
TECHNICAL PROJECT REPORT

# PREDICTING ANDROID APP SUCCESS

GROUP 8

GROUP MEMBERS:

Arush Rao

Rohit Yadav

SriHarsha Marella

## CONTENTS

# 1. EXECUTIVE SUMMARY

In this rapidly growing world of mobile applications, one of the major players is the Google Play Store. With the number of apps being developed and published on the Play Store every day, it is becoming increasingly difficult for developers to understand what factors could contribute to the success of an app and for publishers to understand what kind of apps to invest in. The business objective is to assess all the attributes of an app which contribute to higher ratings and installations to improve business decisions mentioned before, by carrying out accurate predictions using insights received from the data being analyzed.

The goal of this project was to predict whether an app would be successfully or not based on its metadata. We ran an analysis on almost 10000 Android apps to help us comprehend the attributes contributing to the success of an app.

After exploring Machine Learning and Data mining algorithms throughout this course, we chose to run Decision Tree, Random Forest and KNN algorithms on our data set. The Random Forest algorithm yielded the highest accuracy of 80.07%.

# 2. BACKGROUND / CONTEXT

## a. Domain

The domain for our project is the Android Application Industry. We look at the insights from Google Play, digital distribution service operated and developed by Google. It serves as the official app store for certified devices running on the Android operating system, allowing users to browse and download applications developed with the Android software development kit (SDK) and published through Google. Google Play also serves as a digital media store, offering music, books, movies, and television programs.

## b. Brief description of the scenario

It is difficult for companies in the app industry to decide what genre of Apps they should focus on, the developers they should sign on and what kind of apps they should publish to the platform. We aim to help in contributing to the decision-making process by finding patterns in the app data over the last decade.

## c. Decision(s) of interest

Key decisions such as 'Investing in an app' and 'Promoting, producing of a certain type/category of app'. Based on this, what type of apps should be made free and what apps should be paid, would be the decisions of interest here.

## d. Decision maker(s)

There will be a couple of decision makers in this case:

i. Google i.e. the business itself which would use this model to promote apps of a popular/trending category or regulate the promotions of less popular genres. This would also help them sign deals with publishing companies focusing on categories which are currently in demand, based on the analysis.

ii. The second decision maker would be the app developers and publishers themselves who are associated with Google Play. This would be key in determining the kind of apps they develop and the renowned publishers from a specialized domain, with which they collaborate.

# 3. DATA UNDERSTANDING

## a. Data requirements

We would require a large dataset with popular and non-popular apps since we intend to analyze trends for the entire decade. This would mean gathering data from every year's published apps. A comprehensive dataset from the Google Play digital platform that highlights the apps that have been on the store and gives us insights on the popularity of a category, features and attributes (metadata) of the apps, rating details, installs and other details which might help us in establishing correlations and help us view patterns in these varied apps.

| **Business Objective** | **Data Requirement** |
|---|---|
| Could we really predict the success of an app? | Existing app data classified as successful or not |
| Does Genre matter? | Genre data for the corresponding app data |
| What features contribute to an app being successful? | Feature Metadata for the corresponding app data |

## b. Describe data

The dataset we are using is a raw dataset of the app data starting from 2010 to 2019. It has a collection of metadata features for each app such as App name, Rating, Reviews, Size, Installs, Type, Content Rating, Genres, etc. It also gives us an overview about the app such as the year it was last updated, the current version of the app and the minimum Android version it is compatible with.

| | |
|---|---|
| **App** | Application name |
| **Category** | Category the app belongs to |
| **Rating** | Overall user rating of the app |
| **Reviews** | Number of user reviews for the app |
| **Size** | Size of the app |
| **Installs** | Number of user downloads/installs for the app |
| **Type** | Paid or Free |
| **Price** | Price of the app |
| **Content Rating** | Age group the app is targeted at - Children / Mature 21+ / Adult |
| **Genres** | An app can belong to multiple genres (apart from its main category) |
| **Last Updated** | Date when the app was last updated on Play Store |
| **Current Ver** | Current version of the app available on Play Store |
| **Android Ver** | Min required Android version |

## c. Sources:

We obtained our dataset from Kaggle which was originally scraped from the Google Play Store. It is the web scraped data of 10k Play Store apps for analyzing the Android market. It consists of in total of 10841 rows and 13 columns.

https://www.kaggle.com/lava18/google-play-store-apps

# 4. DATA PREPARATION

## a. Data Cleaning

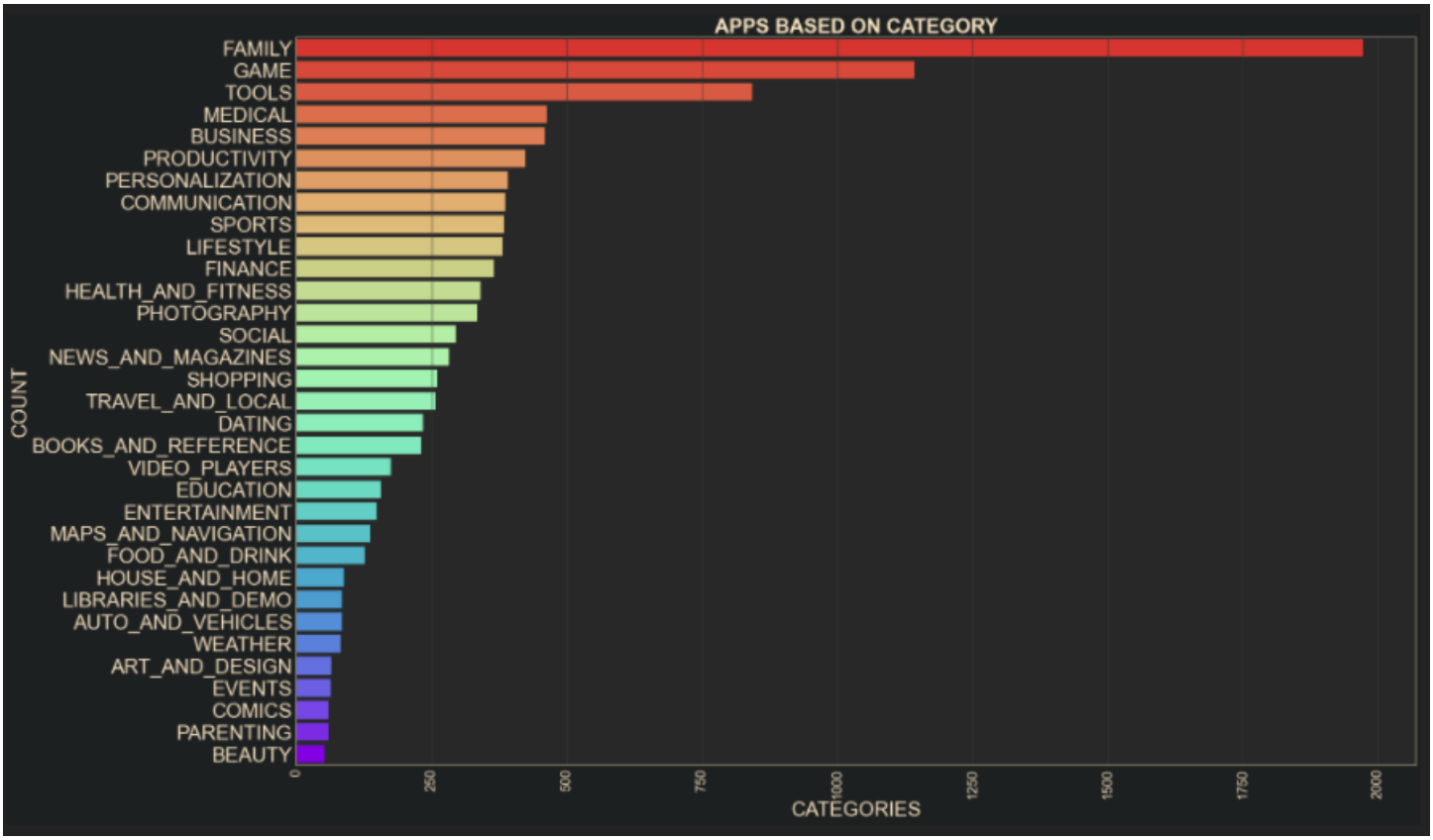
We start off by checking our dataset for any N/A's and find that there are a total of 1487 null values. As we see, there are N/A values in the '**Rating', 'Type', 'Content Rating', 'Current Ver'** and '**Android Ver'** columns. We eliminate these records as they may cause inconsistencies in our modelling. We also summarize the rating column to check for any outliers and find that the maximum value is 19 which clearly is an outlier. We eliminate any value greater than 5 which exists in the rating column as the rating on the Play Store exists only on a scale of 1-5. We also eliminate any special characters such as '**+**', '**$**' etc. in fields such as '**Installs**', '**Price**' etc. as these are numeric columns and any symbol would be irrelevant for our analysis.

```
In [6]: # Checking for NULL values in all the columns
        print(df.isnull().sum())
        #df[df.isnull()]
        #df[df.Rating>5]
        df[df.isnull().any(axis=1)]
```

```
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
```

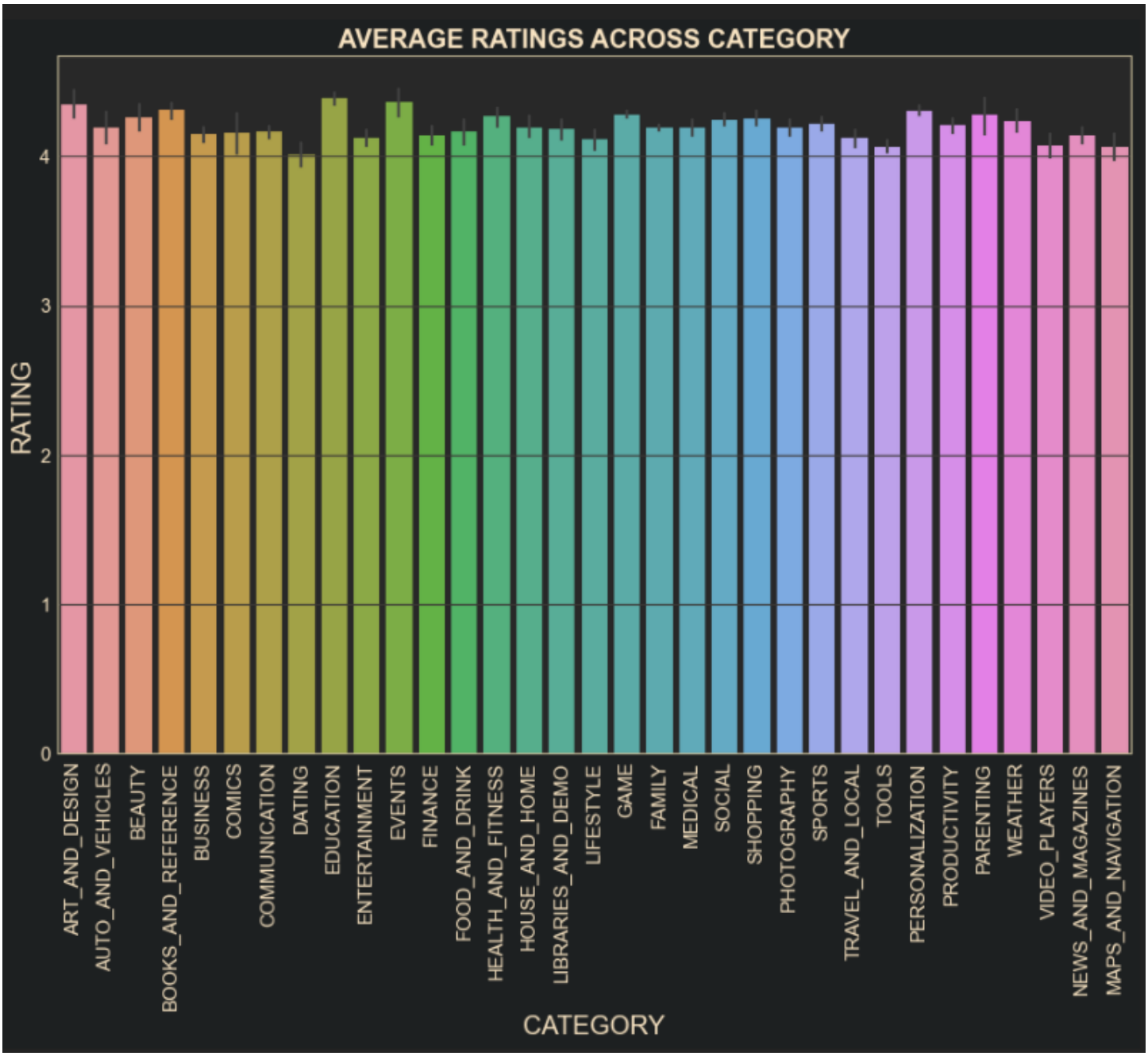
## b. Data Visualizations

We start off by visualizing the various categories of apps and have a look at the number of apps each category has.

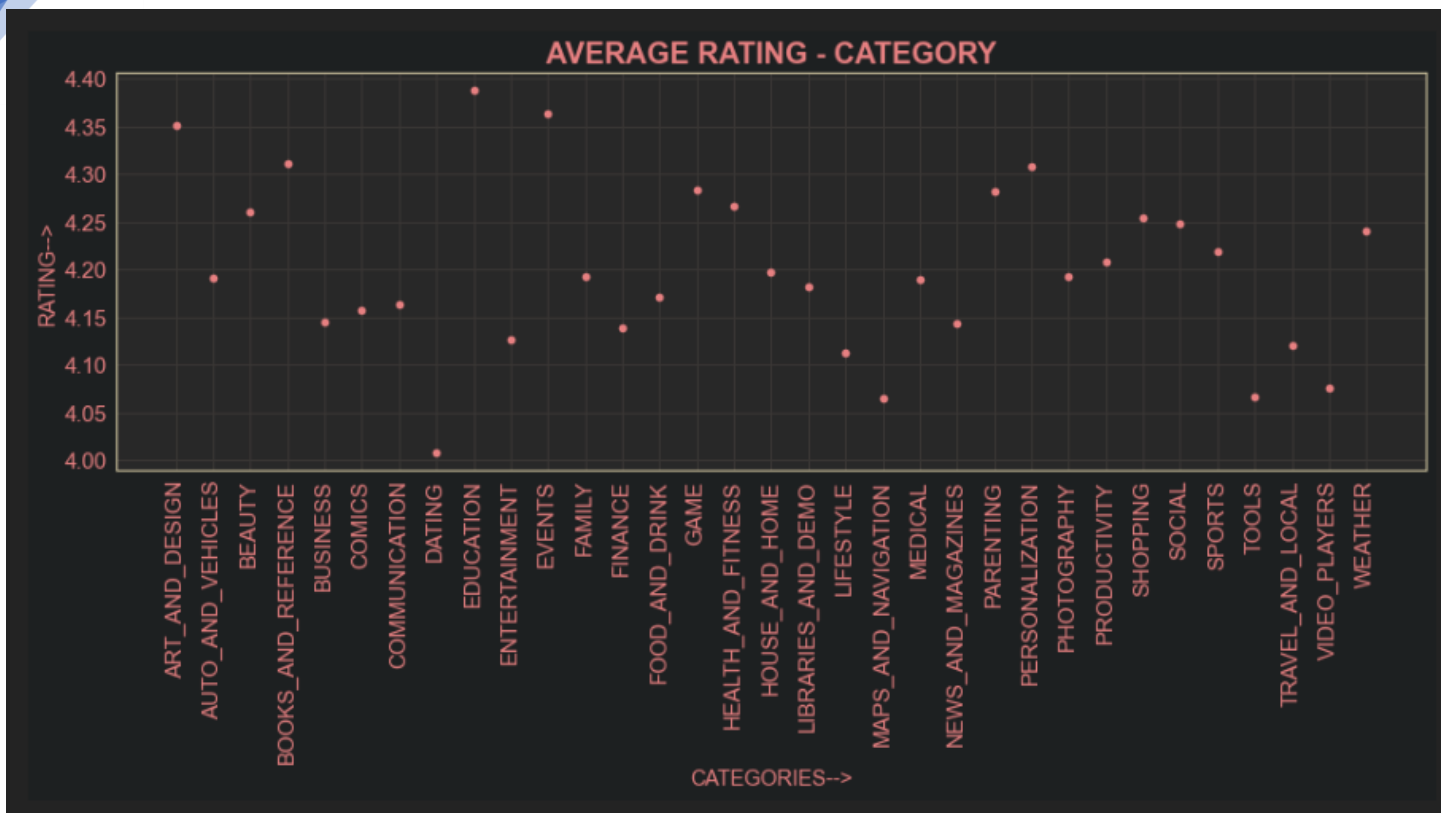



Based on our plot, we see that the ‘Family’ category has the maximum number of apps followed by ‘Game’, ‘Tools’ and ‘Medical’. The ‘Beauty’ category has the lowest number of apps.

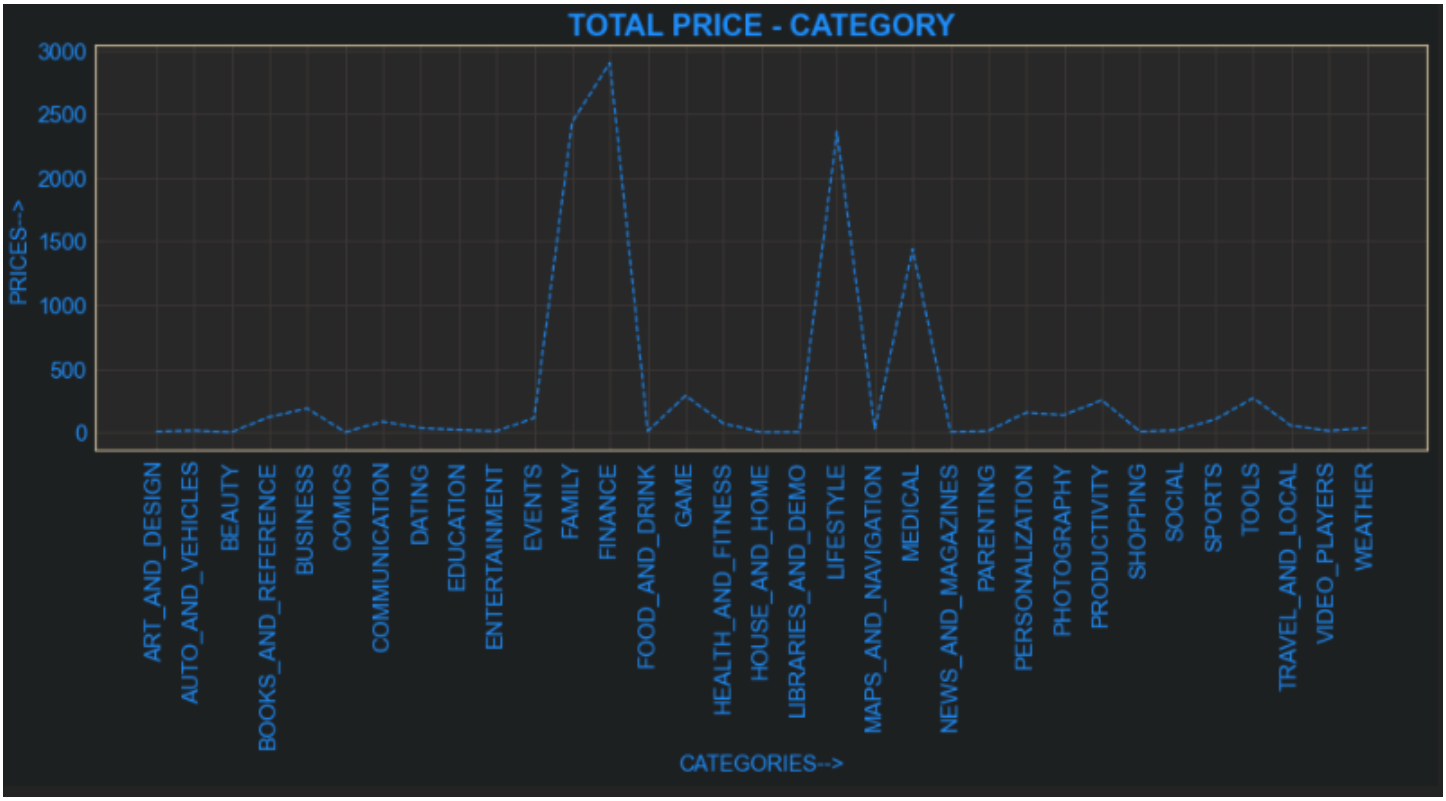
Next, we have a look at the average rating across each category of apps.



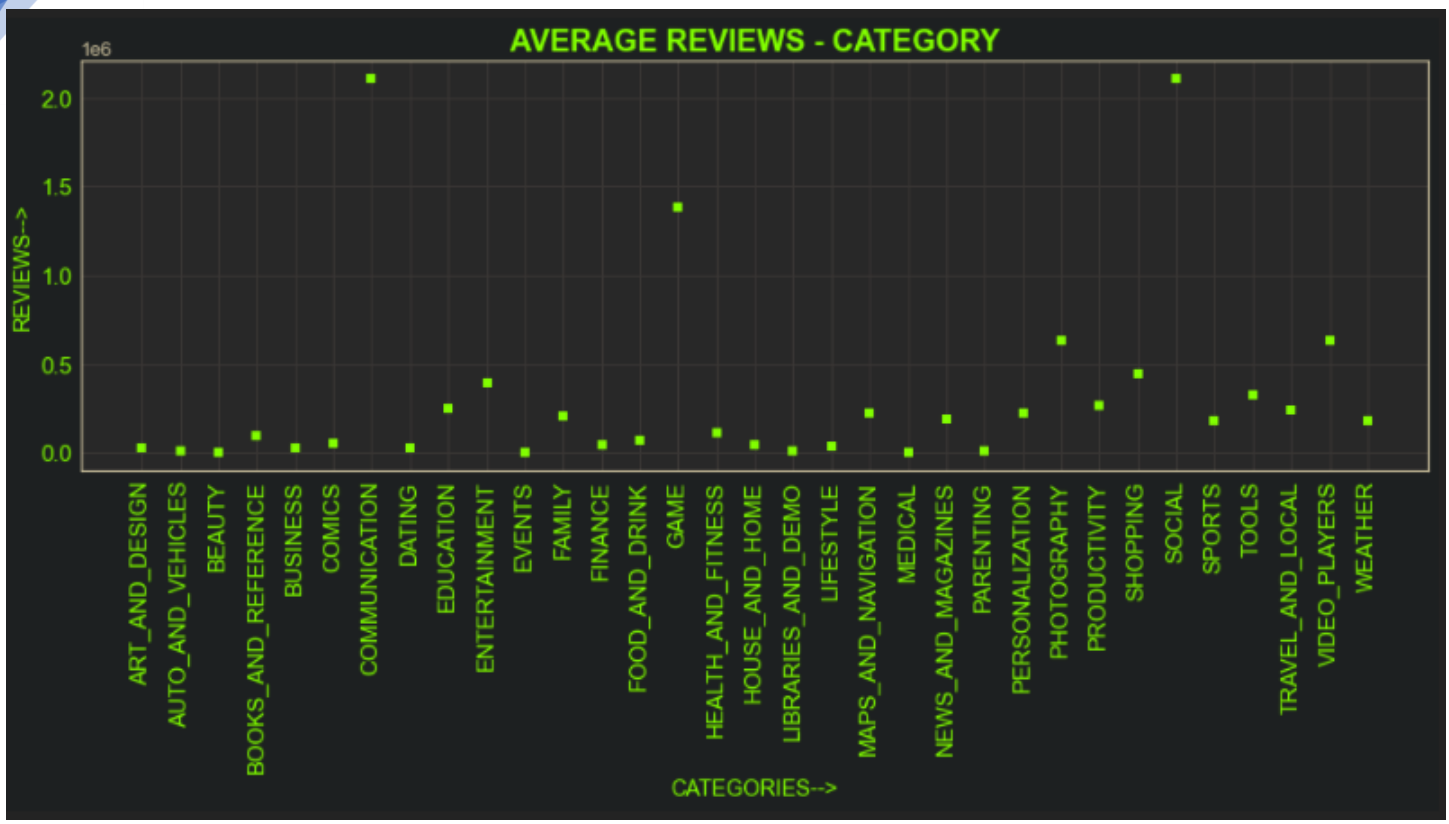

Next, we consider the ratings across categories on a scale of 1-5 and see that the ratings are all extremely close to each other on a scale of 1-4.



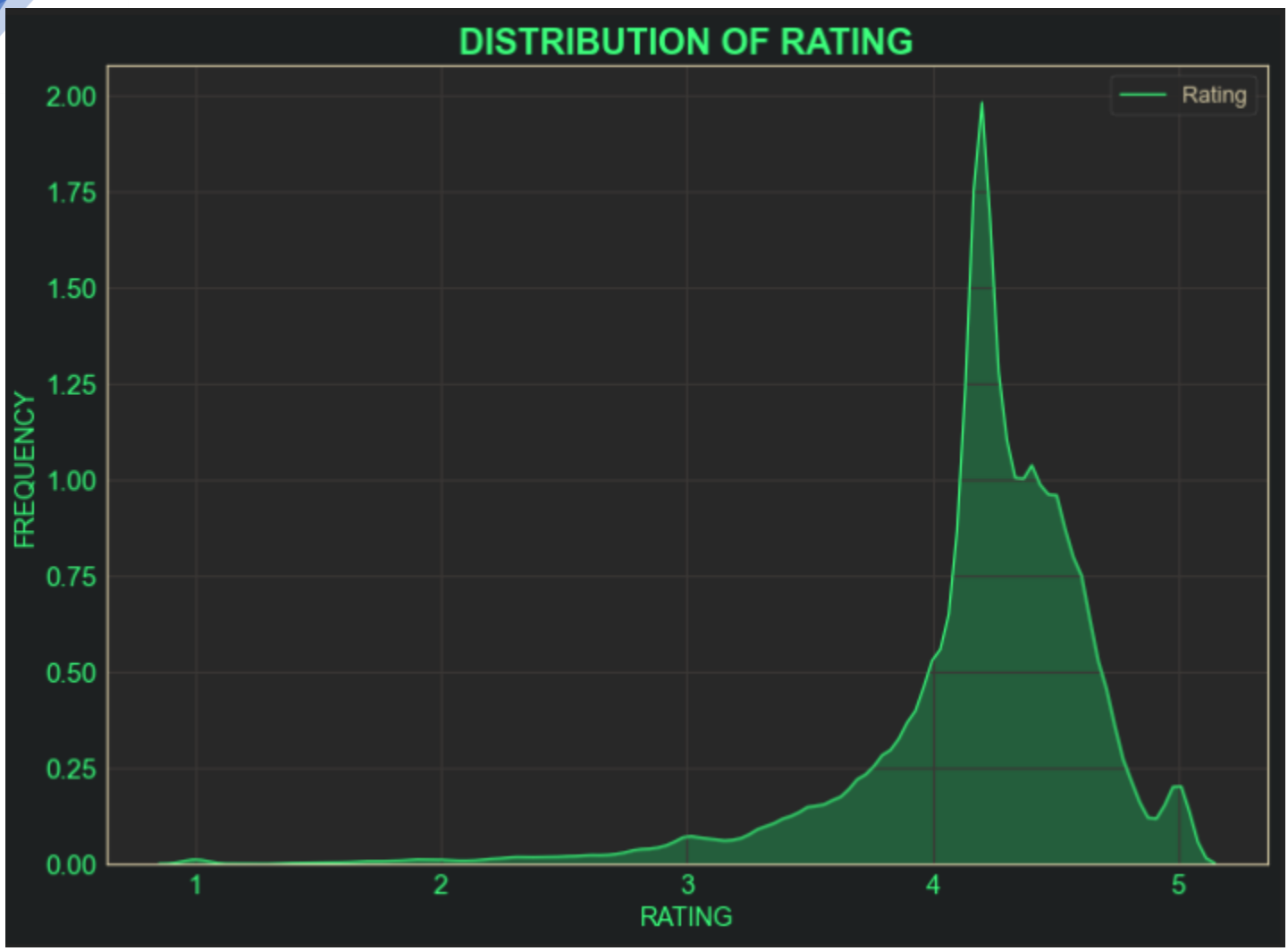

We see that the 'Education' category has the maximum average rating across all categories followed by 'Events' and 'Art and Design'. Note that we only look at ratings spread across 4 - 4.4.



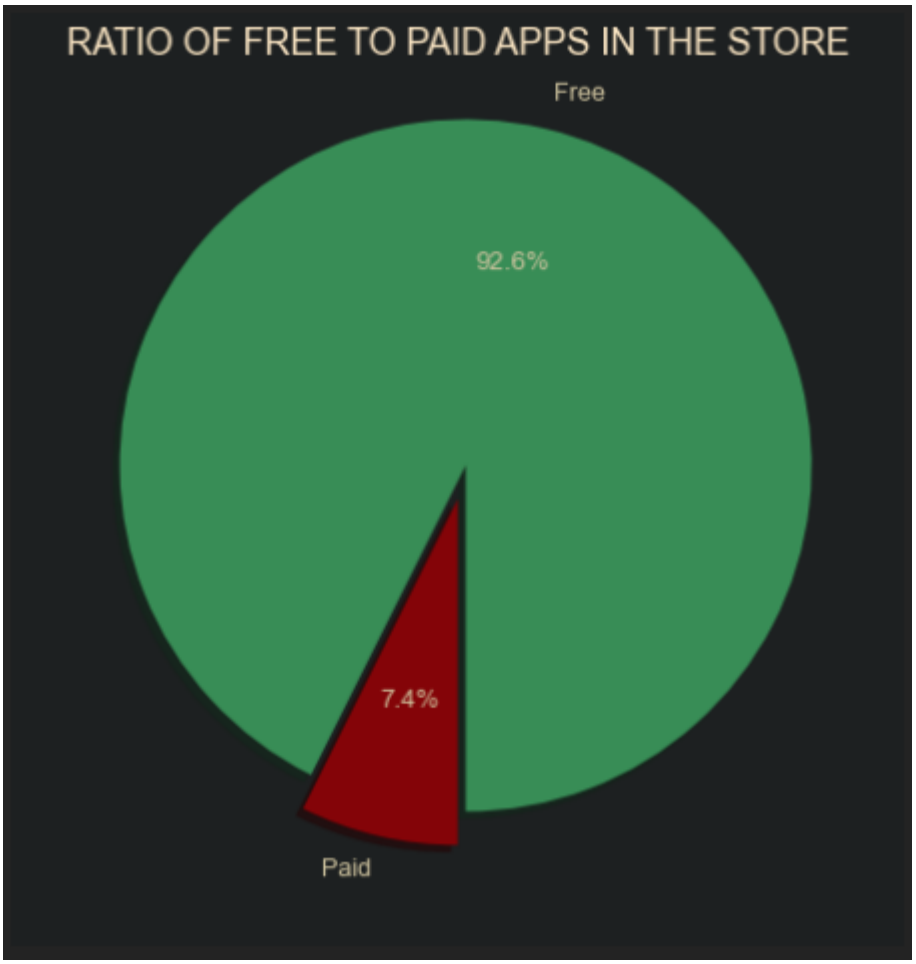

We see that the Events and Family category of apps have the highest prices followed by Lifestyle.




We plot the average reviews across each category and see that the **'Communication'** and **'Social'** categories have the maximum number of reviews as compared to any other category followed by the **'Game'** category.

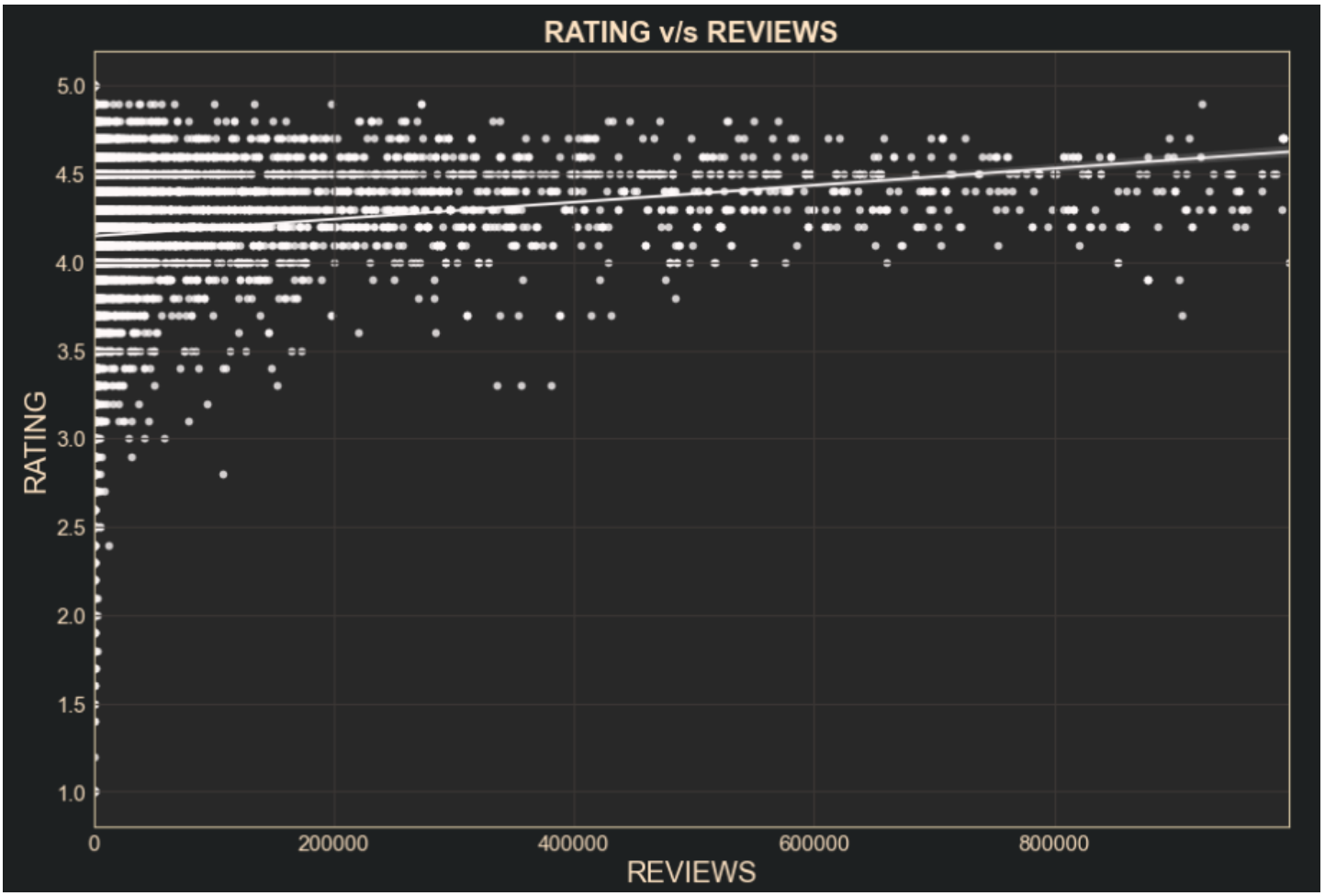



We see that the average rating across majority of the apps are closer to 4 and between 4 & 5. This indicates that our dataset consists of most apps that have a good rating since the apps are being rated on a scale of 1-5.

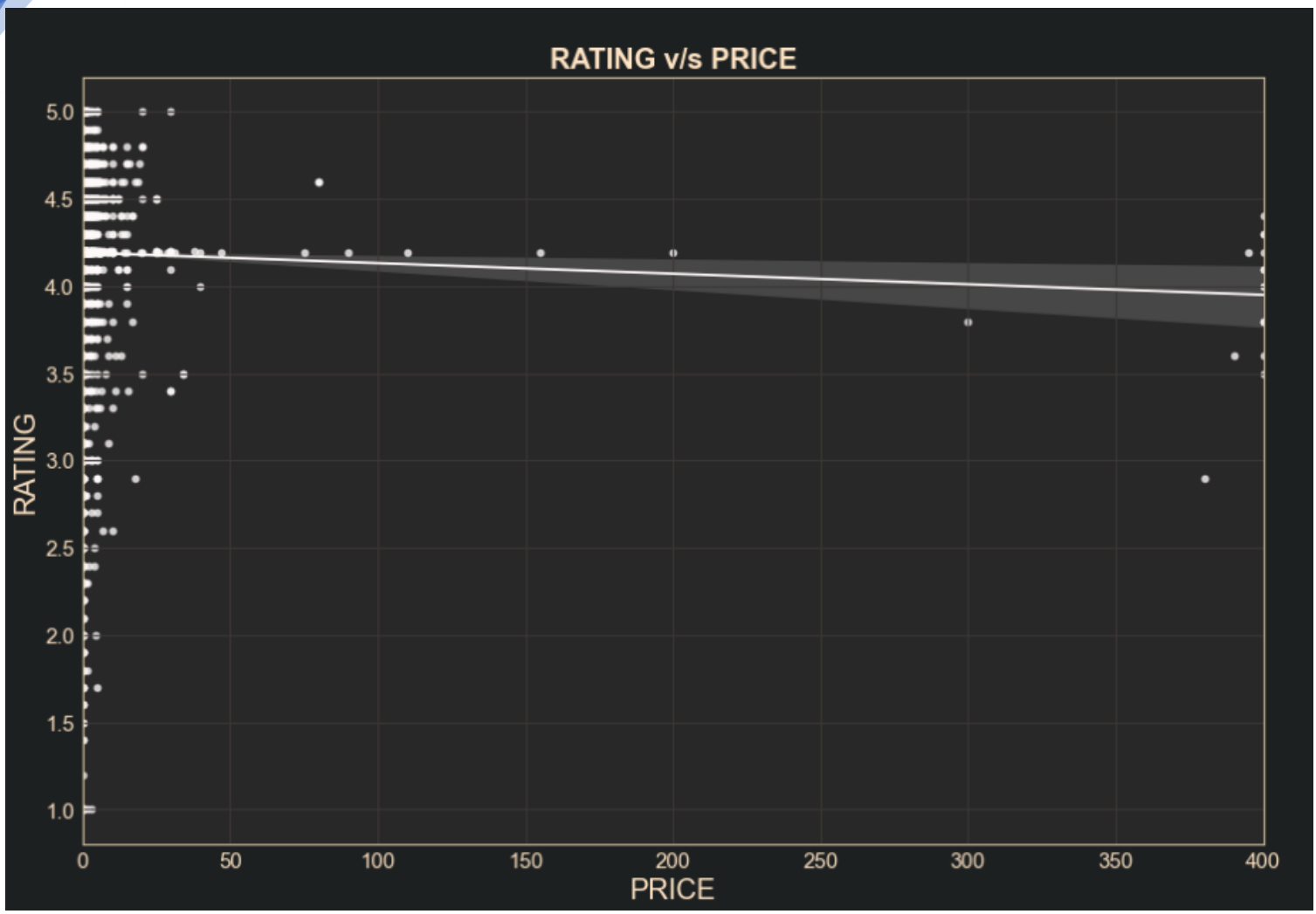



We see that 92.6% of the apps in our dataset are Free apps indicating that majority of the apps are free and only 7.4% of the apps in our dataset are paid apps.

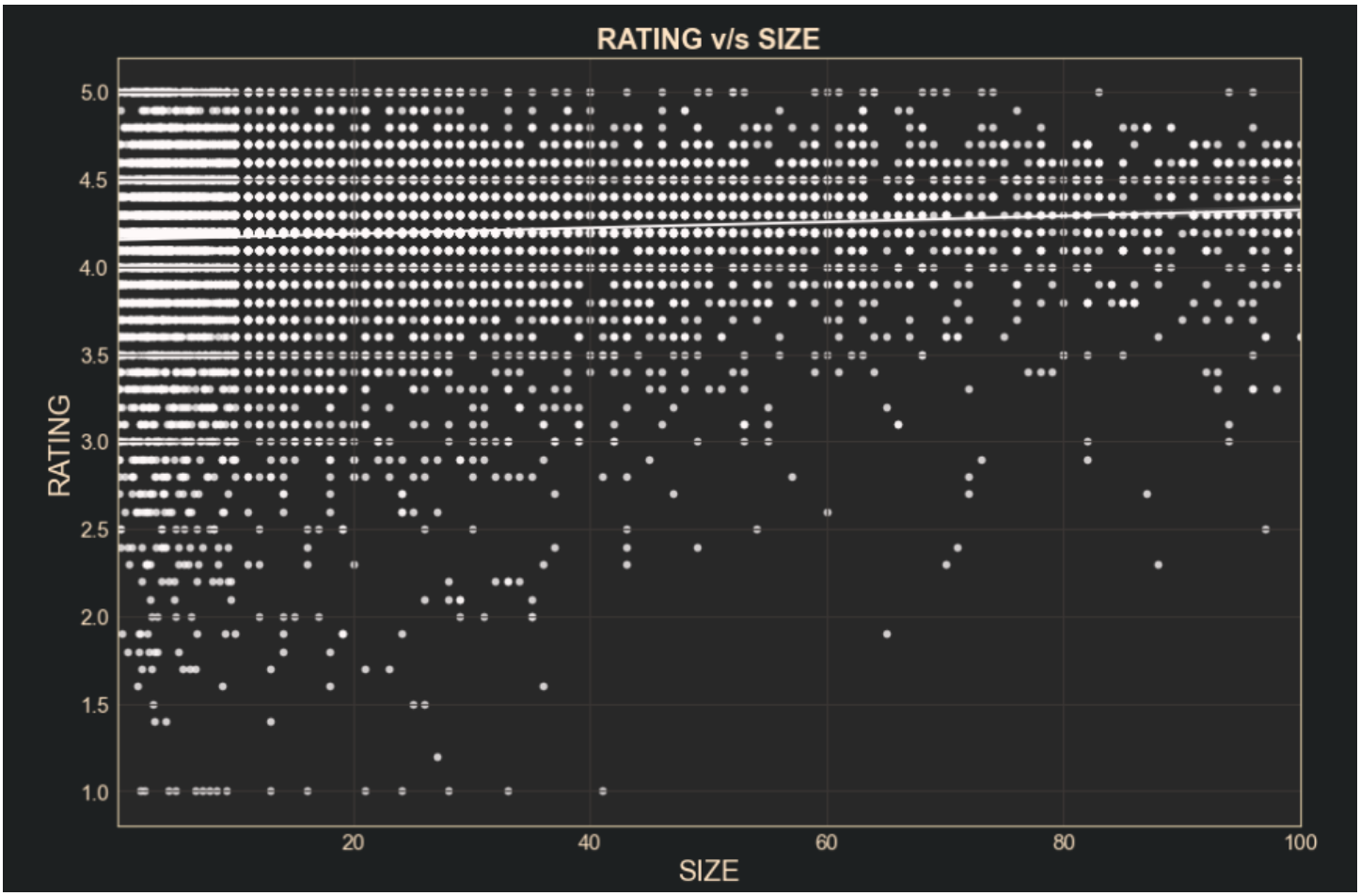
## Regression Plots



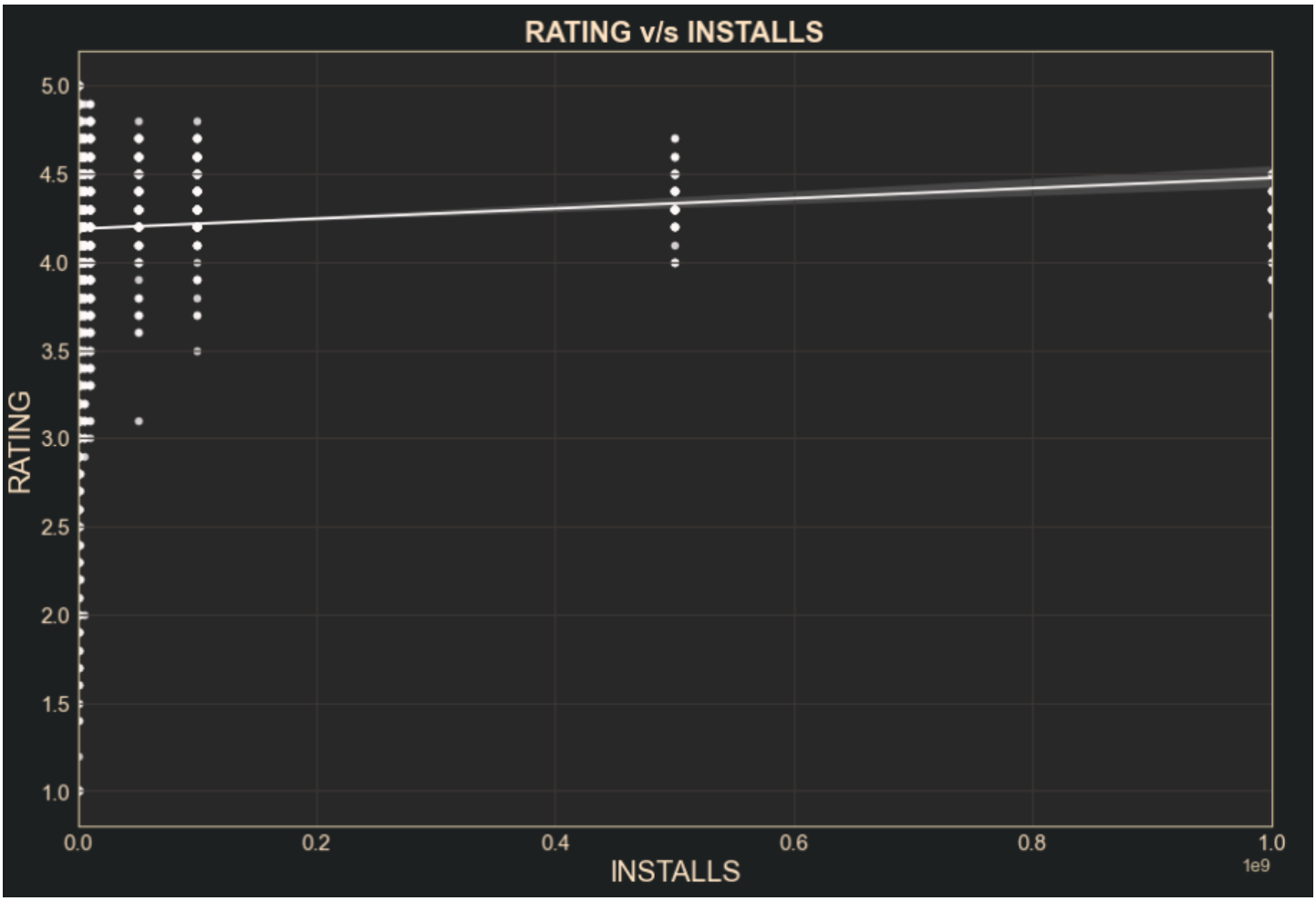

Based on the regression plot, we see that apps, with a higher rating usually tend to get more reviews as compared to low rated apps. This indicates that popular applications attract more users to review them.



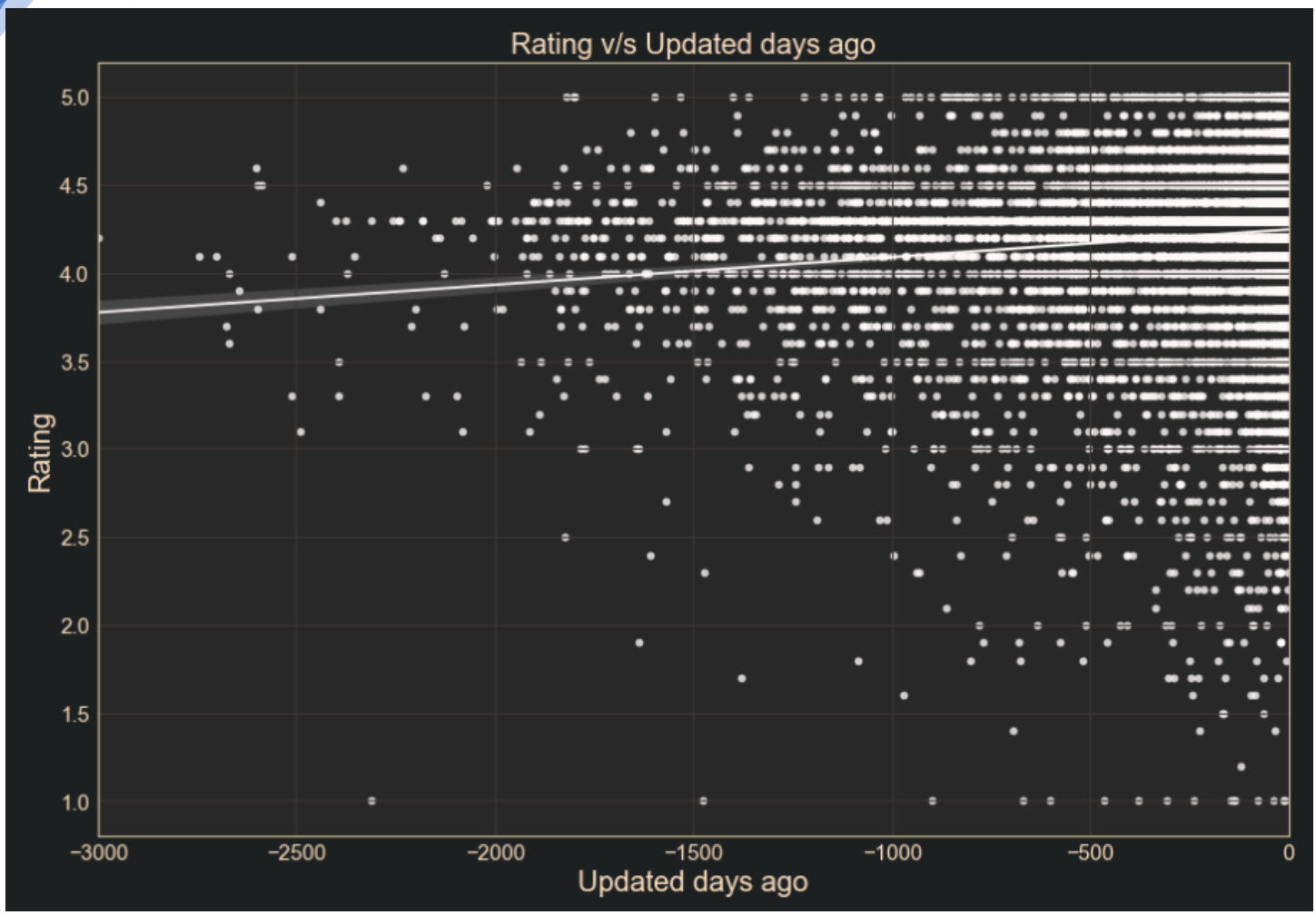

Here, we see that as the price of an app increases, the rating of the app decreases which also holds true in-reality. Users prefer free apps and tend to get reluctant to use them which results in a reduced rating as the price increases. However, majority of the apps in our dataset are free apps, therefore this is insignificant.



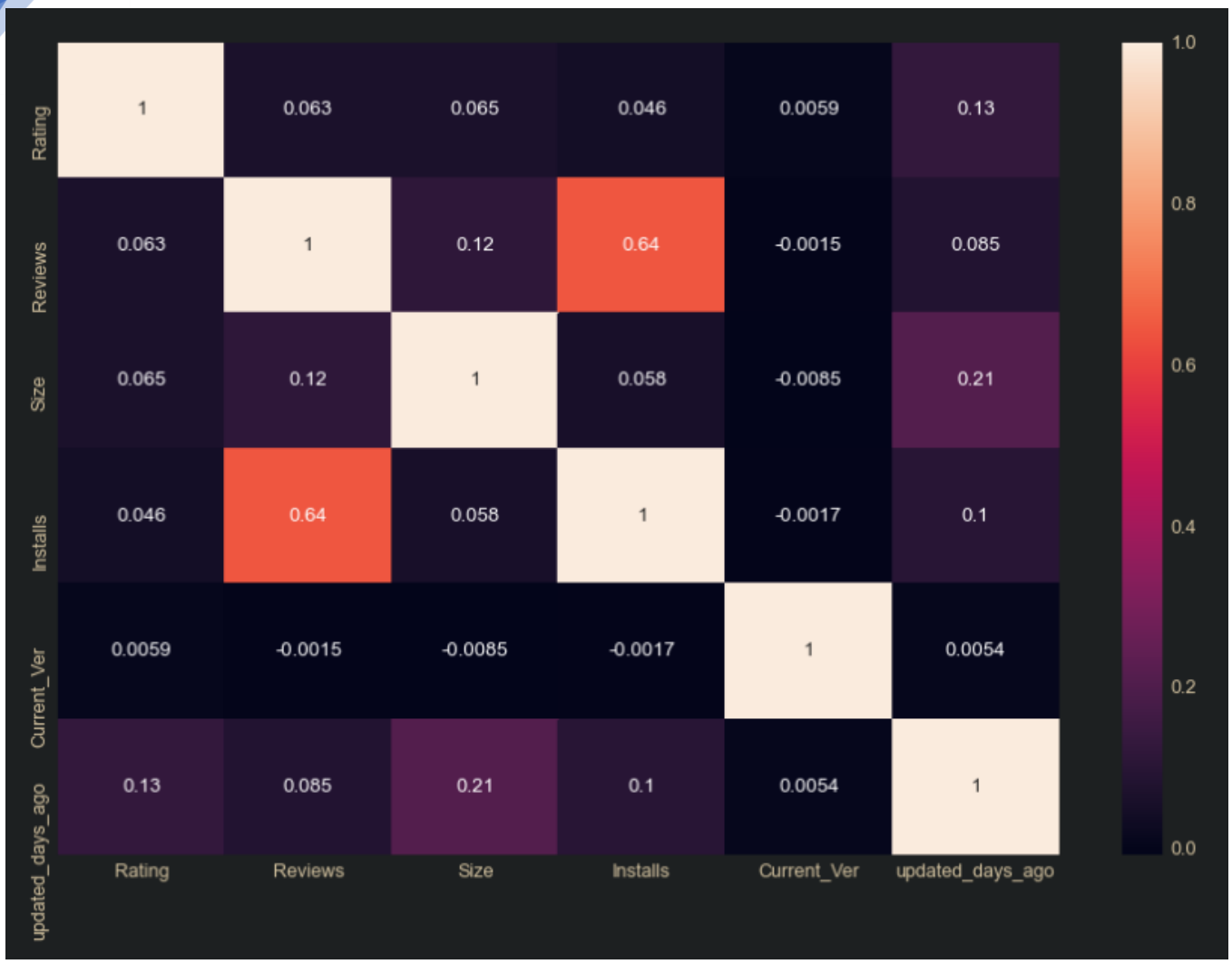

We see that there is a slight upward trend while considering the size of an app, but it is not significant enough for us to conclude that size does affect the rating of an app.




We see an upward trend when considering the number of installations of an app, as popular apps tend to get installed more which also results in an increase in the overall rating.




Based on this plot, we see that recently updated apps have better ratings as compared to apps that haven't been updated since a long time.




Based on this correlation plot, we see that the number of installs is highly correlated with the reviews. Our outcome variable, Rating, is not directly highly correlated with any of our explanatory variables.

## c. Prepare Data

For ease of understanding and usability, we make a few changes to the data structure of certain variables. We convert the '**Reviews'** (number of user reviews of an app) from object to float as it is a numerical column. We also change make a few changes to the '**Size**' column and represent all sizes in MB to maintain consistency. Before proceeding to build our models, we also eliminate variables like '**App(Name)', 'Genres', 'Last Updated', 'Current Ver', 'Android Ver'** as these variables do not affect the rating of an app. We consider the variables which affect our dependent variable Rating and drop the rest, mentioned above.

```
In [5]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
App                10841 non-null object
Category           10841 non-null object
Rating             9367 non-null float64
Reviews            10841 non-null object
Size               10841 non-null object
Installs           10841 non-null object
Type               10840 non-null object
Price              10841 non-null object
Content Rating     10840 non-null object
Genres             10841 non-null object
Last Updated       10841 non-null object
Current Ver        10833 non-null object
Android Ver        10838 non-null object
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

| | 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 |

We also one-hot-encode the category column as there are over 30 categories and it makes more sense to include a binary column for each category which would signify the category for each record/app. We one hot-encode the genre columns as well since there are many sub genres within the genres themselves and we prefer having the sub-genres under the main genres itself while building the models. We also have a type column for indicating whether the app is paid or free, we map this column as a binary indicator where 0

represents a free app and  represents a paid app. We follow a similar mapping procedure for the content rating column which signifies the age group the app is suitable for. After cleaning and making the required changes, our final data frame which we use for our modelling looks like this:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10837 entries, 0 to 10840
Data columns (total 85 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Rating                      10837 non-null  float64
 1   Reviews                     10837 non-null  int64
 2   Size                        10837 non-null  float64
 3   Installs                    10837 non-null  int64
 4   Current_Ver                 10837 non-null  float64
 5   updated_days_ago            10837 non-null  int64
 6   CAT_AUTO_AND_VEHICLES       10837 non-null  uint8
 7   CAT_BEAUTY                  10837 non-null  uint8
 8   CAT_BOOKS_AND_REFERENCE     10837 non-null  uint8
 9   CAT_BUSINESS                10837 non-null  uint8
 10  CAT_COMICS                  10837 non-null  uint8
 11  CAT_COMMUNICATION           10837 non-null  uint8
 12  CAT_DATING                  10837 non-null  uint8
 13  CAT_EDUCATION               10837 non-null  uint8
 14  CAT_ENTERTAINMENT           10837 non-null  uint8
 15  CAT_EVENTS                  10837 non-null  uint8
 16  CAT_FAMILY                  10837 non-null  uint8
 17  CAT_FINANCE                 10837 non-null  uint8
 18  CAT_FOOD_AND_DRINK          10837 non-null  uint8
 19  CAT_GAME                    10837 non-null  uint8
 20  CAT_HEALTH_AND_FITNESS      10837 non-null  uint8
 21  CAT_HOUSE_AND_HOME          10837 non-null  uint8
 22  CAT_LIBRARIES_AND_DEMO      10837 non-null  uint8
 23  CAT_LIFESTYLE               10837 non-null  uint8
 24  CAT_MAPS_AND_NAVIGATION     10837 non-null  uint8
 25  CAT_MEDICAL                 10837 non-null  uint8
 26  CAT_NEWS_AND_MAGAZINES      10837 non-null  uint8
 27  CAT_PARENTING               10837 non-null  uint8
 28  CAT_PERSONALIZATION         10837 non-null  uint8
 29  CAT_PHOTOGRAPHY             10837 non-null  uint8
 30  CAT_PRODUCTIVITY            10837 non-null  uint8
 31  CAT_SHOPPING                10837 non-null  uint8
 32  CAT_SOCIAL                  10837 non-null  uint8
 33  CAT_SPORTS                  10837 non-null  uint8
 34  CAT_TOOLS                   10837 non-null  uint8
 35  CAT_TRAVEL_AND_LOCAL        10837 non-null  uint8
 36  CAT_VIDEO_PLAYERS           10837 non-null  uint8
```

```
37  CAT_WEATHER                  10837 non-null  uint8
38  TYPE_Paid                    10837 non-null  uint8
39  GEN_Adventure                10837 non-null  uint8
40  GEN_Arcade                   10837 non-null  uint8
41  GEN_Art & Design             10837 non-null  uint8
42  GEN_Auto & Vehicles          10837 non-null  uint8
43  GEN_Beauty                   10837 non-null  uint8
44  GEN_Board                    10837 non-null  uint8
45  GEN_Books & Reference        10837 non-null  uint8
46  GEN_Business                 10837 non-null  uint8
47  GEN_Card                     10837 non-null  uint8
48  GEN_Casino                   10837 non-null  uint8
49  GEN_Casual                   10837 non-null  uint8
50  GEN_Comics                   10837 non-null  uint8
51  GEN_Communication            10837 non-null  uint8
52  GEN_Dating                   10837 non-null  uint8
53  GEN_Education                10837 non-null  uint8
54  GEN_Educational              10837 non-null  uint8
55  GEN_Entertainment            10837 non-null  uint8
56  GEN_Events                   10837 non-null  uint8
57  GEN_Finance                  10837 non-null  uint8
58  GEN_Food & Drink             10837 non-null  uint8
59  GEN_Health & Fitness         10837 non-null  uint8
60  GEN_House & Home             10837 non-null  uint8
61  GEN_Libraries & Demo         10837 non-null  uint8
62  GEN_Lifestyle                10837 non-null  uint8
63  GEN_Maps & Navigation        10837 non-null  uint8
64  GEN_Medical                  10837 non-null  uint8
65  GEN_Music                    10837 non-null  uint8
66  GEN_News & Magazines         10837 non-null  uint8
67  GEN_Parenting                10837 non-null  uint8
68  GEN_Personalization          10837 non-null  uint8
69  GEN_Photography              10837 non-null  uint8
70  GEN_Productivity             10837 non-null  uint8
71  GEN_Puzzle                   10837 non-null  uint8
72  GEN_Racing                   10837 non-null  uint8
73  GEN_Role Playing             10837 non-null  uint8
74  GEN_Shopping                 10837 non-null  uint8
75  GEN_Simulation               10837 non-null  uint8
76  GEN_Social                   10837 non-null  uint8
77  GEN_Sports                   10837 non-null  uint8
78  GEN_Strategy                 10837 non-null  uint8
79  GEN_Tools                    10837 non-null  uint8
80  GEN_Travel & Local           10837 non-null  uint8
81  GEN_Trivia                   10837 non-null  uint8
82  GEN_Video Players & Editors  10837 non-null  uint8
83  GEN_Weather                  10837 non-null  uint8
84  GEN_Word                     10837 non-null  uint8
```

We then proceed to split our data set into training and validation sets in a 80-20 ratio. As a pre-processing step, we also scale all the features so that the data is normalized, and we can have speedy calculations.

# 5. MODELLING – BUILDING DECISION SUPPORT MODELS

## i. Decision Tree

Decision trees are a popular classification method. Decision Tree algorithm belongs to the family of supervised learning algorithms. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data). The Decision Tree which has a categorical target variable is called a Categorical variable decision tree which we will be using.

The decision of making strategic splits heavily affects a tree's accuracy. The decision criteria are different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

We intend to use a decision tree to classify an app into a rating category. It would consider all the attributes we included to classify it into a 1,2,3 or 4 rated app. We consider the same split of 80-20 as our training and validation sets.

As a traditional process during machine learning, we create a base model and fit it on our training and validation data. Then, we conduct a grid search to tune the hyperparameters to the best possible so that our model will perform optimally. We consider all the predictors in our final data frame and set the parameters of max_depth = 5, criterion = entropy, min sample split = 2, max leaf nodes = 7 which we obtain as the best estimators through our grid search.

## ii. Random Forest

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees. In a more simplified way, random forest builds multiple decision trees and merges them together to get a more accurate and table prediction. We consider the same split of 80-20 as our training and validation sets. We then use the RandomForestClassifier to build our model and fit the training and validation sets. We choose the n_estimators as 50 while setting the parameters.

## iii. KNN Classifier

The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It is easy to implement and understand but has a major drawback of becoming significantly slows as the size of the data in use grows.

KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression).

In the case of classification and regression, we saw that choosing the right K for our data is done by trying several Ks and picking the one that works best.

**Advantages**

a. The algorithm is simple and easy to implement.
b. There is no need to build a model, tune several parameters, or make additional assumptions.

c. The algorithm is versatile. It can be used for classification, regression, and search (as we will see in the next section).

**Disadvantages**

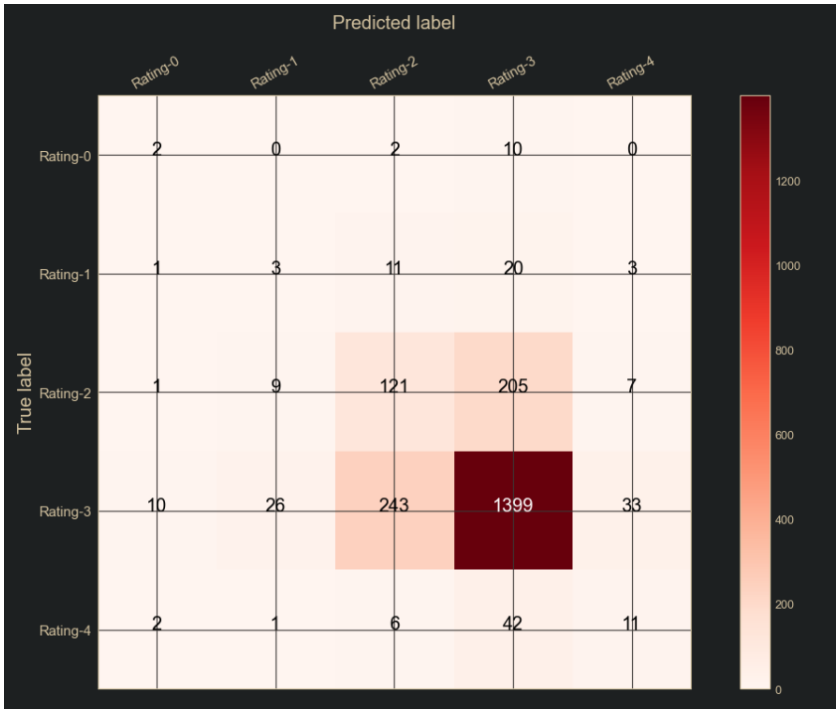
a. The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

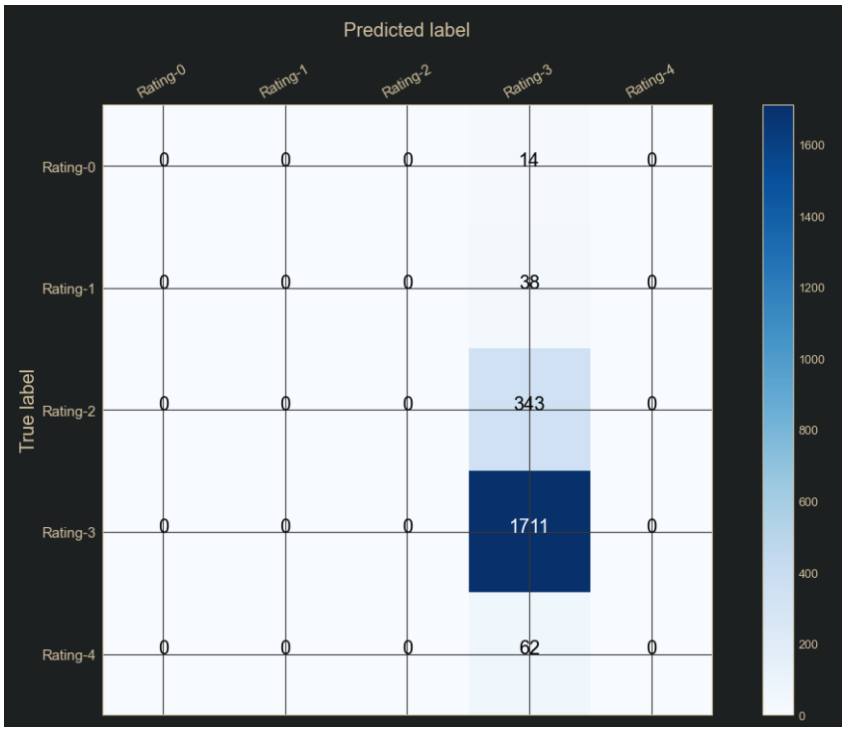
Next, we use the K- nearest neighbors classifier using knn and using the same split of 80-20 of training and testing for training the model and then validating it. We use the knn() function to train a model based on the training set. We initiate the base model and inspect the base parameters, and then create a grid search based on these parameters. We consider all the predictors in our final data frame and set the parameters of n_iter = 100, cv = kfold and n_jobs = 4 which we obtain as the best estimators through our grid search.

# 6. MODEL EVALUATION

## i. Decision Tree

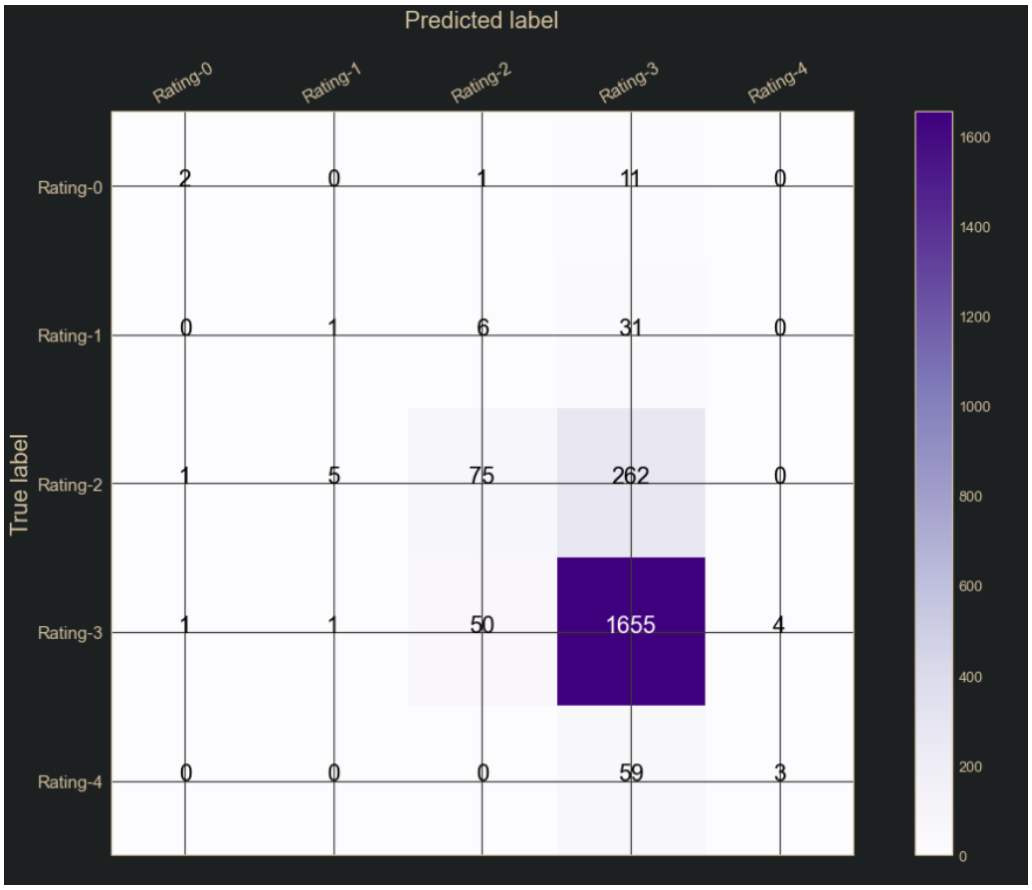
We build a confusion Matrix to understand the TPR and FPR of the model. The final accuracy obtained for our model was 78.92%



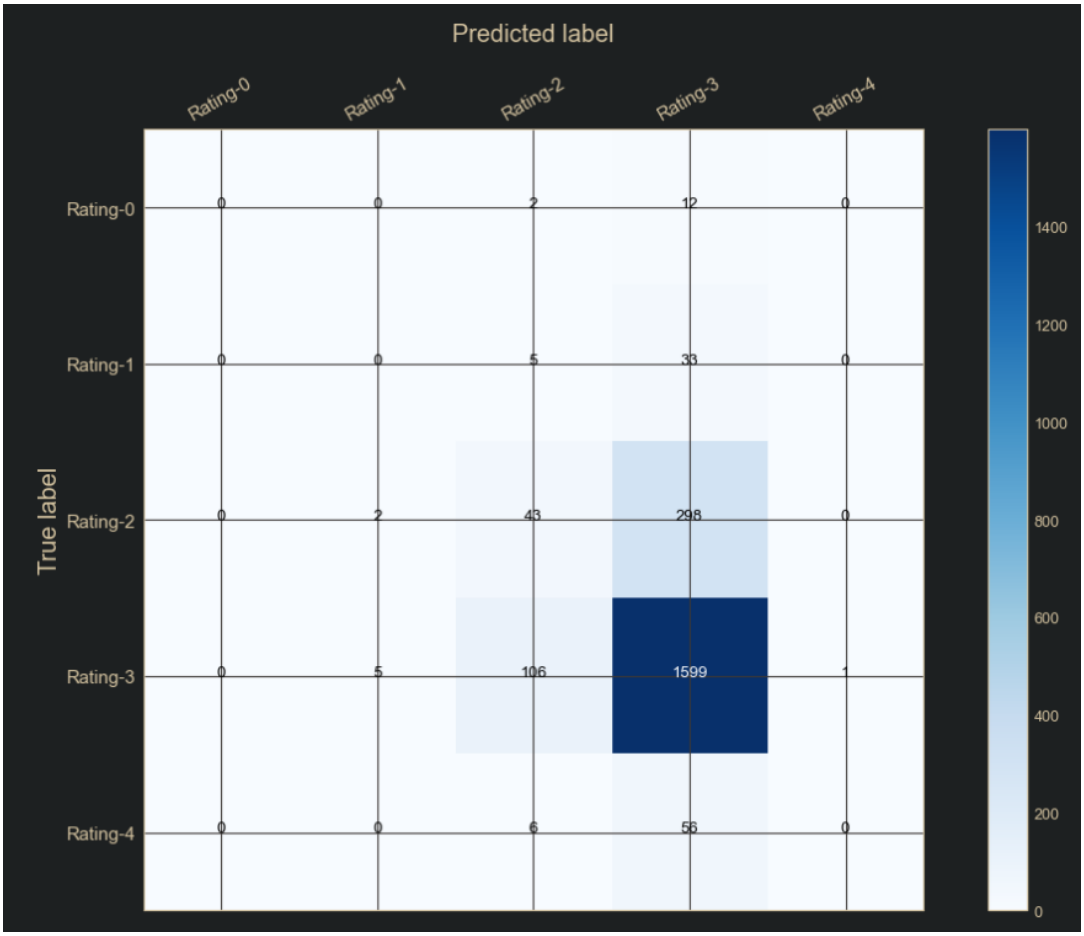
## ii. Random Forest

To evaluate the model performance, we check the results on the validation set that the model predicted as compared to their actual values.



### iii. KNN

To evaluate the model performance, we check the results on the validation set that the model predicted as compared to their actual values.




## Comparing Model Accuracies:

| MODEL | ACCURACY |
|---|---|
| | |
| **Decision Tree** | **78.92** |
| **Random Forest** | **80.07** |
| **KNN** | **75.73** |

# 8. DISCUSSION

## a. DSM recommendations

We consider Random Forest as our best model with an accuracy of 80.07% indicating that our model will be able to correctly predict an app to its expected rating ≈ 80% of the time.

The analysis and model built here focuses on both, the app industry as well as the developers and publishers. Publishing companies and App scouts can use this data to decide whether they should be interested in building an app in a certain category. Developers can use this as a reference to decide app attributes while developing an app.

## b. DSM Limitations

The limitations of the DSM are as follows:

1. Majority of the apps are free. So, our model cannot accurately predict the rating of Paid apps and cannot benefit the premium in-app services which require payments.
2. This model depends on the audio features provided by Google Play store which is a unique platform exclusive to Android users. It is not an industry standard hence the analysis will be limited to data provided by Google Play.
3. It does not consider external factors such as current environment(pandemic), publisher popularity, current trends, advertisements, celebrity endorsements etc. which are very influential to the popularity of an app.

## c. Enhancements/Future Work

1. We would like to have more attributes about the apps so that we can provide better results for the task of predicting rating.
2. A few attributes about external factors influencing rating would be helpful. For example: knowing the advertising rate of amount spent at advertising etc.
3. User data such as, age groups, location etc. would help us make more accurate and specific predictions
4. We would also like to implement Neural Networks and deep learning techniques to understand if we can classify the app ratings better.

## REFERENCES:


1. https://www.kaggle.com/lava18/all-that-you-need-to-know-about-the-android-market
2. https://www.kaggle.com/accountstatus/google-play-store-apps-eda-and-reccomendations