Analysis of the relationship between review sentiment, app ratings, and installations in the Google Play Store

Ethan Hu, Jianlin Hu, Edgar Guzman

Background

User-based reviews are a quinitessential tool for app developers to understand the needs and wants of the community. Positive reviews allow developers to understand where they went right with the app, provides a "sanity" check to make sure bugs and errors are not occurring, and leaves a favorable first impression to people looking to install a new app. On the other hand, negative reviews provide useful information for developers to quickly pinpoint where problems are occurring, and if left unchecked, results in a poor performing app that people are averted by.

The Problem

Analyzing the relationship between review sentiment, app ratings, and installations in the Google Play Store is crucial for several reasons. Firstly, understanding the sentiment expressed in user reviews provides valuable insights into user satisfaction, which is vital for app developers and marketers seeking to improve their products and enhance user experience. Secondly, uncovering patterns between sentiment, ratings, and installations can help identify factors influencing app success or failure, guiding developers in making informed decisions about app development, marketing strategies, and user engagement.

In general, this problem is important, as well as our analysis, as it contributes to the broader understanding of consumer behavior in the digital marketplace, shedding light on how user perceptions and preferences impact app performance.

Our Data

The primary data sources for our analysis are datasets from the Google Play Store, specifically the Google Play Store Apps dataset and the sentiment analysis dataset available on platforms like Kaggle. These datasets provide comprehensive information on app attributes, user reviews, sentiment scores, ratings, and installation counts, allowing for a thorough exploration of the relationships between these variables.

The data we have collected is from this souce: https://www.kaggle.com/datasets/lava18/google-play-store-apps/data?select=googleplaystore.csv

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	G
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & D
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Design;Pr
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & D
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & D
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Design;Crea

Description

The Dataset above contains information for many Apps in the Google Play store, with over 10,000 apps in this dataset. The following columns are described below:

Column	Description
App	The name of the application
Category	The category of the app
Rating	The average rating (out of 5) of the app
Reviews	The number of reviews submitted to the Google Play Store for that app
Size	The size (in Mb or KB) of the app
Installs	The number of installations of the app (binned by base-10 magnitude)
Type	The monetary type of the app (Free / Paid)
Price	The price of the app installation (0 if free)

Column	Description
Content Rating	The conntent based on ESRB rating
Genres	The type of genre the app is categorized as
Last Updated	The date the app was last updated
Current Ver	The version of the app as of the date the dataset was collected
Android Ver	The android version of the app as of the date the dataset was collected

Data Cleaning

We will remove the last 3 columns in order to save space and computation time, as they will not be used for any of our analyses. We will also remove `Genres` in favor of the simpler column `Category`, as many of the categories and genres are identical.

Before we can perform our EDA, we will need to clean our data by setting the proper dtypes for the following columns:

- Numbers should either be `float` or `int`, and any trailing symbols should be removed
- Nominal variables should be binarized or one-hot encoded

```
App                object
Category           object
Rating            float64
Reviews            object
Size               object
Installs           object
Type               object
Price              object
Content Rating     object
dtype: object
```

`Reviews`, `Size`, `Installs`, and `Price` will be converted to `int` or `float`.
`Category`, `Type`, and `Content Rating` will be converted into `category`.

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0M	1,000+	Free	0	Everyone	NaN

There is one row that is missing its `Category`, causing all of the data to shift to the left. While we could fix this issue manually, it is safer to remove the row.

Now that we have modified the dtypes, we can move on to the next step: Checking cells for missing values.

```
App                  0
Category             0
Rating            1474
Reviews              0
Size              1695
Installs             0
Type                 1
Price                0
Content Rating       0
dtype: int64
```

The cell above indicates that there are 1474 apps without a rating. Since we are looking to determine a relationship between app installations based on app ratings, we will have to remove these.

```
<class 'pandas.core.frame.DataFrame'>
Index: 9366 entries, 0 to 10839
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype   
---  ------          --------------  -----   
 0   App             9366 non-null   object  
 1   Category        9366 non-null   category
 2   Rating          9366 non-null   float64 
 3   Reviews         9366 non-null   int32   
 4   Size            7729 non-null   float64 
 5   Installs        9366 non-null   int32   
 6   Type            9366 non-null   category
 7   Price           9366 non-null   float64 
 8   Content Rating  9366 non-null   category
dtypes: category(3), float64(3), int32(2), object(1)
memory usage: 468.1+ KB
```

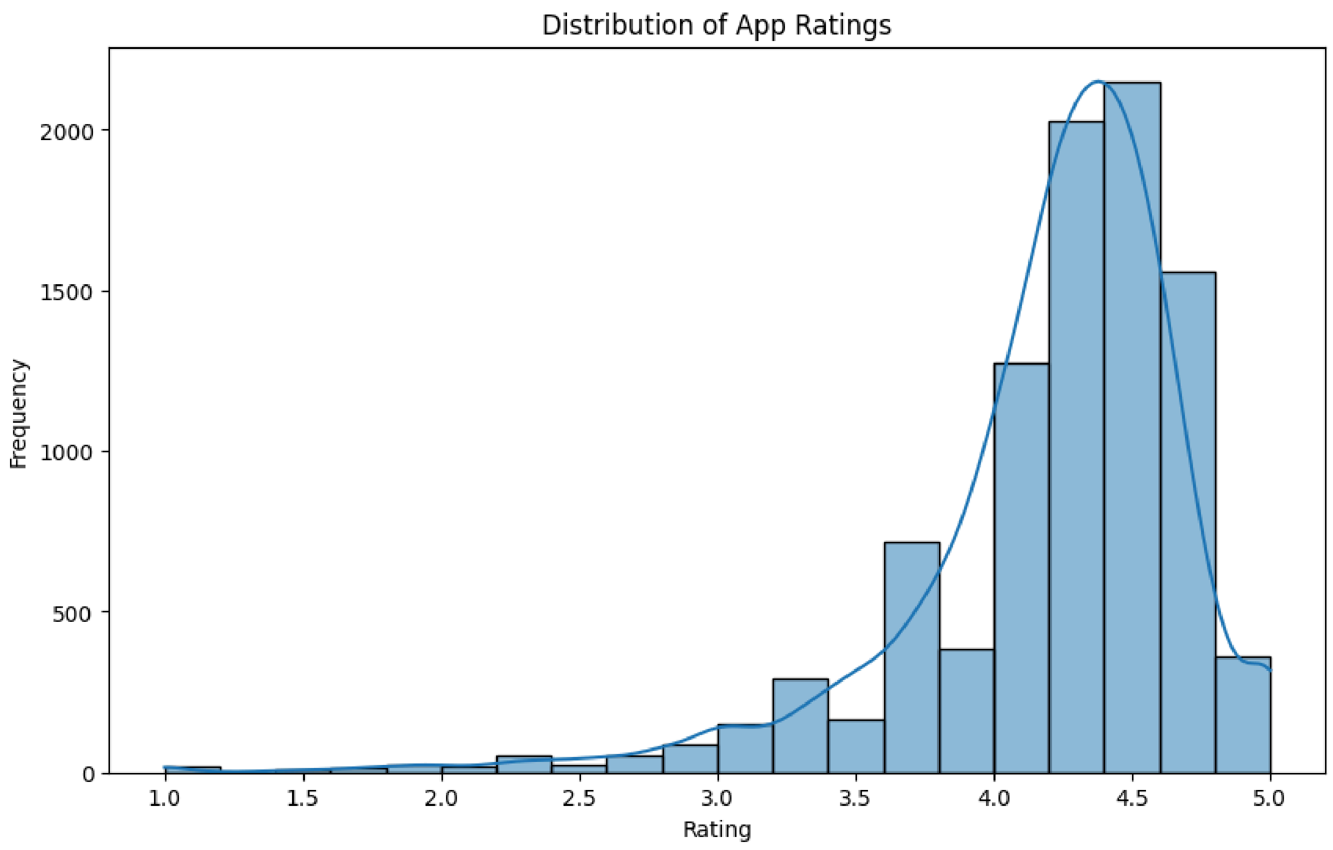
There are still 1637 rows with no `Size` data; however, removing these datapoints would significantly alter most hypothesis tests, so we choose to keep them. Should we decide to explore a hypothesis test using app size, we will perform mean imputation by category to determine an ideal estimate.

Exploratory Data Analysis

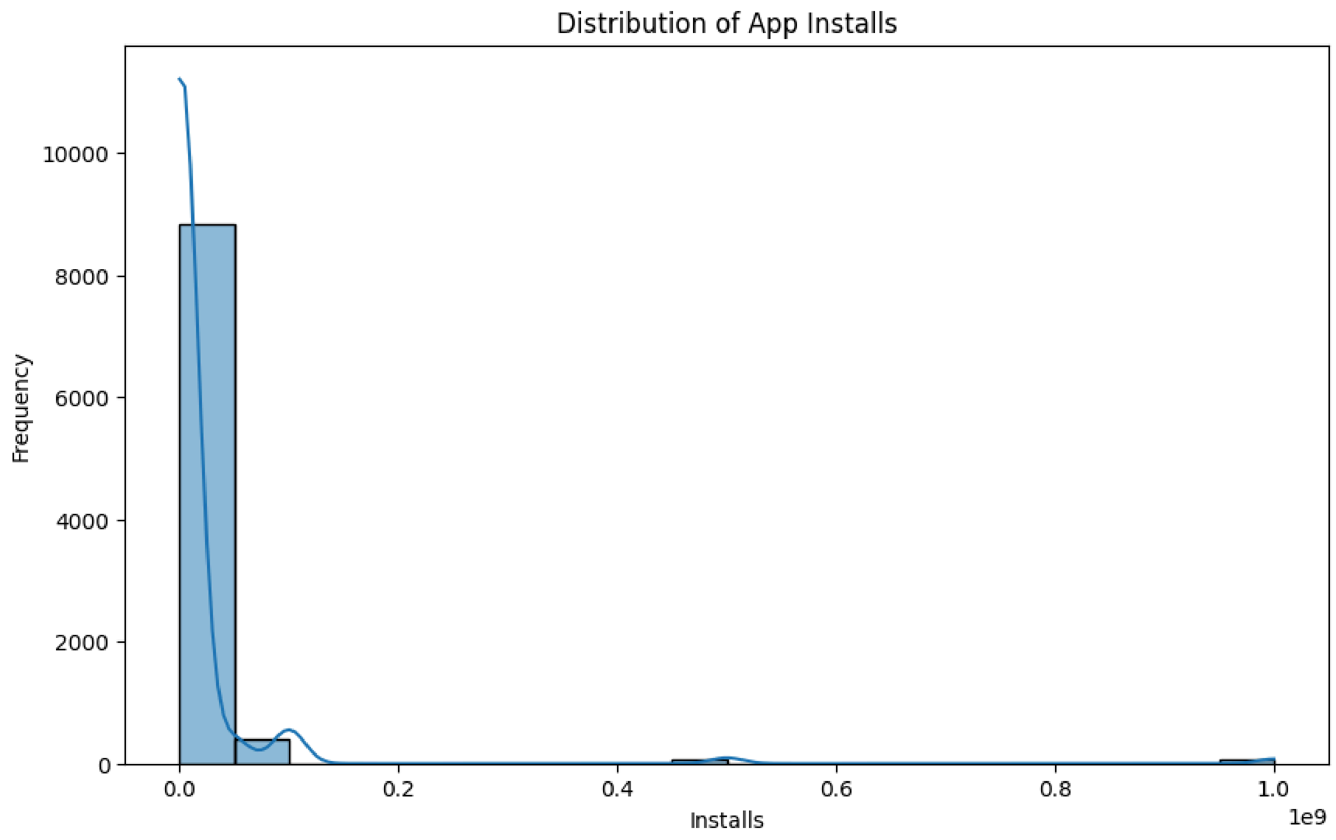
Below, we will perform both Univariate and Bivariate analyses to get a better understanding of our data. Using univariate analyses, we will uncover trends and information not readily available from looking at just the DataFrame. Our bivariate analyses will include hypothesis testing, running multiple hypothesis or permutation tests to determine correlations between any of the variables in our dataset.

EDA - Univariate Analysis

Now we can perform our EDA. For our first step, we will perform univariate analyses, starting with the distribution of app ratings.

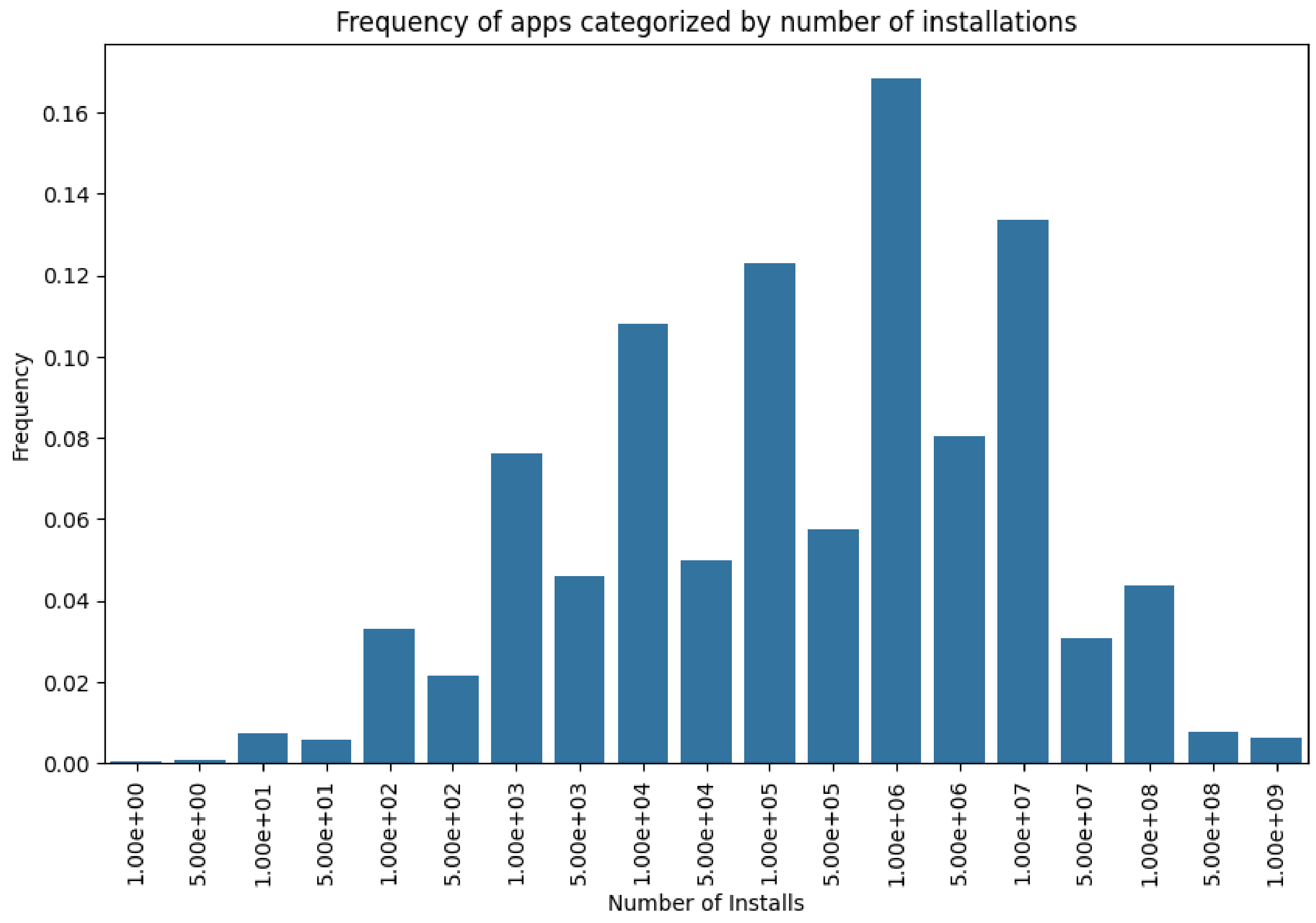


Based on our distribution, we can deduce that the ratings are not uniformly distributed, concentrated between 4 and 5 star reviews, and skewed to the left. This makes sense from a user point-of view, as many people are willing to give 3 or 5 star reviews when prompted to; not many people see the importance of the 4-star rating, or even anything thats 2-stars or below unless the app is poorly made. Now, let's look at the distribution of apps based on the number of installations.



Based on the plot above, we can see that the distribution of installations is skewed to the right, with most apps having less than 1 million installations. Why does our plot look so skewed? It turns out that the values in `Installs` are not accurate counts of the number of installations, but instead are **categories**, with each category increasing by a multiple of $.5e+$.

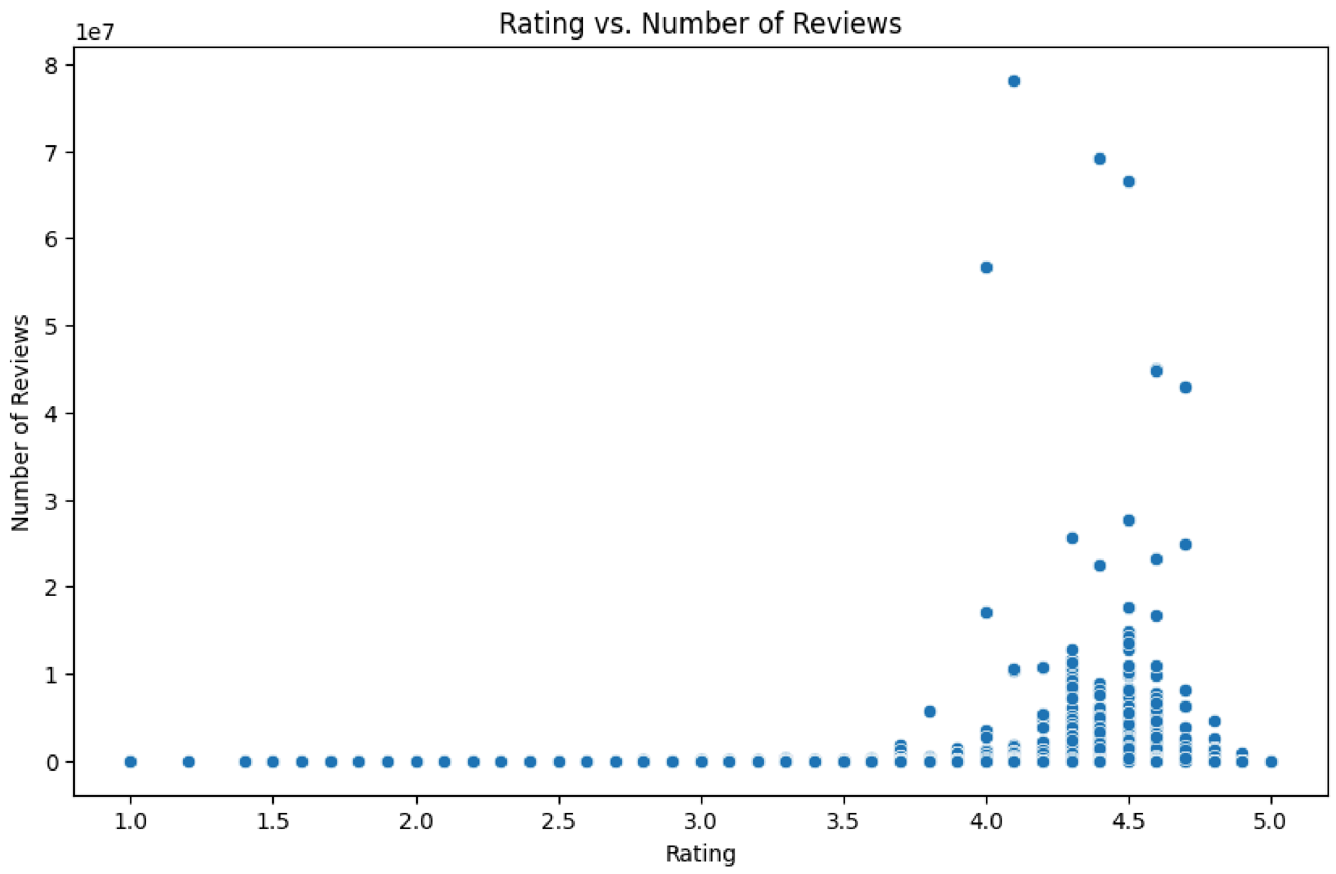
When working with skewed data distributions, outliers are detrimental to our hypothesis tests as well as for regression models.
What if we instead binned apps based on the number of downloads? By Changing the `Installs` to a categorical dtype and getting the number of apps in each category, we can create a plot that resembles more closely a normal distribution.



When converting the number of installs into a category, the data becomes more uniformly distributed. This could be useful for our analyses if we were looking to find a correlation between the distribution of new apps not found in our dataset. For now, we will move onto our bivariate analyses.

EDA - Bivariate Analysis

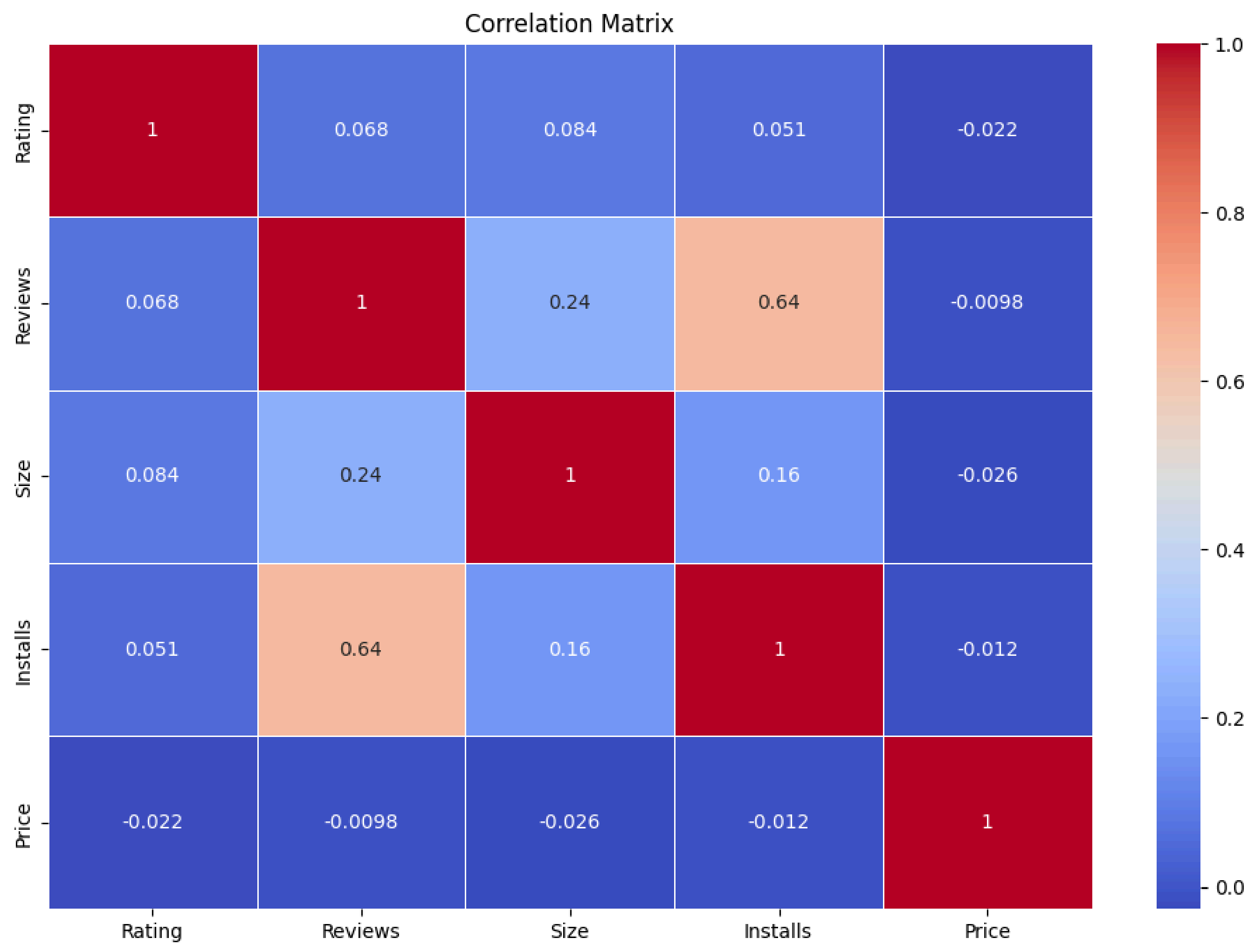
To start our analyses, we examined the relationship between app rating, and the total number of reviews



Looking at the plot above, we start to notice a few patterns between the variables:

- Apps that are rated low tend to have less reviews than apps with higher ratings. This is due to many reason, one of the more likely reasons is that people are put off by apps that have low ratings; therefore, there are less people to try the app and give it a review.
- The above statement is also corroborated by the low installations for low-rated apps
- Successful apps (those with hundreds of millions of installations) do not correspond with the highest ratings. This is because there is a higher population percentage that is experiencing app flaws that are likely to make users submit negative reviews.

Instead of performing multiple multivariate analyses, we san perform a correlation matrix to determine if there is colinnearity between any of the quantitative variables. The results are shown in the matrix below:



Hypothesis Testing

Hypothesis Test #1

Now that we have deduced that there may be a correlation between reviews and installs, lets determine just how significant other links are. We will determine if there is collinearity between 2 different categories based on `Rating`. We start by answering the following question:

> Is there a correlation between the distribution of ratings based on the catgories `COMMUNICATION` and `GAME`?

We will answer this question using the following hypotheses:

> Null Hypothesis H_0: `COMMUNICATION` and `GAME` have similar mean values.
> Alternate Hypothesis H_a: `COMMUNICATION` and `GAME` do not have the same means.

Category	count	mean	std	min	25%	50%	75%	max
COMMUNICATION	328.0	4.158537	0.426192	1.0	4.0	4.3	4.4	5.0
GAME	1097.0	4.286326	0.365375	1.0	4.1	4.4	4.5	5.0

Under the following hypotheses, we will perform Welch's 2-tailed t-test to determine if the two distributions have a difference in means. Welch's t-test is useful for determining whether 2 samples come from the same distribution, and when the sample sizes are different.

```
p-value: 1.2095995057771278e-06
Reject H_0
```

Hypothesis test #2

Since our null hypothesis was rejected when testing for difference in means, let's determine if there are any correlations between two categories using the Komolgorov-Smirnov Test.

> Is there a correlation between the distribution of ratings based on the catgories `COMMUNICATION` and `GAME`?

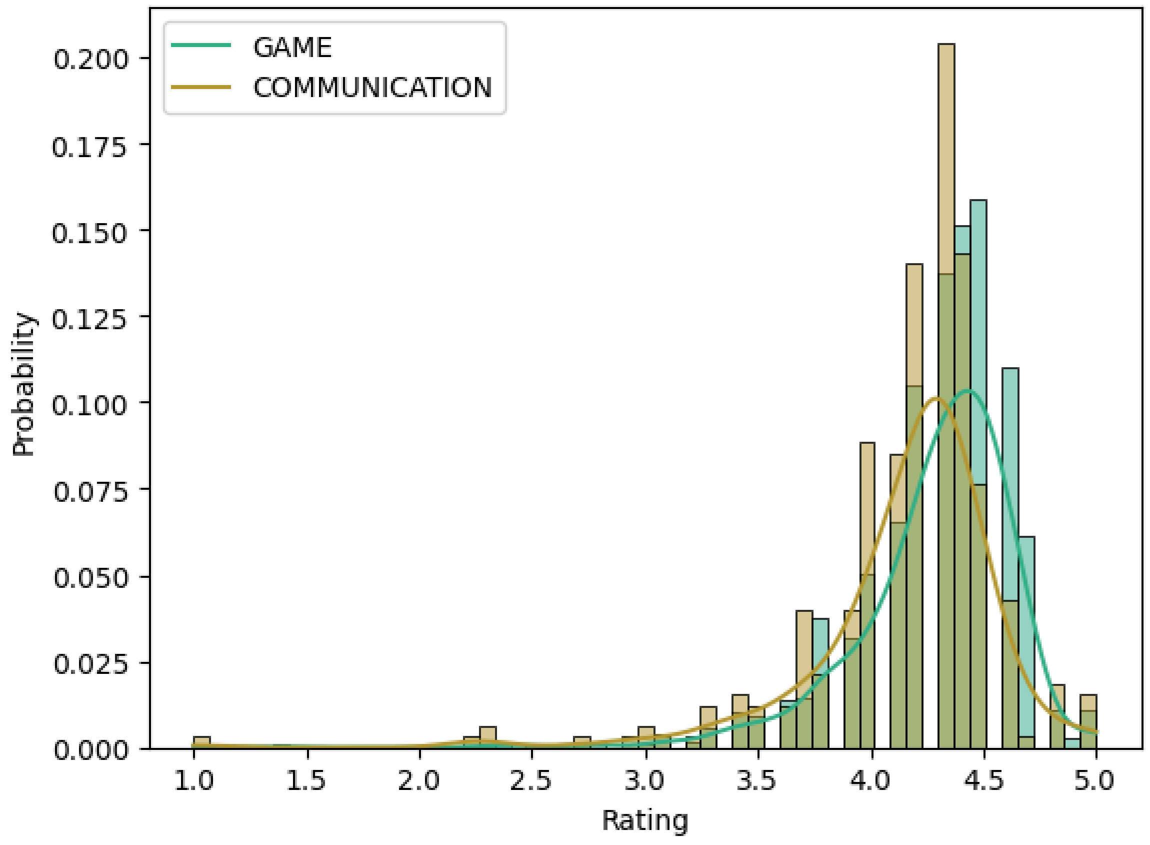
We will answer this question using the following hypotheses:

> Null Hypothesis H_0: `COMMUNICATION` and `GAME` come from the same distribution.
> Alternate Hypothesis H_a: `COMMUNICATION` and `GAME` do not come from the same distribution

```
p-value: 5.651039581432086e-10
Reject H_0
```

Once again, out test statistic reveals that there is a statistical significance in the difference in distributions between our two observed categories. We reject the nul hypothesis and claim that the two distributions are indeed not comming from the same distribution.

Below we plot the normalized distributions for `GAME` and `COMMUNICATION`



We can deduce the following from the 2 hypothesis tests we performed as well as the plot shown above:

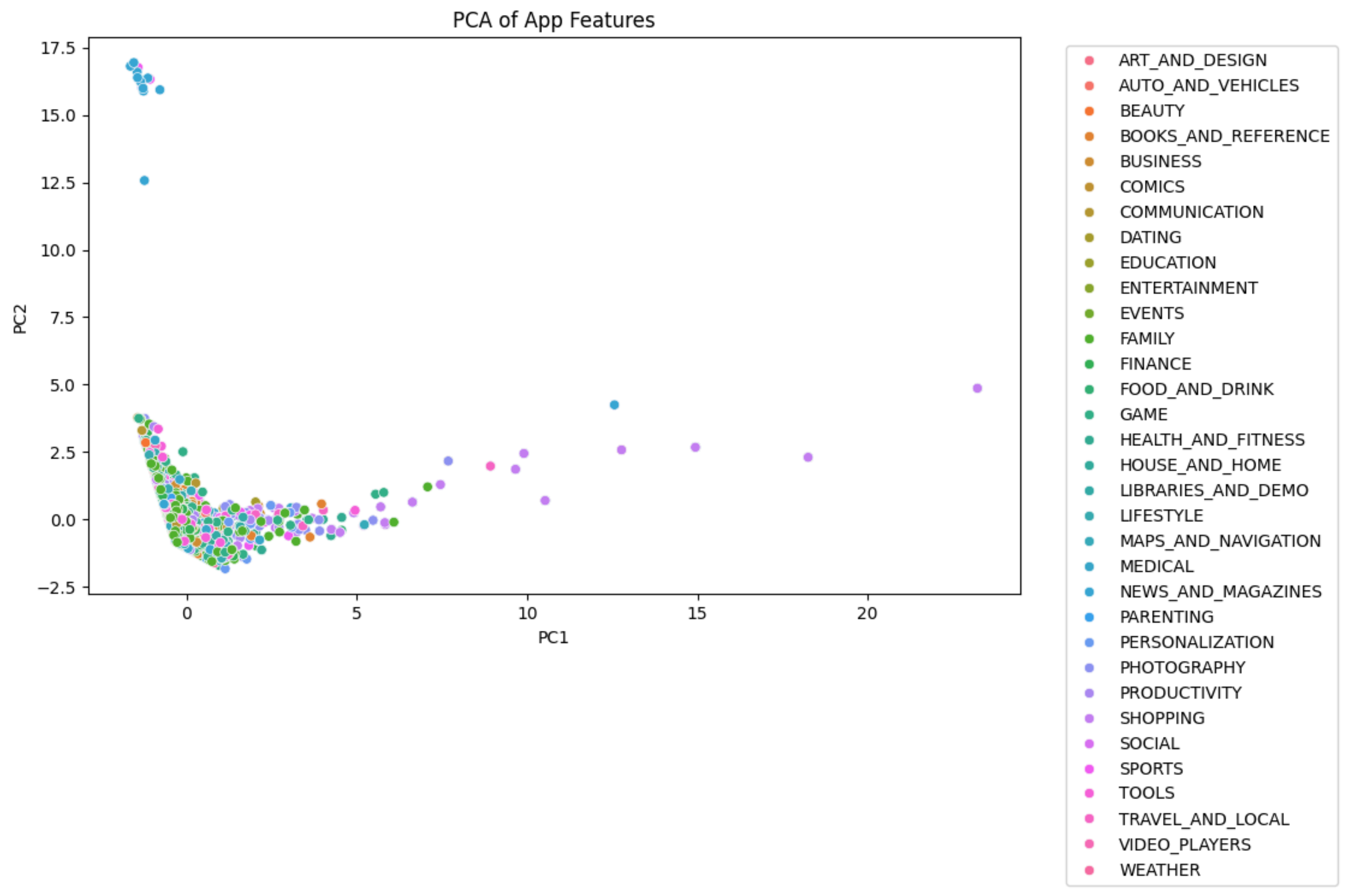
- Although visually the two distributions seem very close, our sample sizes were large enough that this difference was statistically significant.
- The variance of `COMMUNICATION` is significantly greater than variance of `GAME`, resulting in a CDF plot showing the distribution of `COMMUNICATION` as "tail heavy"
- The correlation matrix had a coefficient of 0.051 for the `Rating` and `Installs` cell. These hypothesis tests both stengthen the validity of the claim that these two variables are uncorrelated.

For our last bivaruate analysis, We will perform a chi-squared contingency test to determine if our dataset can pass a goodness of fit test.

```
Chi-square statistic: 0.0
p-value: 1.0
```

Goodness of Fit Test: This test evaluates how well the observed frequencies align with the expected frequencies under a specified null hypothesis. A significant chi-square statistic suggests a poor fit.

PCA



PCA: The scatter plot shows distinct clustering of app categories in the principal component space, indicating the ability of PCA to reduce dimensionality while preserving category separation.

```
Lasso coefficients: [ 0.          0.53460298  0.          0.          0.         -0.        ]
Ridge coefficients: [ 0.00000000e+00  5.44602263e-01  4.38235335e-08  4.84133992e-08
  2.11122443e-09 -1.34627418e-08]
Elastic Net coefficients: [ 0.          0.53691838  0.          0.          0.         -0.
]
```

Regularization: Lasso, Ridge, and Elastic Net regressions show that the number of reviews is the most influential factor on app ratings. The coefficients for other variables are shrunk to zero, indicating their lesser impact.

```
Optimization terminated successfully.
         Current function value: 0.505084
         Iterations 11
                           Logit Regression Results                           
==============================================================================
Dep. Variable:             HighRating   No. Observations:                 7723
Model:                          Logit   Df Residuals:                     7718
Method:                           MLE   Df Model:                            4
Date:                Sun, 09 Jun 2024   Pseudo R-squ.:                 0.06976
Time:                        19:29:04   Log-Likelihood:                -3900.8
converged:                       True   LL-Null:                       -4193.3
Covariance Type:            nonrobust   LLR p-value:                2.705e-125
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.8620      0.038     22.391      0.000       0.787       0.937
Reviews     1.023e-05   8.49e-07     12.054      0.000    8.57e-06    1.19e-05
Size           0.0018      0.001      1.333      0.182      -0.001       0.004
Installs   -1.421e-08   3.26e-09     -4.364      0.000   -2.06e-08   -7.83e-09
Price         -0.0021      0.001     -1.577      0.115      -0.005       0.001
==============================================================================
```

Logistic Regression: The model indicates that the number of reviews significantly predicts high ratings ($p < 0.001$). However, size and price do not significantly affect the likelihood of an app receiving a high rating.

Results

Based on our analyses, we can conlude the following statements:

- The number of reviews correlates highly with positive ratings, which in turn corresponds with higher installations.
- While the number of apps with low installations are very sparse, many apps with low installations also have low ratings. This strengthens our claims in the begining of our assessment that user are put off by poorly rated apps.
- The biggest apps -those with the highest number of installations- are not the best rated apps. It is easier for an app with a few thousand ratings to keep its high rating, whereas there are millions of people who have the potential to be disgruntled by a top app that is not performing well for them.
- `GAME` and `COMMUNICATION` apps are some of the biggest apps published in the Google Play Store. While `GAME` apps are some of the most abundant apps, `COMMUNICATION` apps generate the most installations out of any other category; however, There was very little correlation between their userbase in terms of ratings and reviews.

Recommendations

- Focus on User Experience: Ensuring high app quality and user satisfaction is crucial for achieving higher ratings and more installations.

- Strategic Category Selection: Developers should consider targeting popular categories with high user engagement potential.
- Monetization Strategies: Free apps can achieve widespread adoption, but developers should explore various monetization strategies to convert high installation numbers into revenue.

Conclusion

Our analysis provides valuable insights into the factors that contribute to an app's success on the Google Play Store. By focusing on user experience, strategic category selection, and effective monetization strategies, app developers can enhance their chances of success in a competitive market.