android Ver     0
dtype: int64
```

○ Cleanse and validate data

The Dataset has many columns that are not having proper numerical contents and they cannot be worked on directly. We will map them to normal numbers so that we are free to work with them.

▾ Data Cleaning

```
[ ] data1_1.head()
```

	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 ...	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

Here, are the code lines that remove unwanted symbols from the columns in the dataset.

```
[ ] data1_1['Installs'] = data1_1['Installs'].map(lambda x: x.rstrip('+'))

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.

data1_1['Installs'] = pd.to_numeric(data1_1['Installs'].str.replace(',',''))

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.

[ ] data1_1['Price'] = pd.to_numeric(data1_1['Price'].str.replace('$',''))

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single charac
  """Entry point for launching an IPython kernel.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

The '+' sign , the ',' sign and the '$' sign have been cleansed and the string data has been converted to strictly numeric.

```
data1_1.head()
```

	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	10000	Free	0.0	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	500000	Free	0.0	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 ...	ART_AND_DESIGN	4.7	87510	8.7M	5000000	Free	0.0	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	50000000	Free	0.0	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	100000	Free	0.0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

○ Data Transformation

The "Installs" column is having too much variance, so the variance has been reduced by applying log to the whole column.

```
[ ] data1_1.var()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=Nor
  """Entry point for launching an IPython kernel.
Rating      2.654959e-01
Installs    8.329549e+15
Price       2.503243e+02
dtype: float64
```

```
[ ] data1_1['Installs with Log'] = np.log(data1_1["Installs"])
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

Now, the variance of the "Installs" column has been reduced and the new values have been added to the new column, "Installs with Log".

```
[ ] data1_1.var() #Now, the variance of Installs column has been normalized
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (w
  """Entry point for launching an IPython kernel.
Rating               2.654959e-01
Installs             8.329549e+15
Price                2.503243e+02
Installs with Log    1.475847e+01
dtype: float64
```

Duplicate values can create a lot of problems. Hence, the duplicates have been removed. Duplicates could be there in the dataset due to web scraping of the same kind of data from the Internet.

```
[ ] data1_1["App"].duplicated().sum()
```

```
1170
```

```
[ ] data1_1.drop_duplicates(inplace = True)
```

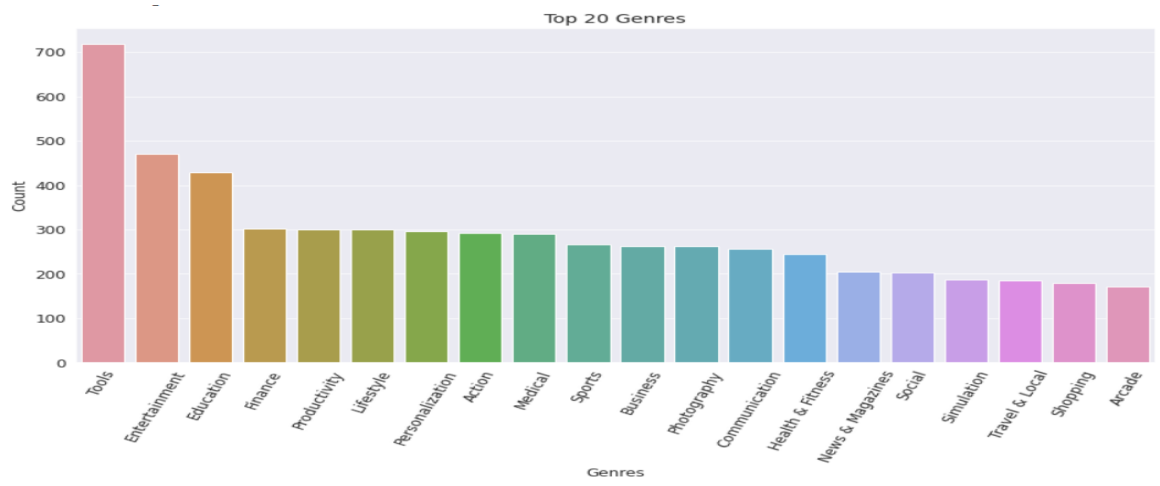
```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return func(*args, **kwargs)
```

IMPLEMENTATION

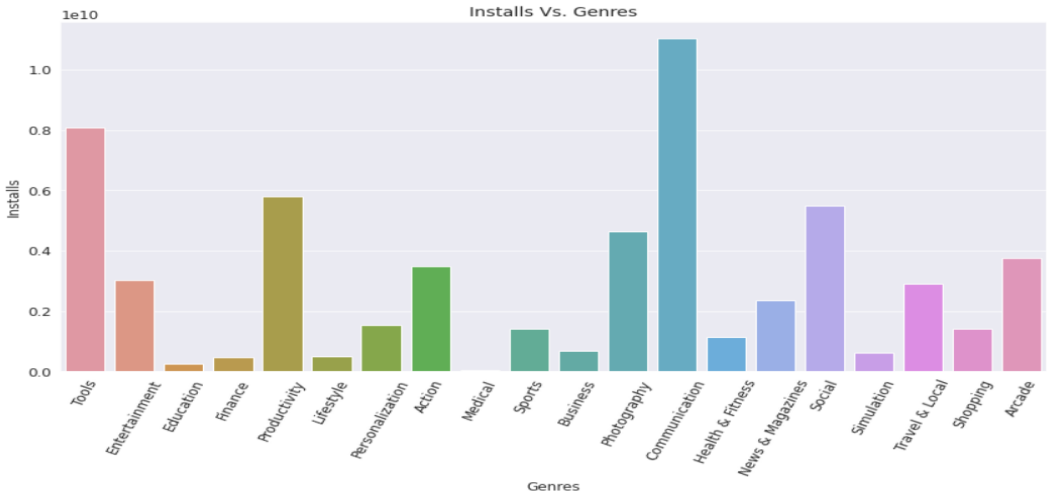
What are the top 20 apps in the Google Play Store organised by genre?

```
best_gen = data1_1.Genres.value_counts().reset_index().rename(columns={'Genres':'Count','index':'Genres'})
install_no = data1_1.groupby(['Genres'])[['Installs']].sum()
final_install = pd.merge(best_gen, install_no, on='Genres')
install_20 = final_install.head(20)
plt.figure(figsize=(14,7))
plt.xticks(rotation=65)
plt.xlabel("Genres")
plt.ylabel("Number of Application")
plt.title("Top 20 Genres")
sns.barplot(install_20.Genres, install_20.Count)
plt.show()
```



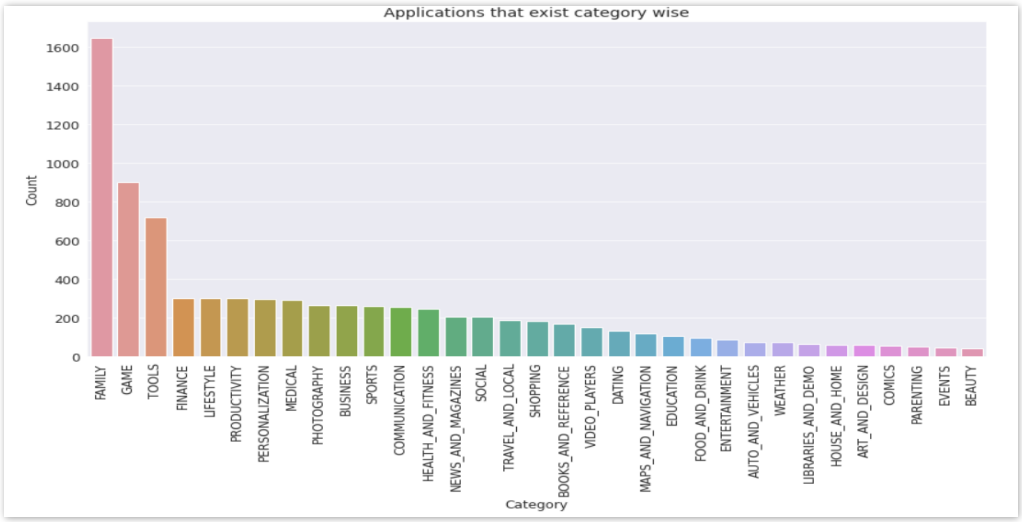
What are the most popular Genres among the top 20 Genres?

```
plt.figure(figsize=(14,7))
plt.xticks(rotation=65)
plt.xlabel("Genres")
plt.ylabel("Installs")
plt.title("Installs Vs. Genres")
sns.barplot(install_20.Genres, install_20.Installs)
plt.show()
```



What is the count of apps in each category?

```
best_cat = data1_1.Category.value_counts().reset_index().rename(columns={'Category':'Count','index':'Category'})
cat_install = data1_1.groupby(['Category'])[['Installs']].sum()
best_cat_install = pd.merge(best_cat, cat_install, on='Category')
best_20_cat_install = best_cat_install
plt.figure(figsize=(14,7))
plt.xticks(rotation=90)
plt.xlabel("Category")
plt.ylabel("Number of application")
plt.title("Applications that exist category wise")
sns.barplot(best_20_cat_install.Category, best_20_cat_install.Count)
plt.show()
```

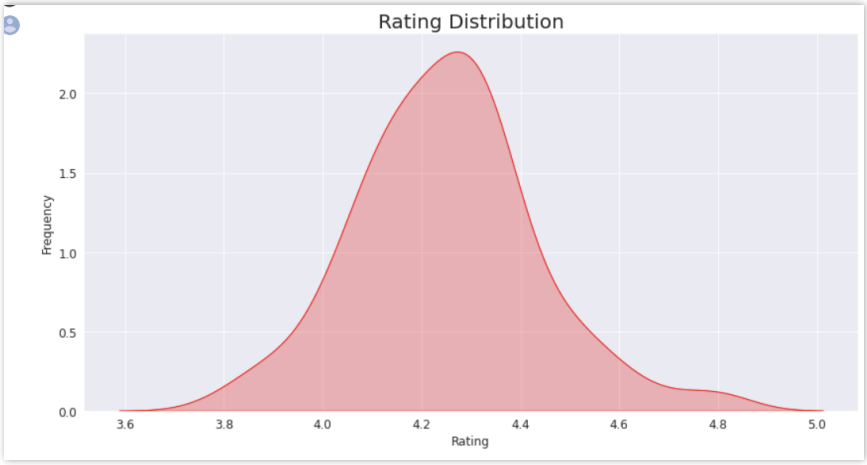


Distribution of average app rating

```
[ ]  average_rating_of_apps = data1_1.groupby(['Genres'])[['Rating']].mean()
```

```
[ ]  ratings_of_genres = pd.merge(best_gen, average_rating_of_apps, on='Genres')
```

```
plt.figure(figsize=(14,7))
g = sns.kdeplot(ratings_of_genres.Rating, color="Red", shade = True)
g.set_xlabel("Rating")
g.set_ylabel("Frequency")
plt.title('Rating Distribution',size = 20)
plt.show()
```



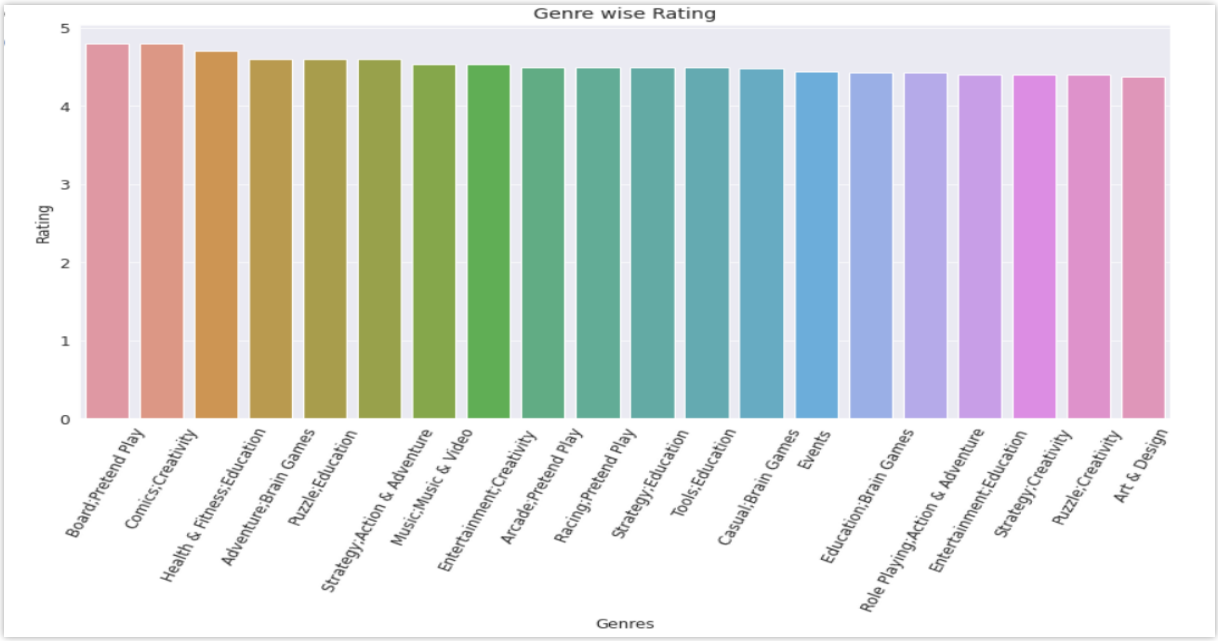
Highest and Lowest Performing Genres

Double-click (or enter) to edit

```
[ ]  ratings_of_genres.sort_values('Rating', ascending =False, inplace=True)
```

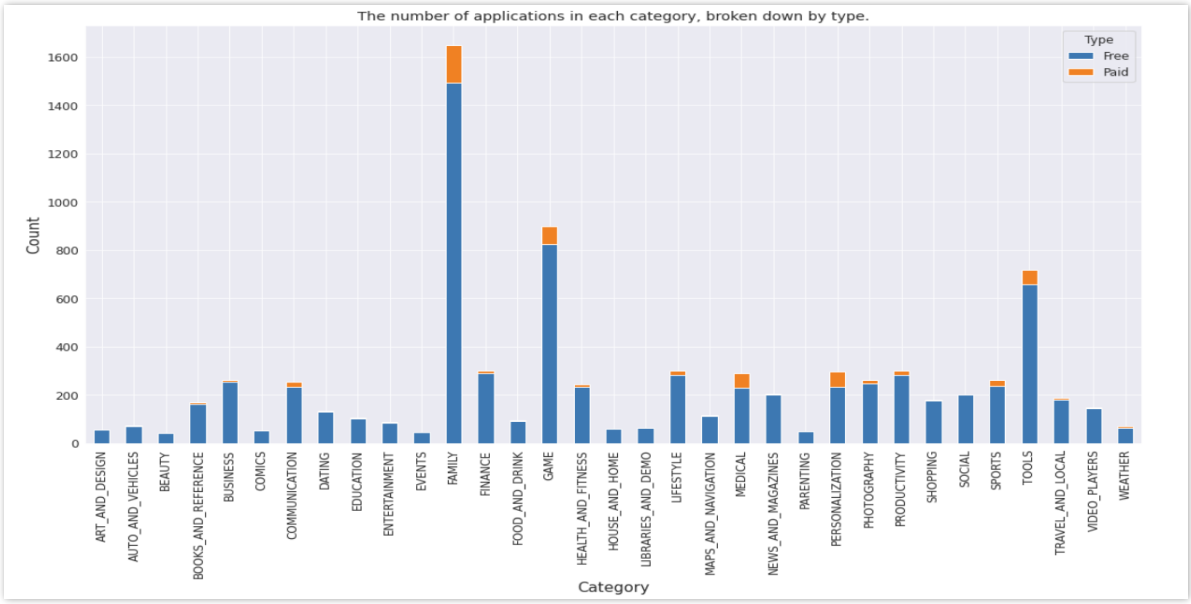
```
[ ]  high_rating_gen = ratings_of_genres.iloc[0:20]
     low_rating_gen = ratings_of_genres.iloc[-20:]
     low_rating_gen = low_rating_gen[low_rating_gen['Rating'].notnull()]
```

```
[ ]  plt.figure(figsize=(14,7))
     plt.xticks(rotation=65)
     plt.xlabel("Genres")
     plt.ylabel("Rating")
     plt.title("Genre wise Rating")
     sns.barplot(high_rating_gen.Genres, high_rating_gen.Rating)
     plt.show()
```



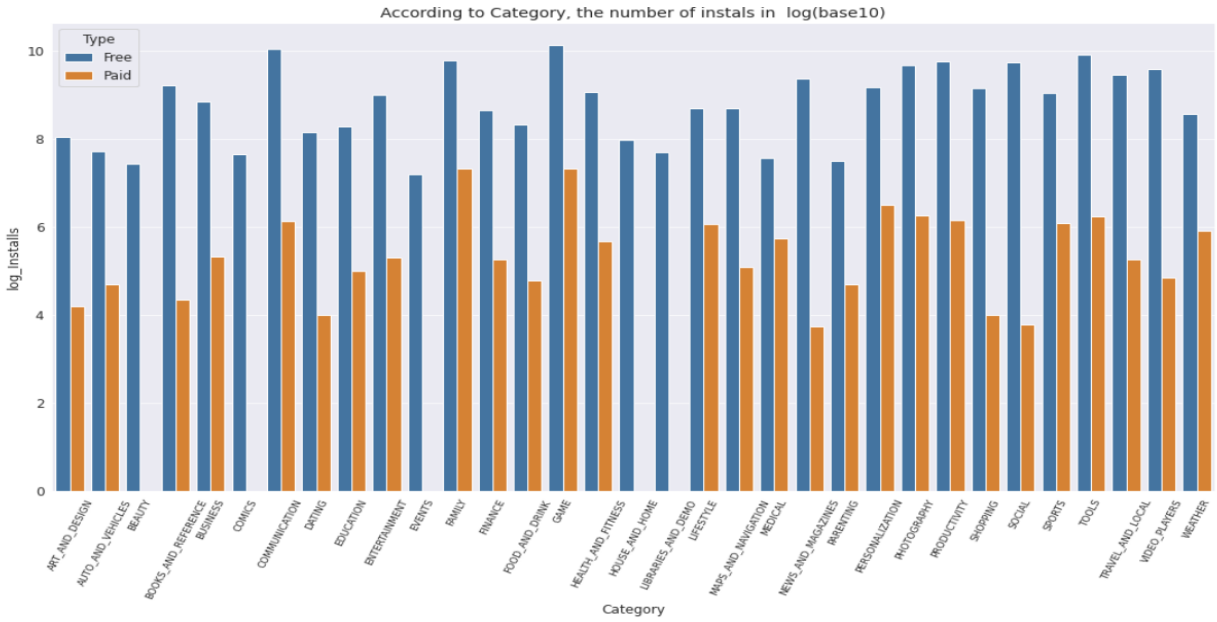
What are the count of applications in each category differentiated by their type?

```
count_of_apps = data1_1.groupby(['Category','Type'])[['App']].count().reset_index().rename(columns={'App':'Count','index':'App'})
count_of_apps_data1_1 = count_of_apps.pivot('Category', 'Type', 'Count').fillna(0).reset_index()
count_of_apps_data1_1.set_index('Category').plot(kind='bar', stacked=True, figsize=(18,9))
plt.xlabel("Category", fontsize=15)
plt.ylabel("Count", fontsize=15)
plt.title("The number of applications in each category, broken down by type.")
plt.show()
```



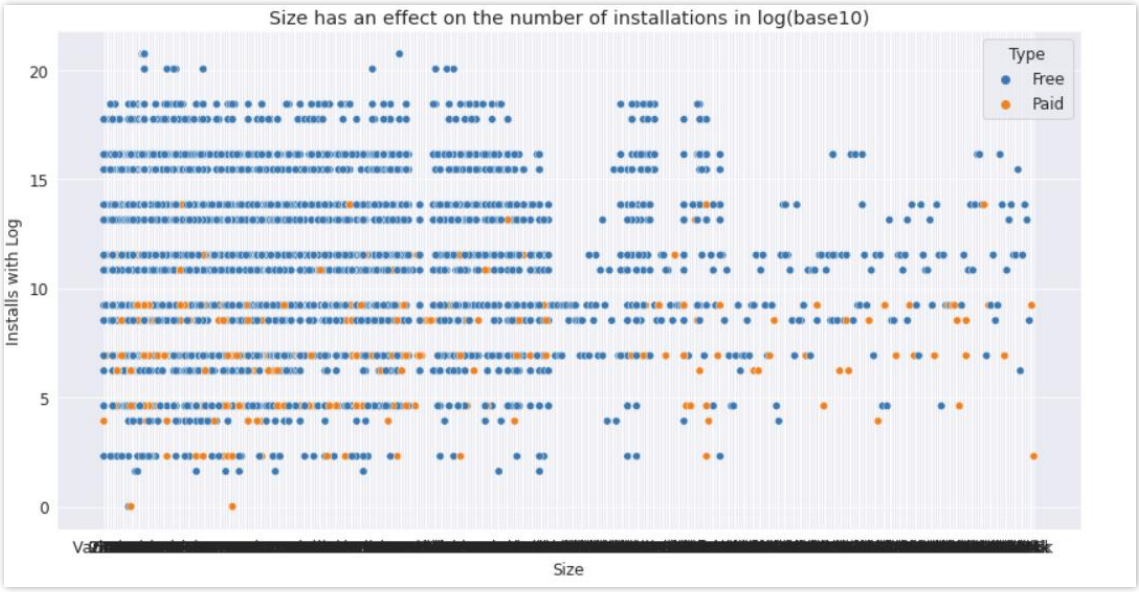
How many apps were installed according to its type?

```
data1_1['Gaming Apps'] = data1_1['Category']=='GAME'
install_by_cat = data1_1.groupby(['Category','Type'])[['Installs']].sum().reset_index()
install_by_cat['log_Installs'] = np.log10(install_by_cat['Installs'])
plt.figure(figsize=(18,9))
plt.xticks(rotation=65,fontsize=9)
plt.xlabel("Category")
plt.ylabel("Installs(base10)")
plt.title("According to Category, the number of instals in  log(base10)")
sns.barplot('Category', 'log_Installs', hue='Type', data = install_by_cat);
plt.show()
```



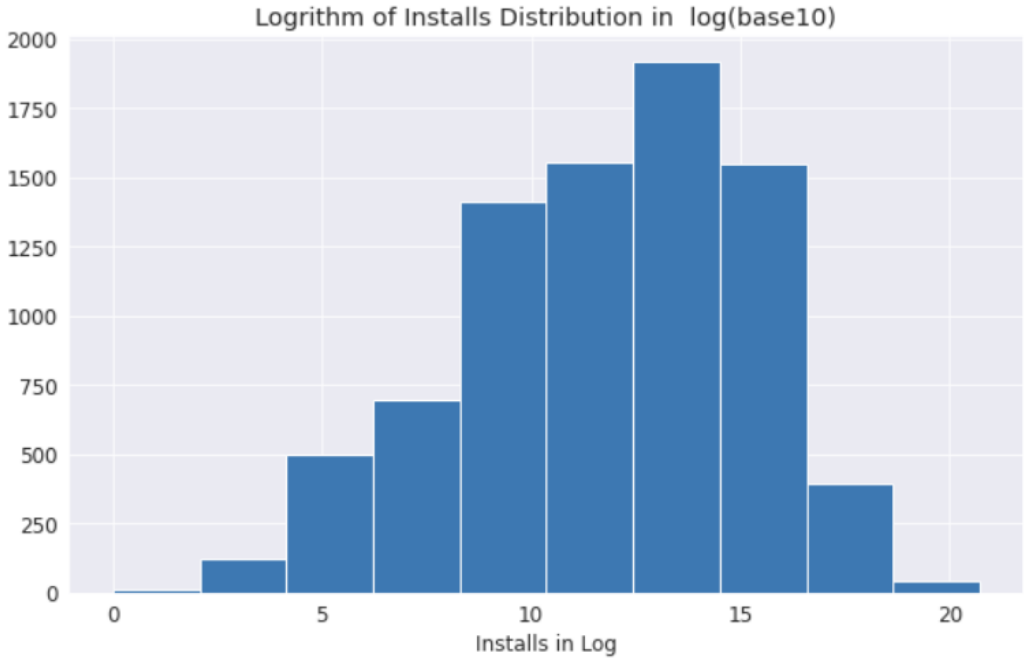
Affect of size on number of installation

```
plt.figure(figsize=(14,7))
plt.title("Size has an effect on the number of installations in log(base10)")
sns.scatterplot(data1_1['Size'], data1_1['Installs with Log'], hue=data1_1['Type'])
plt.show()
```



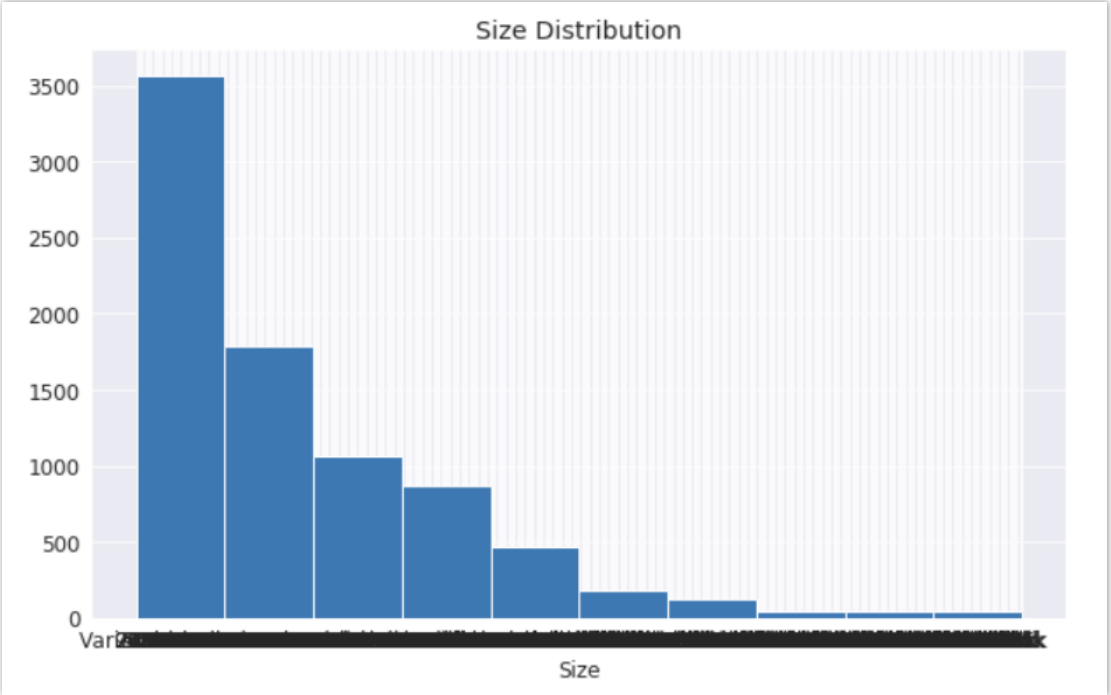
Histogram for various attributes

```
data1_1.loc[data1_1['Installs with Log']==data1_1['Installs with Log'].min(),'Installs with Log']=0
plt.xlabel("Installs in Log")
plt.title("Logrithm of Installs Distribution in  log(base10)")
plt.hist(data1_1['Installs with Log']);
```



Histogram for various attributes

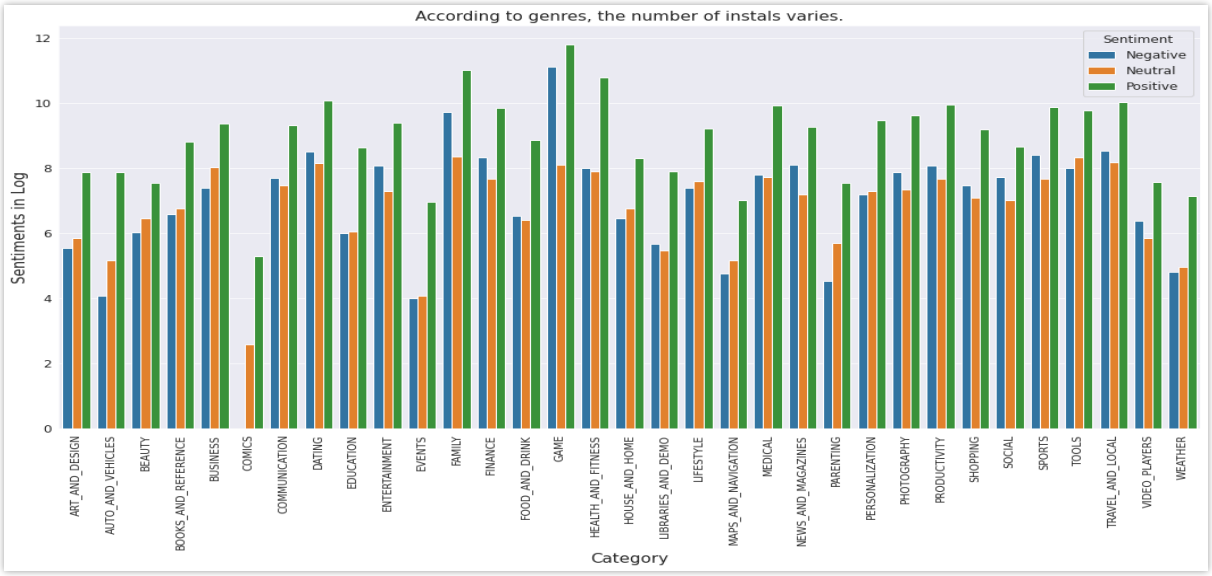
```
[ ]  plt.xlabel("Size")
     plt.title("Size Distribution")
     plt.hist(data1_1['Size']);
     plt.show()
```



Now, we will merge the seocnd dataset with the current dataset

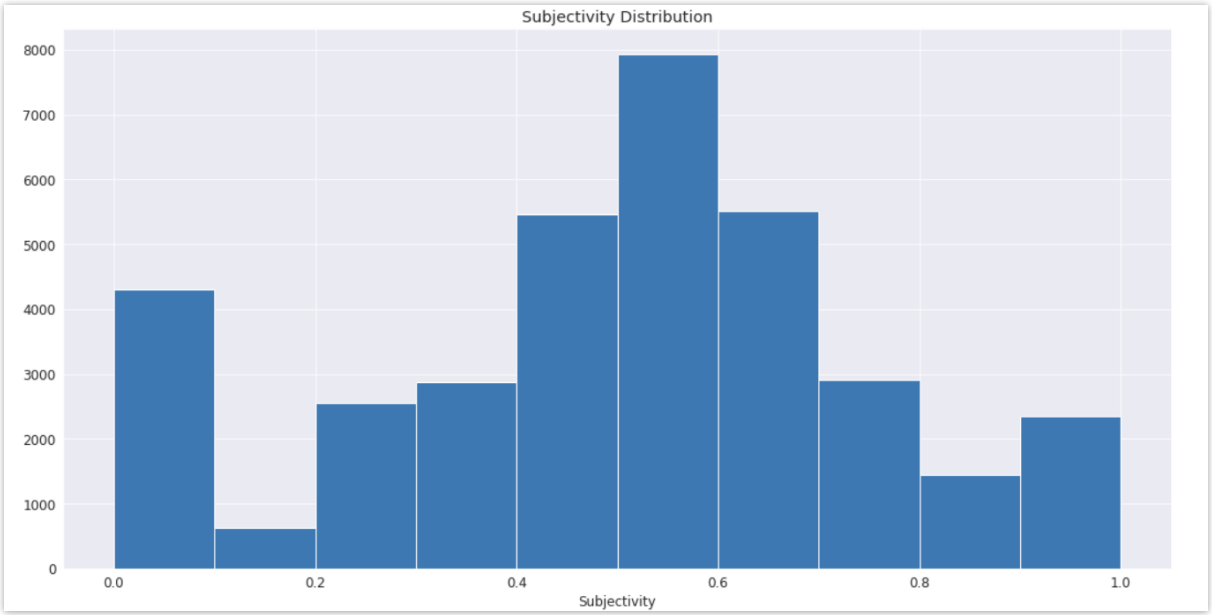
```
[ ]  data_merge = data1_1.merge(data2, on="App")
```

```
sentiments = data_merge.groupby(['Category','Sentiment']).size().reset_index(name='Sentiment Count')
sentiments['Sentiments in Log'] = np.log2(sentiments['Sentiment Count'])
plt.figure(figsize=(18,9))
plt.xticks(rotation=90,fontsize=11)
plt.xlabel("Category", fontsize=15)
plt.ylabel("Installs", fontsize=15)
plt.title("According to genres, the number of instals varies.",fontsize=15)
sns.barplot('Category', 'Sentiments in Log', hue='Sentiment', data = sentiments);
```



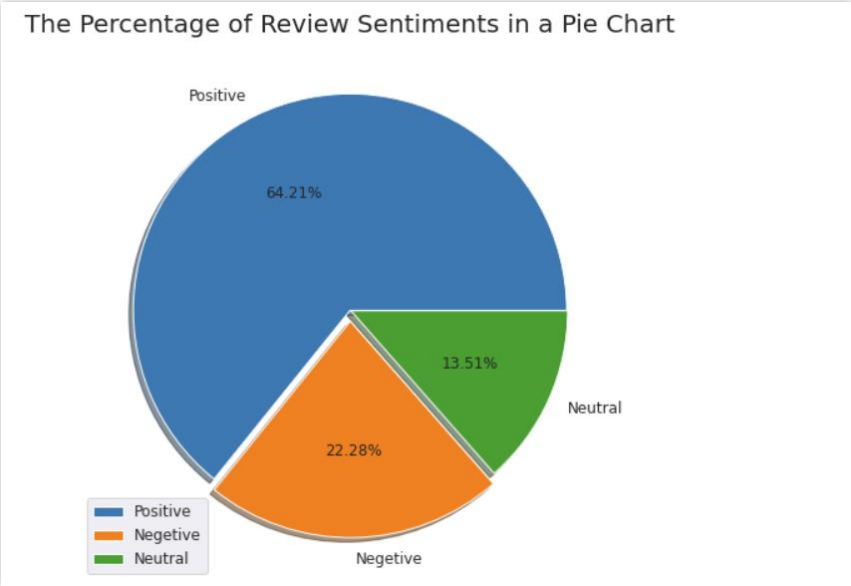
Subjectivity Distribution

```
plt.figure(figsize=(18,9))
plt.xlabel("Subjectivity")
plt.title("Subjectivity Distribution")
plt.hist(data_merge[data_merge['Sentiment_Subjectivity'].notnull()]['Sentiment_Subjectivity'])
plt.show()
```

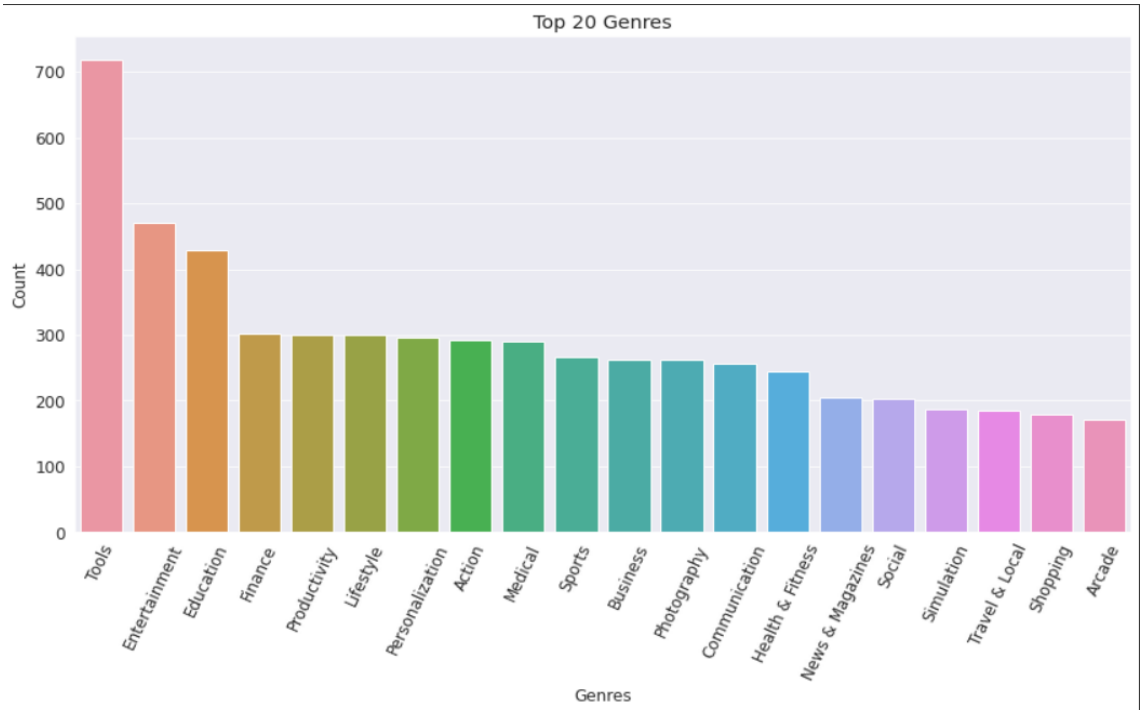


Review Sentiments as a Percentage

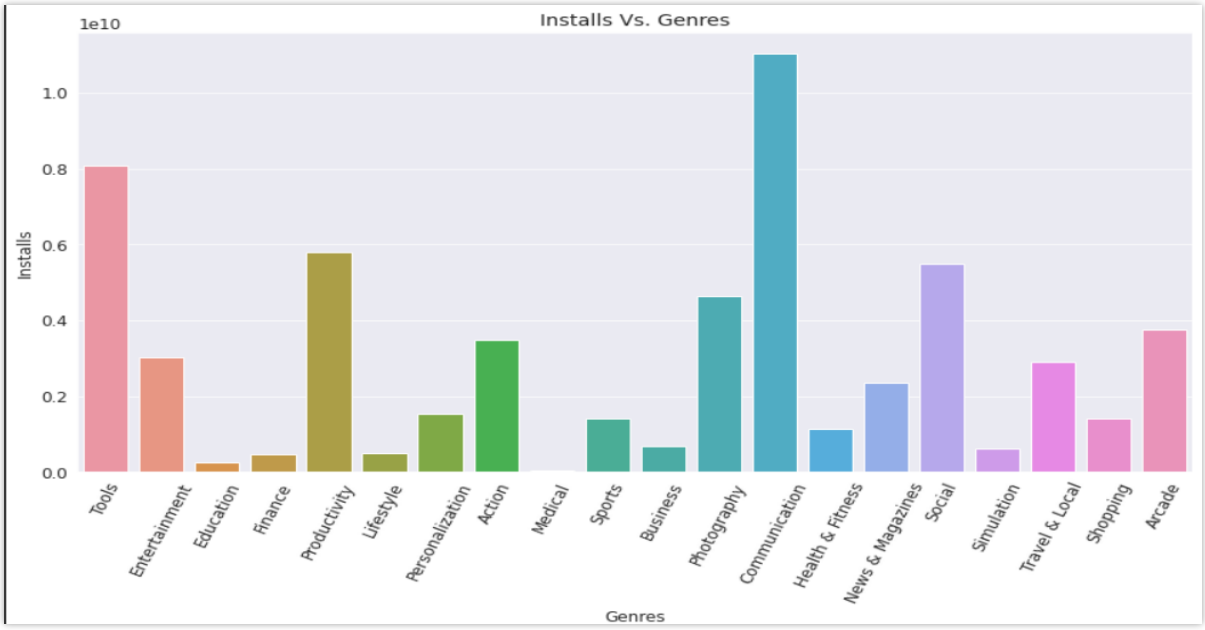
```
import matplotlib
counting = list(data_merge['Sentiment'].value_counts())
label_names = 'Positive', 'Negetive','Neutral'
matplotlib.rcParams['font.size'] = 12
matplotlib.rcParams['figure.figsize'] = (8, 8)
plt.pie(counting, labels=label_names, explode=[0, 0.05, 0.005], shadow=True, autopct="%.2f%%")
plt.title('The Percentage of Review Sentiments in a Pie Chart', fontsize=20)
plt.axis('off')
plt.legend()
plt.show()
```



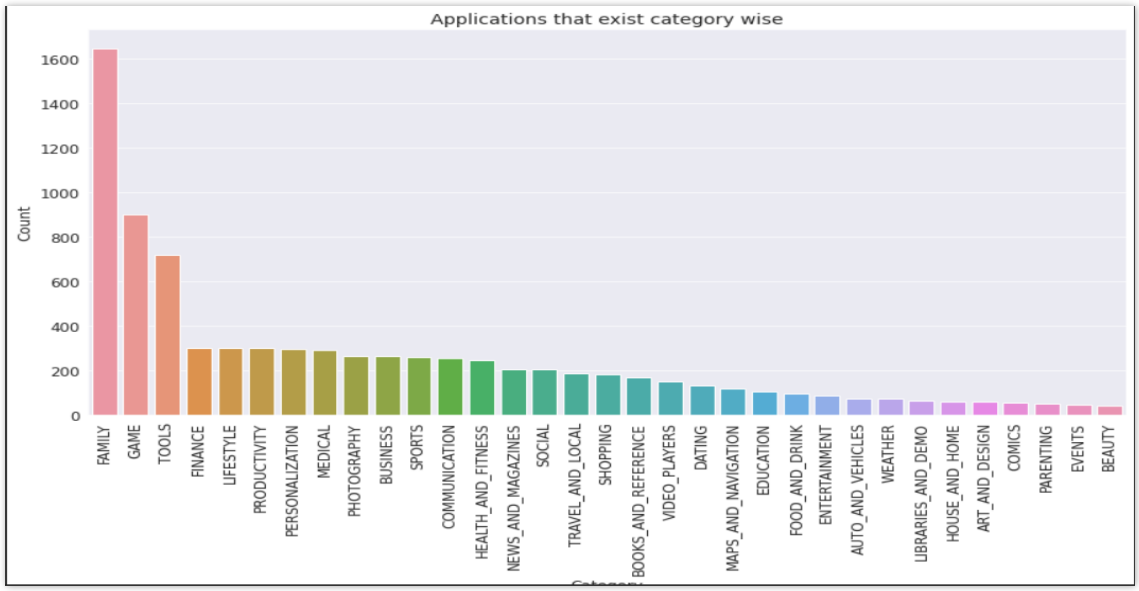
RESULTS AND ANALYSIS



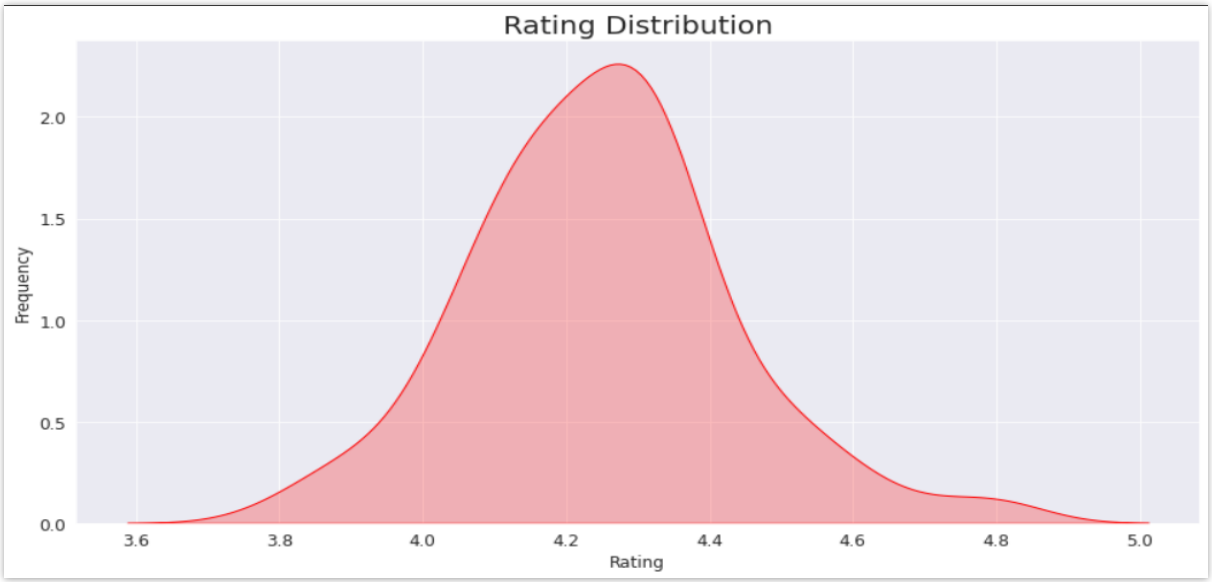
In the above graph, we have plotted a bar graph comparing the number of apps for the top 20 genres. Out of the 9660 different apps present on google play store, nearly 700 (8%) apps belong to the genre “Tools”, thus making it the genre with highest number of apps. At a distant second position comes the genre “Entertainment” with nearly 450-475 (6%) apps. The genre “Education” is at third position with nearly same number of apps as that of “Entertainment”. In the top 20 genre categories, the genres “Shopping” and “Arcade” with nearly 175 apps have the least number of apps under them.



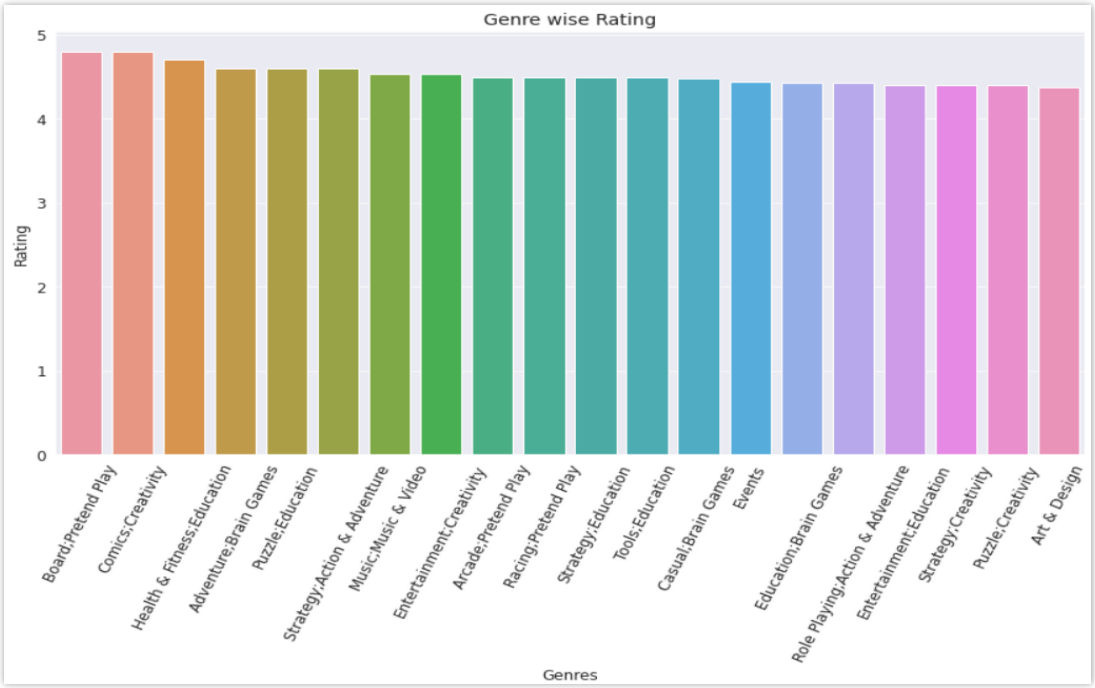
Using the bar plot, we have visualized the number of installs of every genre. We can infer that the highest number of installs by genre is for "Communication" boasting nearly 11.5 billion installs. The second highest number of installs by genre is for "Tools" which has been installed nearly 8 billion times. The least number of installs is for genre "Medical". The number of installs for "Medical" is so negligible that it is not even visible in the visualization.



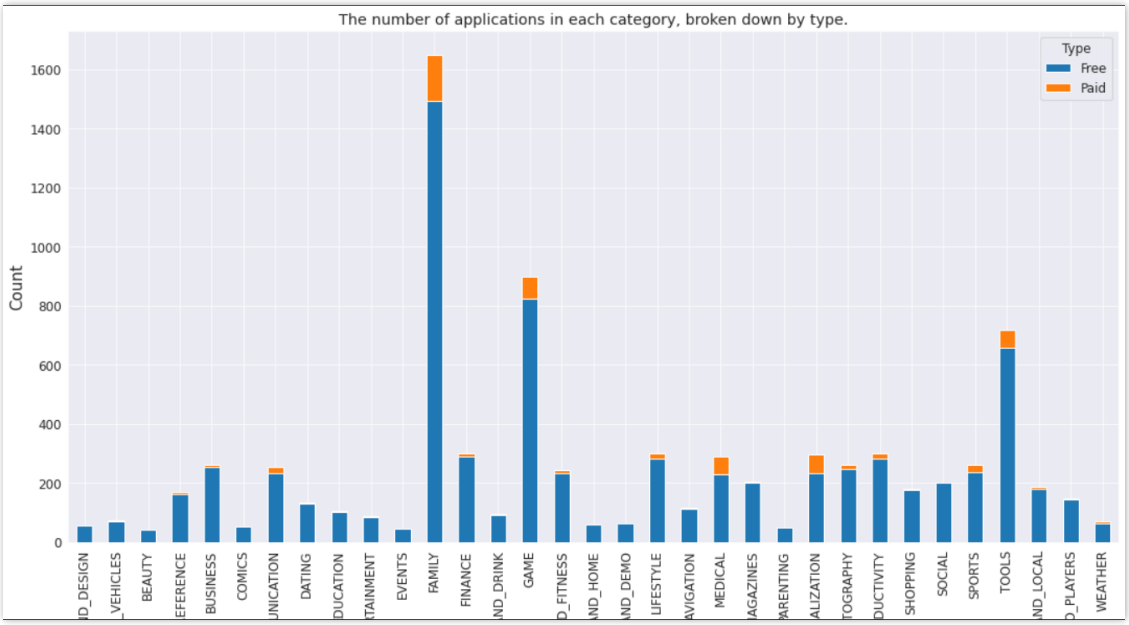
The graph above plots a bar graph comparing the number of apps category-wise. Of the 9660 types of apps in the Google Play store, nearly 1600 of the apps belong to the category ‘family’, which is the genre with the largest number of apps. In the second position far away, there is a category “Game” with apps close to 900. The category “Tools” ranks third with about 700 apps under its spectrum. The "Events" and "Beauty" categories, which include nearly 30-50 apps, have the lowest number of apps underneath.



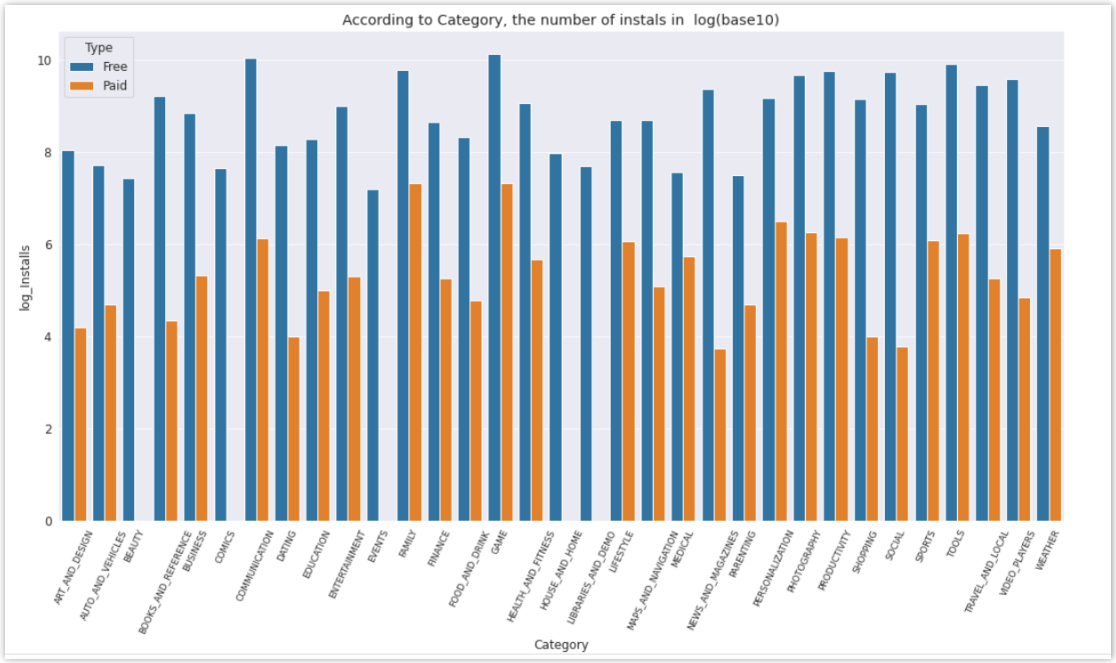
The gaussian distribution above is slightly positively skewed i.e., the mean of the ratings is greater than the median. As we can infer from the above graph, the ratings of most of the apps are nearly 4.2-4.3, i.e., the mode of the ratings is almost 4.3. The median will be slightly higher than the mode, and the mean will be slightly higher than the median of the ratings.



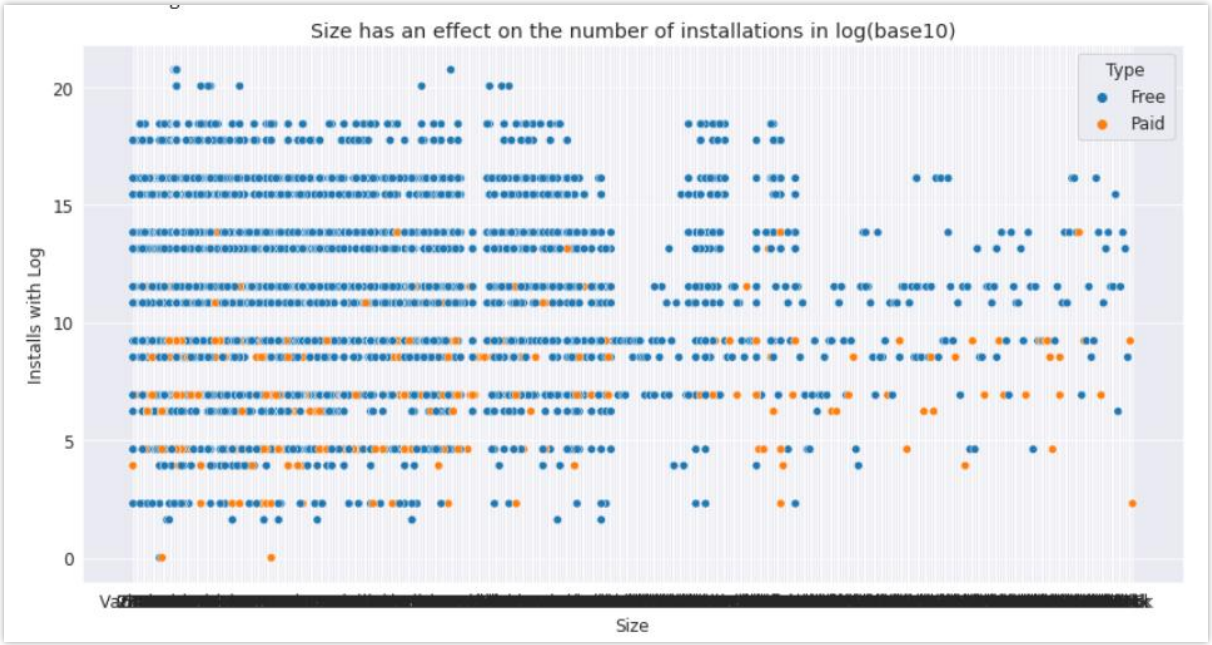
Using bar plot, we have visualized the highest rated genres. Top Genres have been ordered in bar plot on the basis of the highest rating of any app under its spectrum. An app belonging to genres “Board” and “Pretend Play” have the highest rating of nearly 4.75; while the app belonging to genres “Comics” and “Creativity” come at second position with nearly same rating as those of “Board” and “Pretend Play”. In the top 20 rating spots of the apps per genre, genre “Art and Design” comes at the last position with the apps under this genre having the highest rating of nearly 4.5.



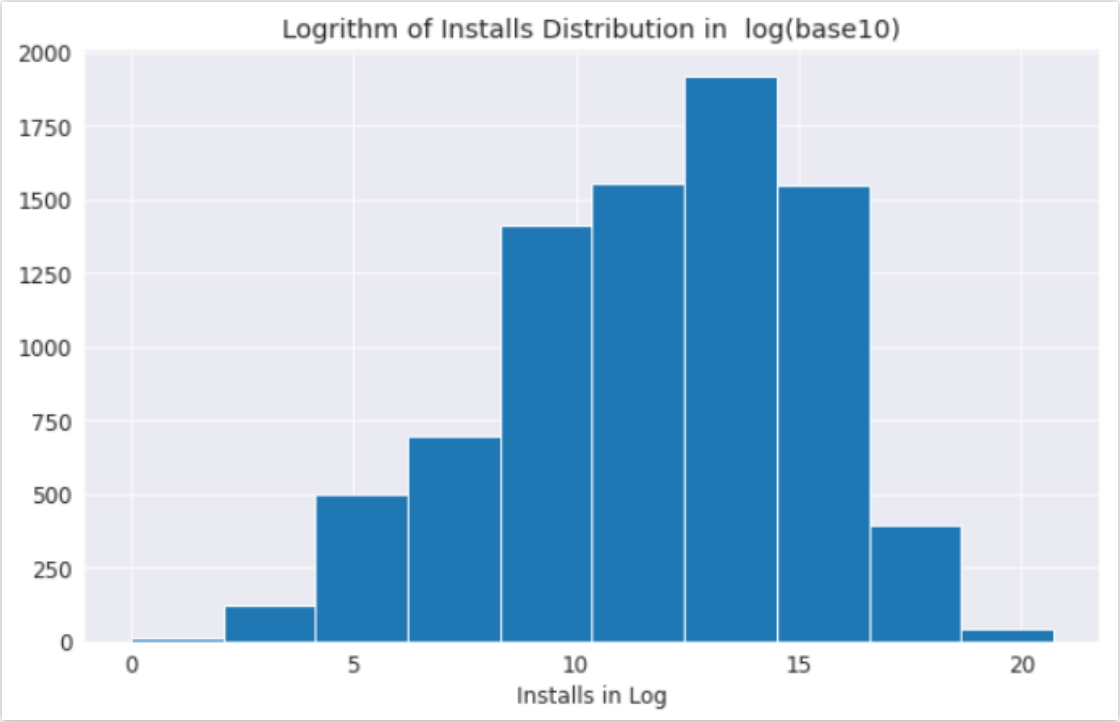
Using Stacked Bar Plot, we have visualized the plot to ascertain the number of free and paid apps per category of app. We can see categories such as “Art and Design”, “Auto and Vehicles”, “Beauty”, “Comics”, “House and Home”, “Social”, “Shopping”, etc. have no or negligible number of paid apps under its spectrum. While apps categories such as “Communication”, “Finance”, “Health and Fitness”, “Photography”, etc. have lesser number of paid apps; “Game”, “Tools”, “Medical”, “Personalization have considerable number of apps which are paid. The list is topped by the category “Family” having the highest number of paid apps by count.



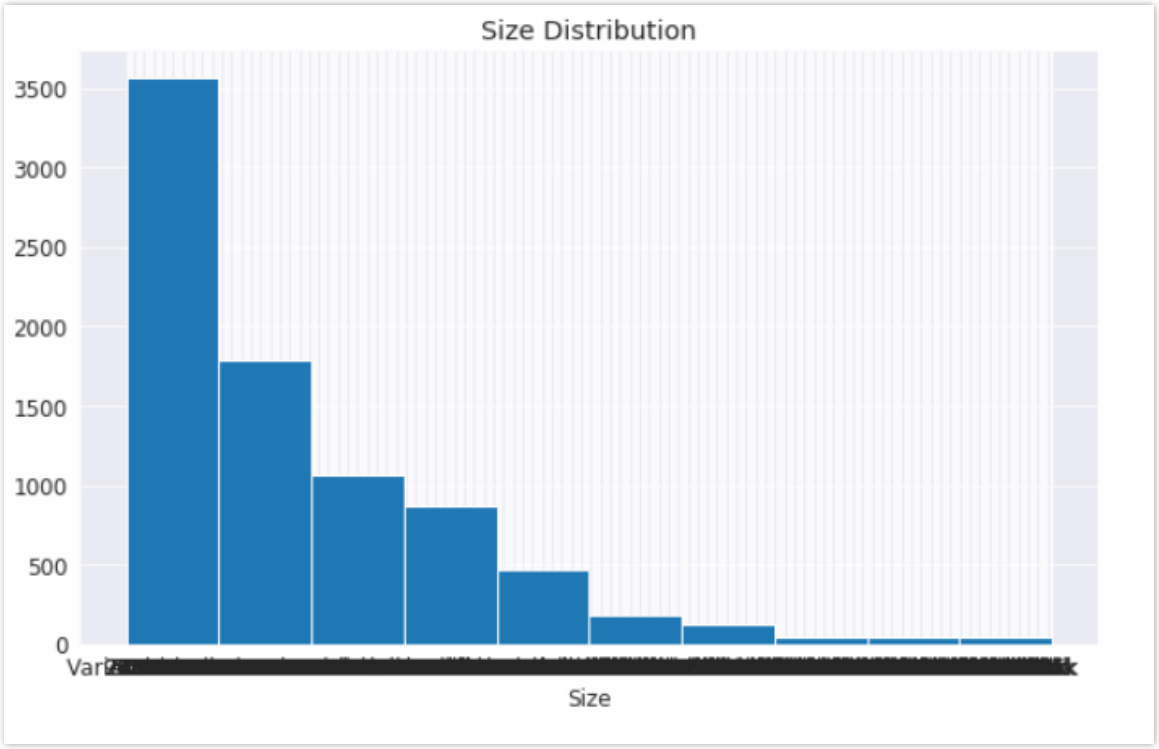
The represented group bar plot visualizes the number of installs (in log base 10 values) according to the category of the app and whether it is free or paid. For all categories free apps have more installs than paid apps. By the absence of orange bars in categories Beauty, Comics, House and Home, Libraries and demo we interpret that the number of installs for paid apps in these categories is very low.



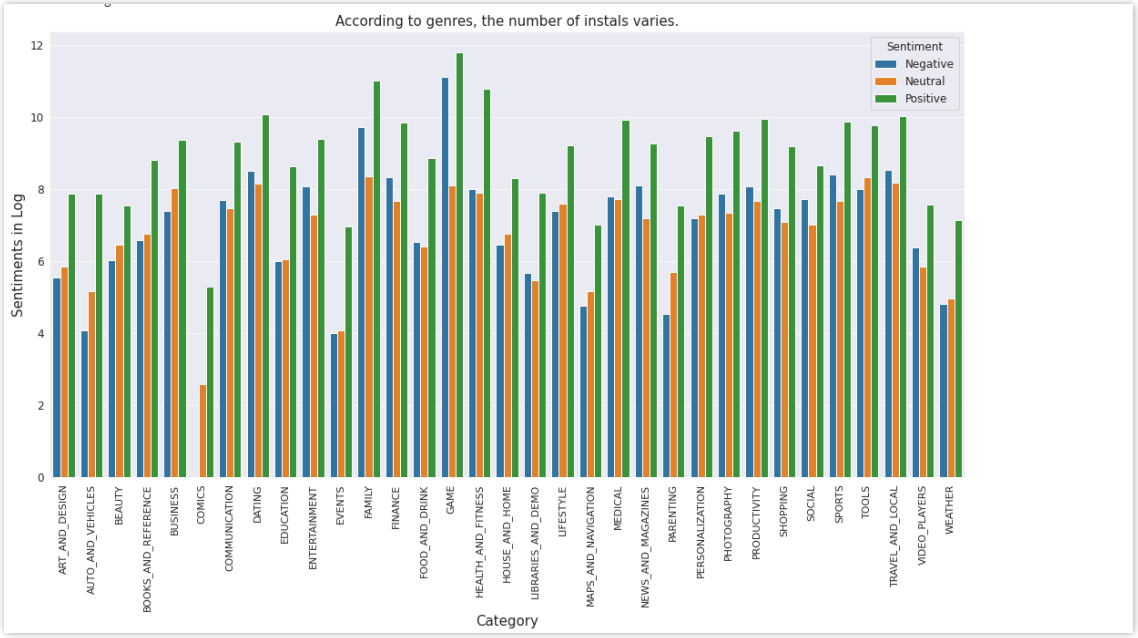
The above plot is a scatter plot of installs with log vs size of the app. The blue dots indicate free apps while orange dots indicate paid apps. It is clear from the above plot that size may impact the number of installations. Bulky applications are less installed by the user.



The histogram is for installs in log. Maximum number of installs are for the values 10^{10} to 10^{15}. Least are for installs 10^{0} to 10^{5}.

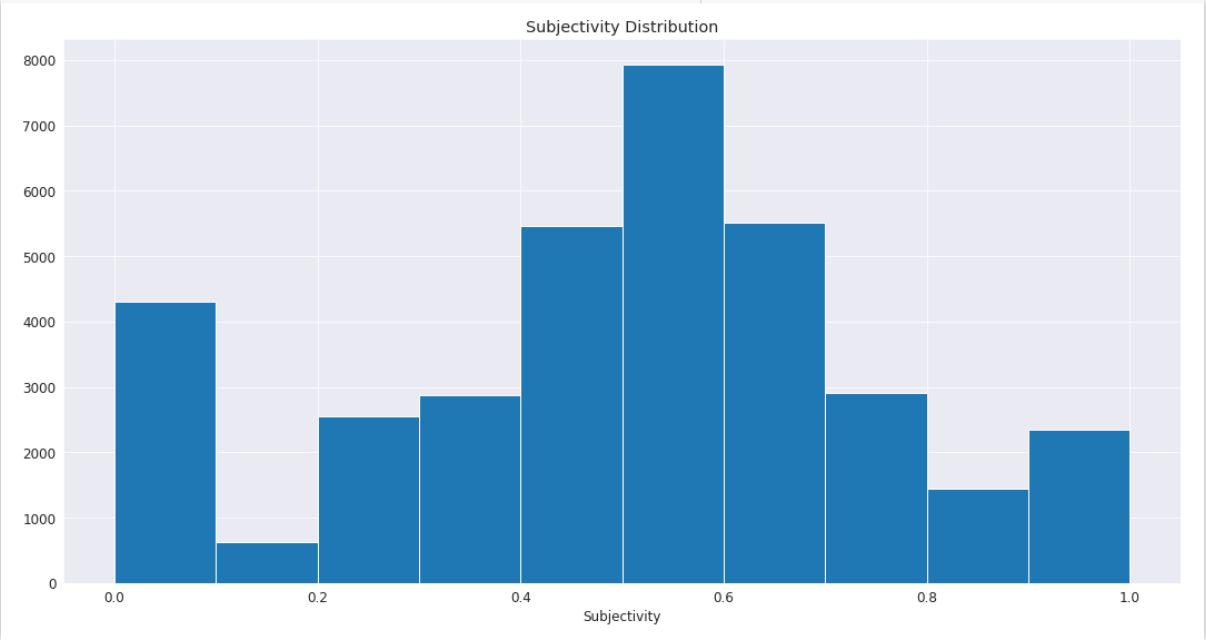


From the above histogram, it can be concluded that maximum number of applications present in the dataset are of small size.

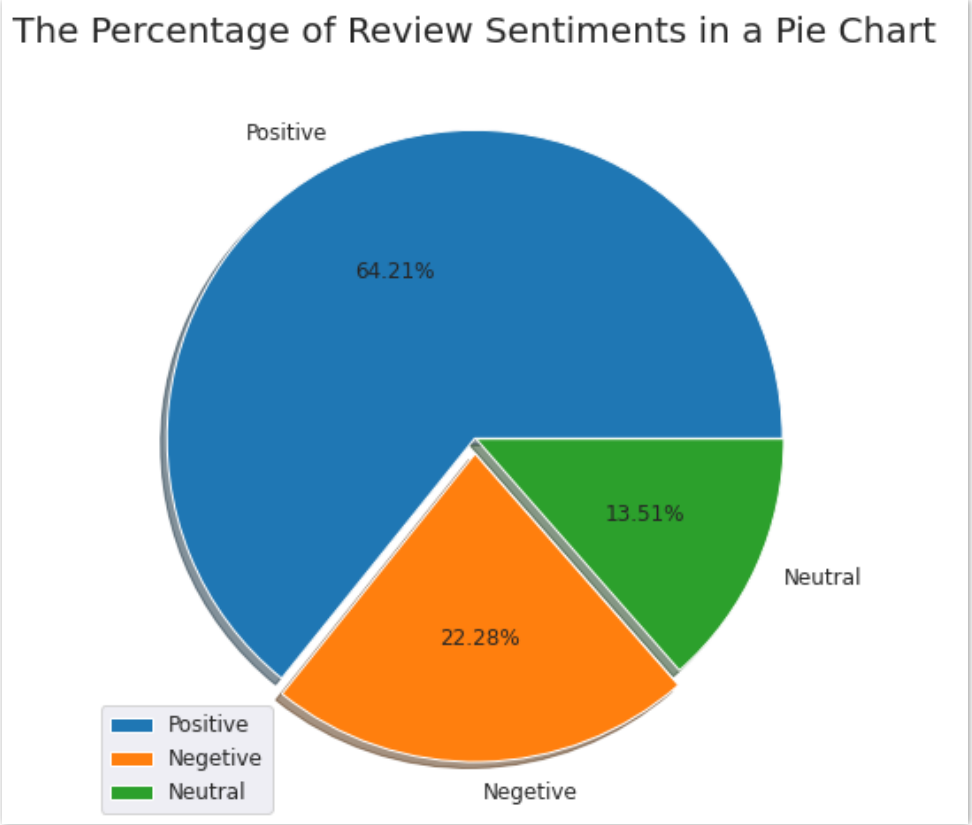


The plotted group bar graph represents the sentiments of the writer of the comment, divide into positive, negative and neutral represented by the colours green, blue and orange respectively. For all the genres there are more positive comments than negative or neutral comments. Comics genre does not have any negative comments, while many

genres like Auto and Vehicles, Business and Tools have more neutral reviews than negative. Some genres like Family, Game, Sport have more number of negative comments than neutral.



It can be seen that maximum number of sentiment subjectivity lies between 0.4 to 0.6. From this we can conclude that maximum number of users give reviews to the applications, according to their experience.



The pie chart represents the sentiments of the reviews. We see that most of the reviews are positive, some are negative and less are neutral

CONCLUSION

The information includes the potential to give insights that will assist developers better understand client wants and, as a result, popularize the product. The dataset may also be used to see if the app's original ratings match the anticipated rating, allowing you to determine whether the app is performing better or worse than other apps on the Play Store.

REFERENCES

1. "Apps Rating Classification on Play Store Using Gradient Boost Algorithm." *IEEE Xplore*, https://ieeexplore.ieee.org/document/9320756.

2. "Data Scraping from Google Play Store and Visualization of Its Content for Analytics." *IEEE Xplore*, https://ieeexplore.ieee.org/document/8673523.

3. "Android Apps Success Prediction before Uploading on Google Play Store." *IEEE Xplore*, https://ieeexplore.ieee.org/abstract/document/9068071/authors.

4. Research, Rahul Aralikatte IBM, et al. "Fault in Your Stars: Proceedings of the ACM India Joint International Conference on Data Science and Management of Data." *ACM Other Conferences*, 1 Jan. 2018, https://dl.acm.org/doi/abs/10.1145/3152494.3152500.

5. "Examining the Rating System Used in Mobile-App Stores." *IEEE Xplore*, https://ieeexplore.ieee.org/abstract/document/7045413.

6. "What Is Data Visualization? Definition, Examples, and Learning Resources." *Tableau*, https://www.tableau.com/learn/articles/data-visualization.

7. "Data Visualization Tools & Solutions." *IBM*, https://www.ibm.com/analytics/data-visualization.

8. Kuo, Jonathan C.T. "Data Science - a Deep Analysis on 'Google Play Store Apps' Dataset from Kaggle." *Medium*, Towards Data Science, 4 Aug. 2021, https://towardsdatascience.com/data-science-a-deep-analysis-on-google-play-store-apps-from-kaggle-8283bbc508b0.

9. "Example Gallery¶." *Example Gallery - Seaborn 0.11.2 Documentation*, https://seaborn.pydata.org/examples/index.html.

10. "Documentation." *Matplotlib*, https://matplotlib.org/3.1.1/index.html.

- **Code Link:**
  https://colab.research.google.com/drive/1yvrJ7iMKwsoRVVN9bc8lO03apGv05NaQ?usp=sharing#scrollTo=XAkJ5qwhHkb4