Google Play Store Applications Analysis

Viviana Pavon
Vanessa Miranda
Jaykumar Raichura
Xinshi Li

Table of contents

01

Introduction

Business Problem. Data Description and Preprocessing

Introduction

Business Problem Analysis

We will assist Google in determining which applications work best on their platform in order to increase user engagement. Google Play takes 30% - 40% of the amount people spend for applications or as in-app transactions. Google can incentivise third-party app producers to build more engaging apps and thus increase revenue by identifying which apps perform better on the platform and why.

Our Agenda

We intend to develop a model that identifies the most critical input variables that result in more downloads, higher ratings, and which applications are more likely to receive favorable or negative feedback. We will define user engagement using these measures.

02

Describing Dataset

Data information - input and output variables

Data Description and Preprocessing

Data Set Source
https://github.com/gauthamp10/Google-Playstore-Dataset

Data Collected:
With the help of Python and Scrapy running on Cloud VM.

Data Last Collected:
June 2021

Author:
Gautham Prakash

Dataset Summary

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2312944 entries, 0 to 2312943
Data columns (total 25 columns):
 #   Column             Dtype  
---  ------             -----  
 0   Unnamed: 0         int64  
 1   App Name           object 
 2   App Id             object 
 3   Category           object 
 4   Rating             float64
 5   Rating Count       float64
 6   Installs           object 
 7   Minimum Installs   float64
 8   Maximum Installs   int64  
 9   Free               bool   
 10  Price              float64
 11  Currency           object 
 12  Size               object 
 13  Minimum Android    object 
 14  Developer Id       object 
 15  Developer Website  object 
 16  Developer Email    object 
 17  Released           float64
 18  Last Updated       int64  
 19  Content Rating     object 
 20  Privacy Policy     object 
 21  Ad Supported       bool   
 22  In App Purchases   bool   
 23  Editors Choice     bool   
 24  Scraped Time       object 
dtypes: bool(4), float64(5), int64(3), object(13)
memory usage: 379.4+ MB
```

We can visualize we have numerical and categorical data, 25 columns in total. 2312944 range of values of the row indexes and our data type varies: bool, float, intg and object.

Summary of our dataset is needed for exploratory analysis of the data.

Missing Values Percentage

	column_name	percent_missing
Unnamed: 0	Unnamed: 0	0.000000
App Name	App Name	0.000086
App Id	App Id	0.000000
Category	Category	0.000000
Rating	Rating	0.989345
Rating Count	Rating Count	0.989345
Installs	Installs	0.004626
Minimum Installs	Minimum Installs	0.004626
Maximum Installs	Maximum Installs	0.000000
Free	Free	0.000000
Price	Price	0.000000
Currency	Currency	0.005837
Size	Size	0.008474
Minimum Android	Minimum Android	0.282324
Developer Id	Developer Id	0.001427
Developer Website	Developer Website	32.894657
Developer Email	Developer Email	0.001340
Released	Released	3.071972
Last Updated	Last Updated	0.000000
Content Rating	Content Rating	0.000000
Privacy Policy	Privacy Policy	18.199879
Ad Supported	Ad Supported	0.000000
In App Purchases	In App Purchases	0.000000
Editors Choice	Editors Choice	0.000000
Scraped Time	Scraped Time	0.000000

We can visualize the attributes with the highest missing values on this dataset are: Developer Website, Released and Privacy Policy

Descriptive Statistics:

Means, medians, modes, correlations, variance, outliers

	Unnamed: 0	Rating	Rating Count	Minimum Installs	Maximum Installs	Price	Released	Last Updated
count	2.312944e+06	2.290061e+06	2.290061e+06	2.312837e+06	2.312944e+06	2.312944e+06	2.241891e+06	2.312944e+06
mean	1.156472e+06	2.203152e+00	2.864839e+03	1.834452e+05	3.202017e+05	1.034992e-01	2.018169e+03	2.019458e+03
std	6.676896e+05	2.106223e+00	2.121626e+05	1.513144e+07	2.355495e+07	2.633127e+00	2.146142e+00	1.575108e+00
min	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	2.010000e+03	2.009000e+03
25%	5.782358e+05	0.000000e+00	0.000000e+00	5.000000e+01	8.400000e+01	0.000000e+00	2.017000e+03	2.019000e+03
50%	1.156472e+06	2.900000e+00	6.000000e+00	5.000000e+02	6.950000e+02	0.000000e+00	2.019000e+03	2.020000e+03
75%	1.734707e+06	4.300000e+00	4.200000e+01	5.000000e+03	7.354000e+03	0.000000e+00	2.020000e+03	2.021000e+03
max	2.312943e+06	5.000000e+00	1.385576e+08	1.000000e+10	1.205763e+10	4.000000e+02	2.021000e+03	2.021000e+03

Comparing the mean and the median, we noticed Unnamed, Rating, Rating Count, Minimum Installs, Maximum Installs - The mean is greater than the median - the data is skewed to the right.
Released and Last Updated have almost symmetric data -If data is symmetric, the mean and the median are similar.

Summarized data by running a single command with describe() - returns the count, mean, standard deviation, first quartile, median, third quartile, minimum and maximum values for each numeric column in our dataset.

Correlation Matrix

	Unnamed: 0	Rating	Rating Count	Minimum Installs	Maximum Installs	Free	Price	Released	Last Updated	Ad Supported	In App Purchases	Editors Choice
Unnamed: 0	1.000000	-0.000600	0.000179	0.001141	0.001067	0.000778	0.000405	-0.000453	-0.000265	-0.000792	-0.000637	0.000279
Rating	-0.000600	1.000000	0.013038	0.011214	0.012615	-0.010756	-0.003674	-0.246686	-0.017599	0.160502	0.153820	0.019485
Rating Count	0.000179	0.013038	1.000000	0.545281	0.547571	0.001410	-0.000411	-0.023040	0.009790	0.005294	0.025737	0.137311
Minimum Installs	0.001141	0.011214	0.545281	1.000000	0.954037	0.001623	-0.000461	-0.019973	0.008367	0.002250	0.014179	0.059932
Maximum Installs	0.001067	0.012615	0.547571	0.954037	1.000000	0.001815	-0.000515	-0.022415	0.009337	0.002882	0.016101	0.064206
Free	0.000778	-0.010756	0.001410	0.001623	0.001815	1.000000	-0.278831	0.137050	0.101877	0.123617	0.008669	-0.003157
Price	0.000405	-0.003674	-0.000411	-0.000461	-0.000515	-0.278831	1.000000	-0.031449	-0.017518	-0.034276	-0.002634	0.000503
Released	-0.000453	-0.246686	-0.023040	-0.019973	-0.022415	0.137050	-0.031449	1.000000	0.529928	0.049614	-0.063600	-0.020052
Last Updated	-0.000265	-0.017599	0.009790	0.008367	0.009337	0.101877	-0.017518	0.529928	1.000000	0.014380	0.062256	0.016147
Ad Supported	-0.000792	0.160502	0.005294	0.002250	0.002882	0.123617	-0.034276	0.049614	0.014380	1.000000	0.138312	0.005702
In App Purchases	-0.000637	0.153820	0.025737	0.014179	0.016101	0.008669	-0.002634	-0.063600	0.062256	0.138312	1.000000	0.046078
Editors Choice	0.000279	0.019485	0.137311	0.059932	0.064206	-0.003157	0.000503	-0.020052	0.016147	0.005702	0.046078	1.000000

The output above shows that Maximum Installs & Minimum Installs and Released & Last Updated are all positively associated.

Maximum Installs & Minimum Installs R^2 = 30%
Released & Last Updated R^2 = 28%

We can visualize the correlation between each of the variables and themselves are all equal to one and the off diagonal elements give the correlation between each of the pair of the variables.

03

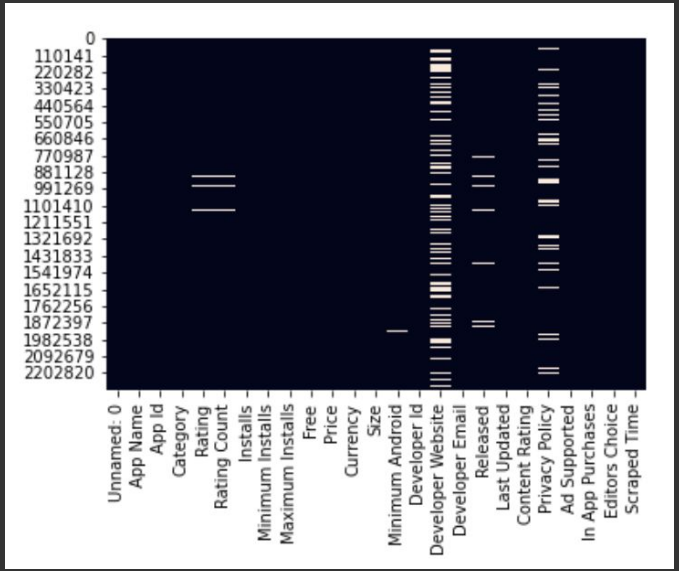
Data Preprocessing

Data cleaning, transforming, dropping columns and rows, inputting data

Data Prepossessing

We can visualize the variables with most missing values are Developer Website, Privacy Policy, followed by Released, Minimum Android, Size, Currency , Installs and Minimum Installs.

	data_type	null_count	unique_count
Unnamed: 0	int64	0	2312944
App Name	object	2	2177830
App Id	object	0	2312944
Category	object	0	48
Rating	float64	22883	42
Rating Count	float64	22883	38482
Installs	object	107	22
Minimum Installs	float64	107	22
Maximum Installs	int64	0	251563
Free	bool	0	2
Price	float64	0	1063
Currency	object	135	15
Size	object	196	1657
Minimum Android	object	6530	154
Developer Id	object	33	758340
Developer Website	object	760835	810440
Developer Email	object	31	950456
Released	float64	71053	12
Last Updated	int64	0	13
Content Rating	object	0	6
Privacy Policy	object	420953	977743
Ad Supported	bool	0	2
In App Purchases	bool	0	2
Editors Choice	bool	0	2
Scraped Time	object	0	1132



- Data Clean - Web observed our dataset containing different size: Megabytes, Kilobyte, Gigabyte and varies with device. We transformed our dataset to megabytes.
- Fixed Install columns - Our dataset contained '+'signs and proceeded to remove.
- Removed duplicate values from APP ID column
- Dropped numerical and categorical variables with zero variance

```
Unnamed: 0            6.676891e+05
Rating                2.106223e+00
Rating Count          2.121626e+05
Installs              1.513144e+07
Minimum Installs      1.513144e+07
Maximum Installs      2.355550e+07
Free                  1.380647e-01
Price                 2.633187e+00
Size                  2.404647e+01
Released              2.146144e+00
Last Updated          1.575112e+00
Ad Supported          4.999940e-01
In App Purchases      2.780544e-01
Editors Choice        1.920092e-02
dtype: float64
```

```
In [40]:
zero_cardinality = []

for i in categorical_var: # for each categorical variables
    if len(df[i].value_counts().index) == 1: # check how many levels it has and if it is one
        zero_cardinality.append(i) # the variable has zero variance as the cardinality is one
        # append it to the list of categorical variables with zero variation

df = df.drop(zero_cardinality, axis = 1)

In [41]:
zero_cardinality

Out[41]:
[]
```

- Transformed Minimum Installs column into integers.
- Dropped categorical variables with high cardinality

```
#High_cardinality - Variables with more than 50000 values
high_cardinality = []
for i in categorical_var: # for each categorical variables
    if len(df[i].value_counts().index) > 50000: # check how many levels it has and if it is more
        high_cardinality.append(i) # than 50000, variable has many levels
        # so append it to the list of categorical variables with high cardinality

print(high_cardinality)
```

```
['App Id', 'Developer Id', 'Developer Email', 'App Name', 'Developer Website', 'Privacy Policy']
```

- Proceeded to fill missing numerical and categorical values
- Verified missing values in our dataset by running the following query.

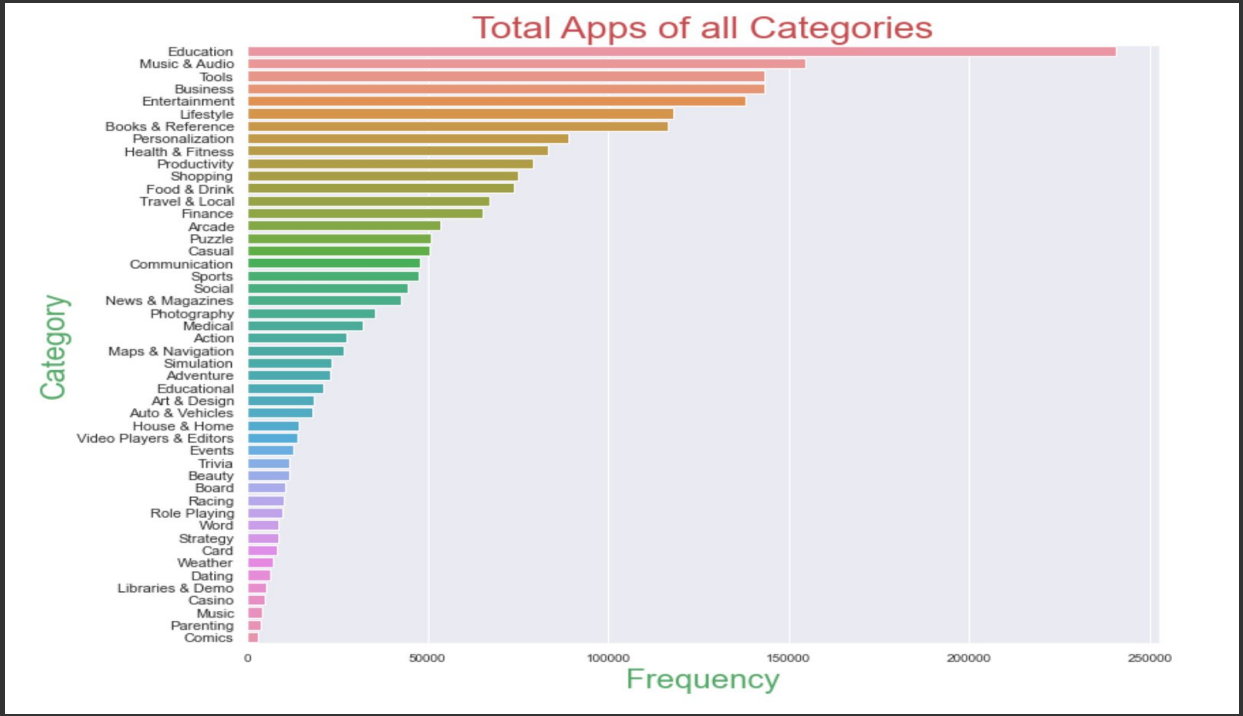
```
In [56]: # Checks the number of missing values by column
         [sum(df[i].isnull()) for i in df.columns]

Out[56]: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

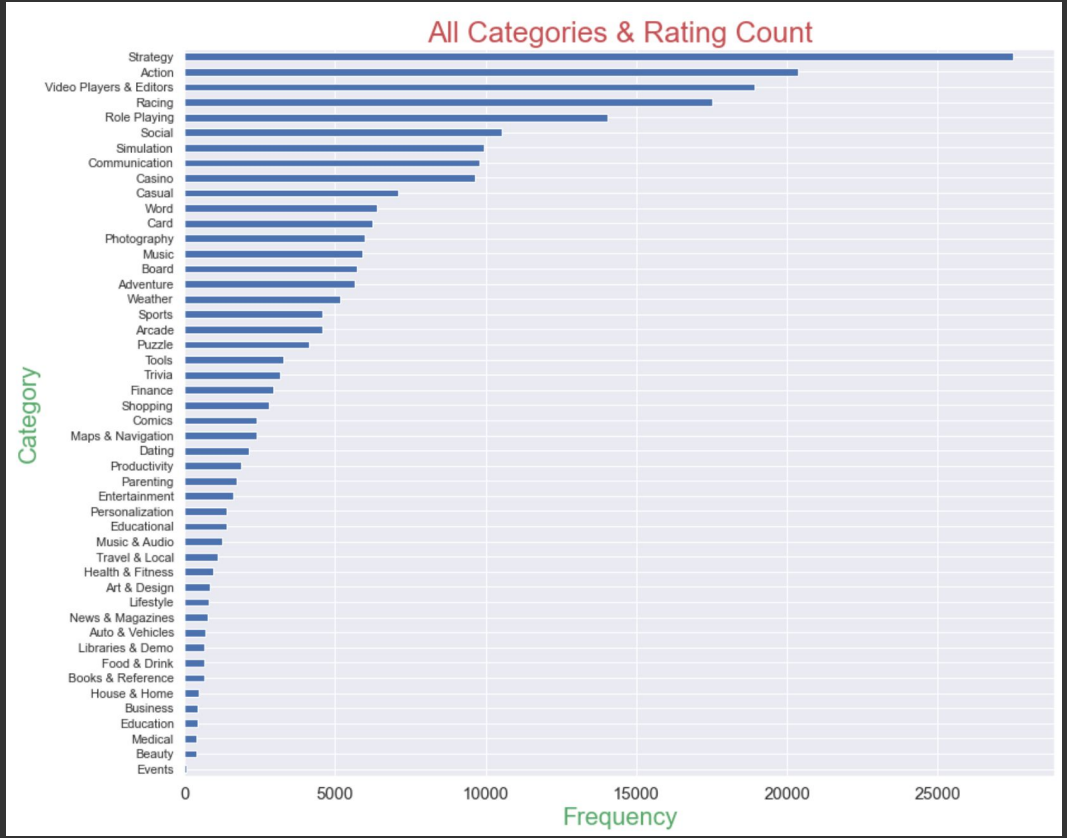
04

Data visualization

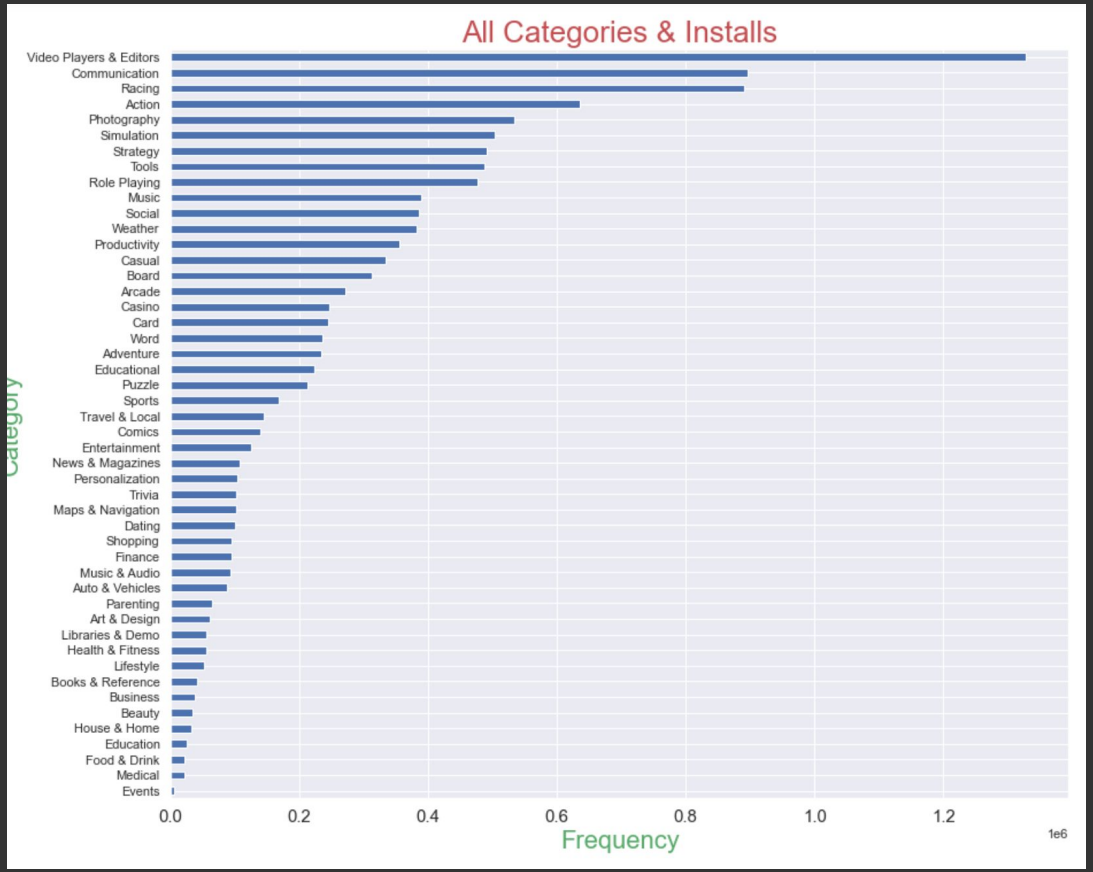
Descriptive Analysis using different graph types



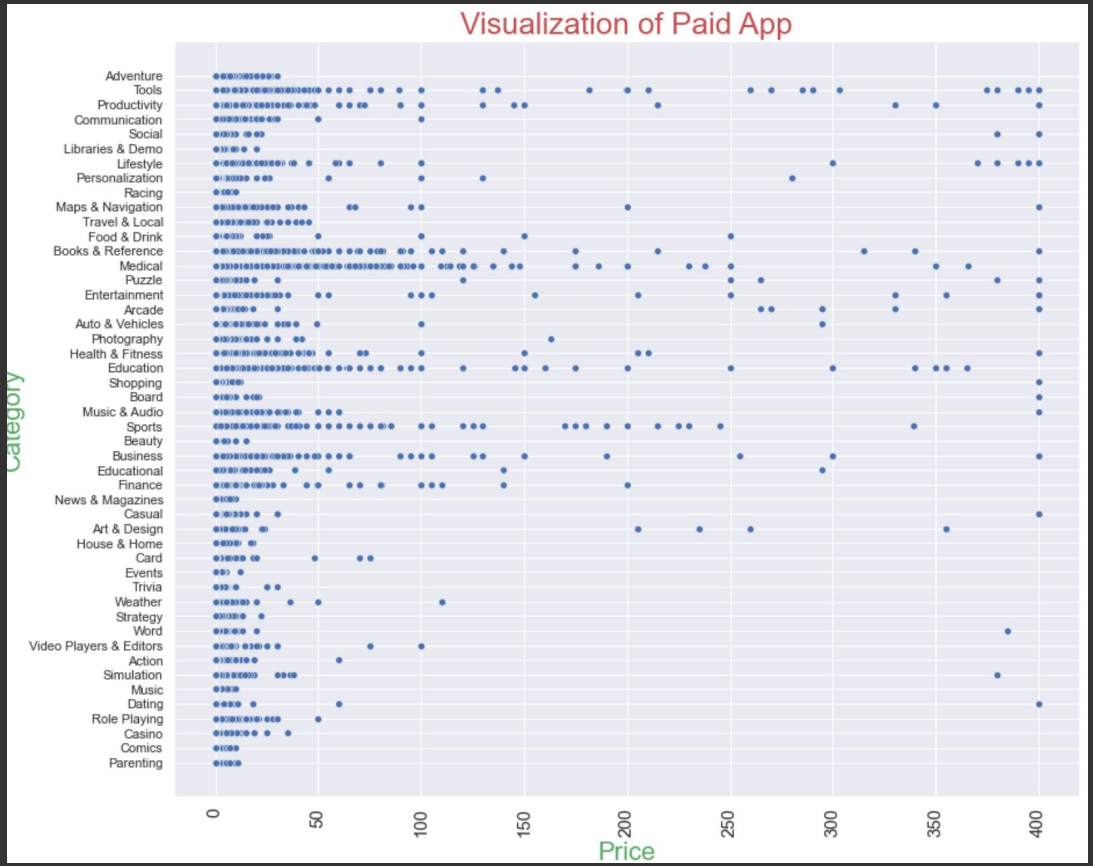
Apps categories with highest frequency are: Education, Music & Audio and Tools



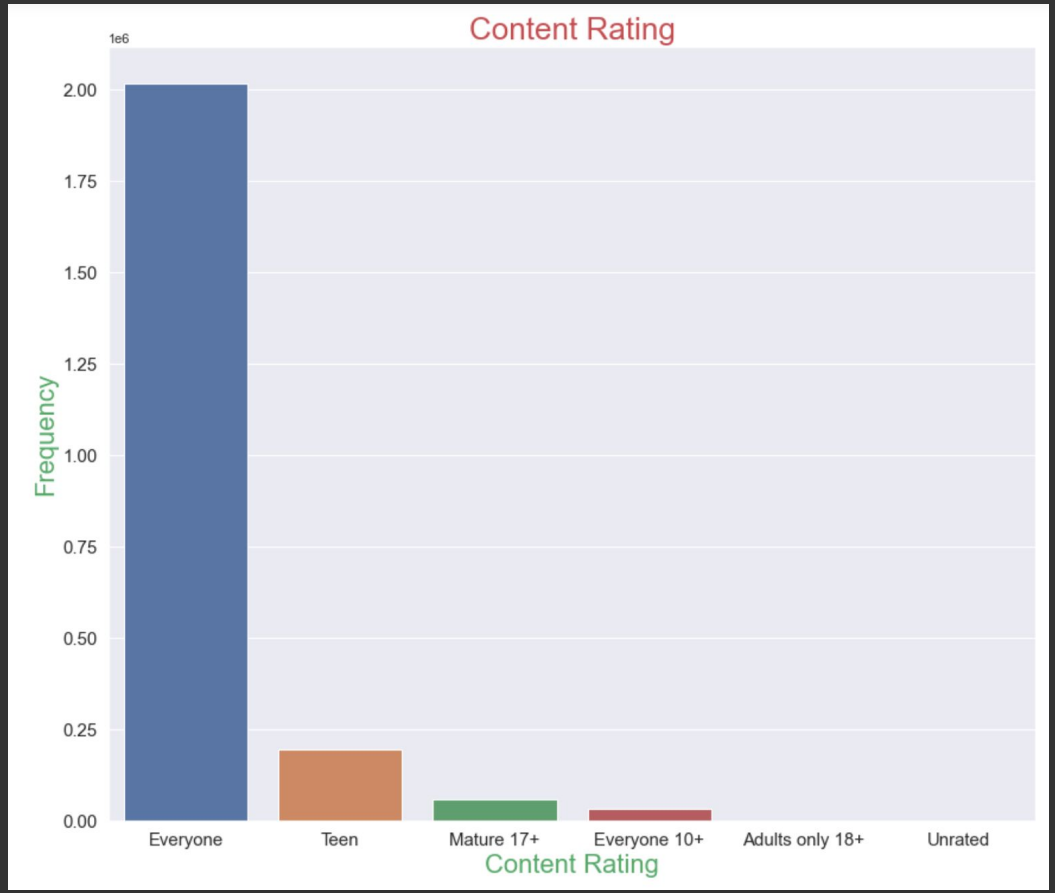
The top three apps with the highest ratings are: Action, Strategy, Video Players and Editors



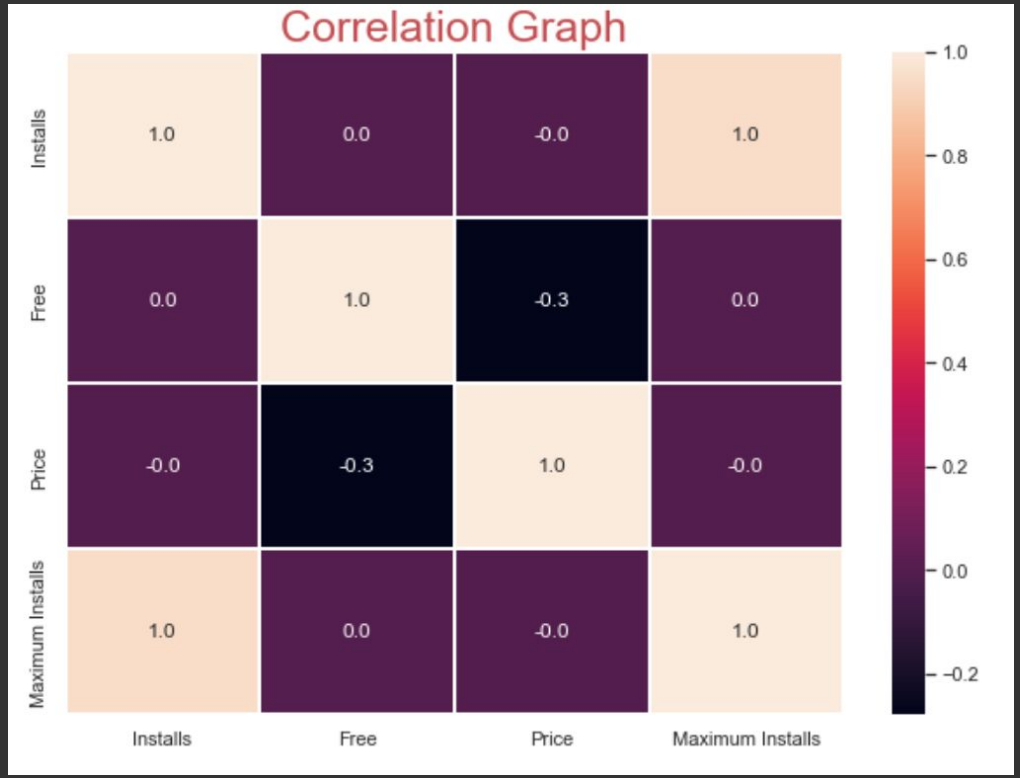
The top three categories with the highest installs are: Video Players and Editors, Communication and Racing



We can visualize most of the apps are free or low price
Also, the most expensive apps are related to Tools and Business.



The highest Content Rating within each App Name are:
Everyone, Teen and Mature 17+



Correlation Matrix - summarize a large amount of data and the goal is to visualize patterns.
Observable pattern -Installs and Maximum Installs correlate or have a good relation with each other

Pie Chart

98.1 %

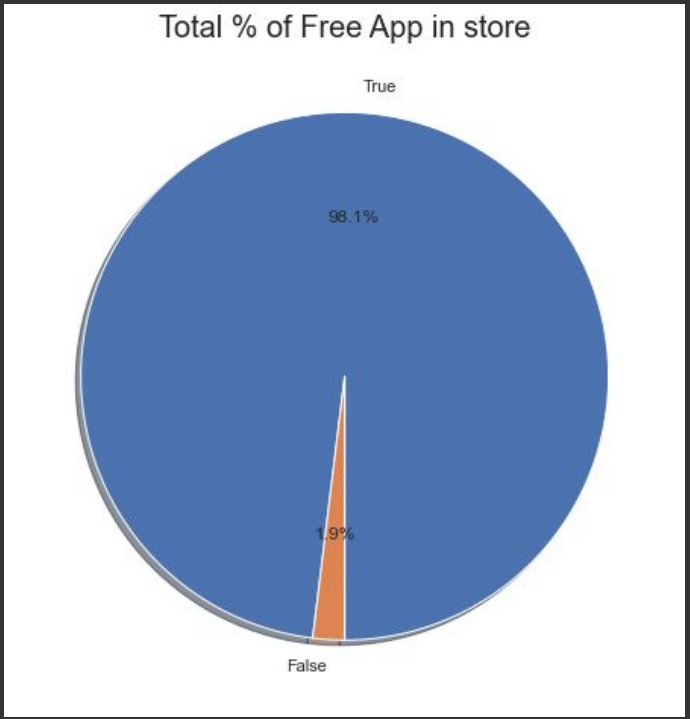
Free

98.1% Apps are free in store

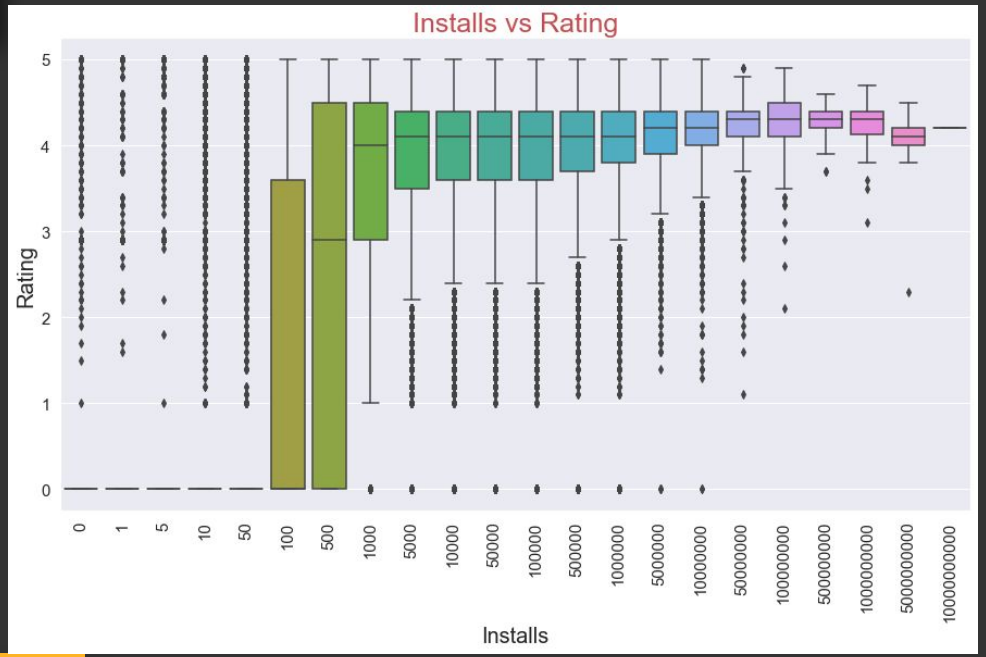
1.9%

Paid

Only 1.9% App needs to pay



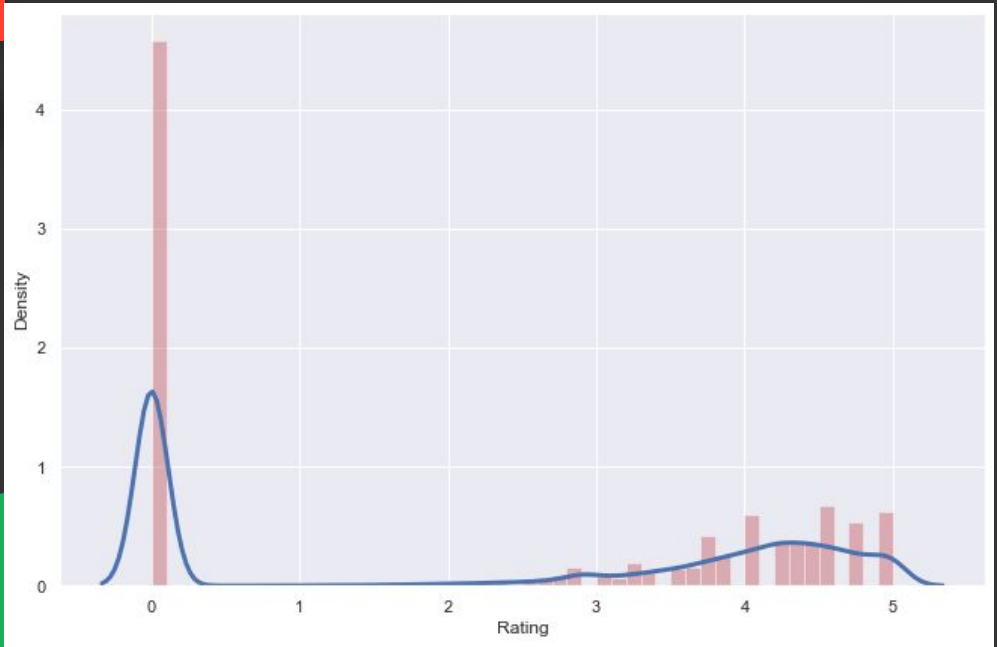
Installs vs. Rating



We can find from the boxplot that avg rating of most app is between 4-5, close to 4, which is very good.

High install apps always got high rates, which means the apps are in good quality.

Distribution of Rating



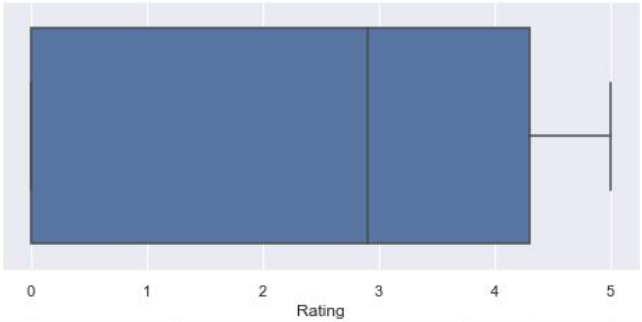
Rating: 0

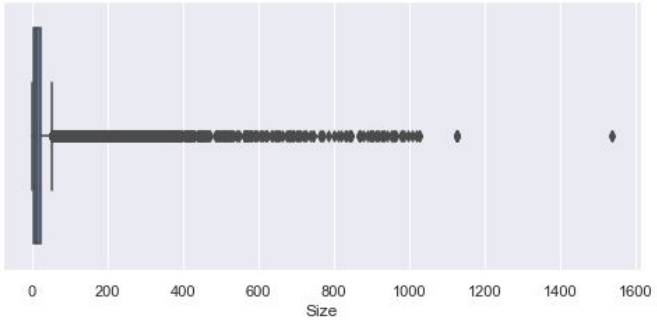
App with 0 rate are in high density means there may be malicious scoring.

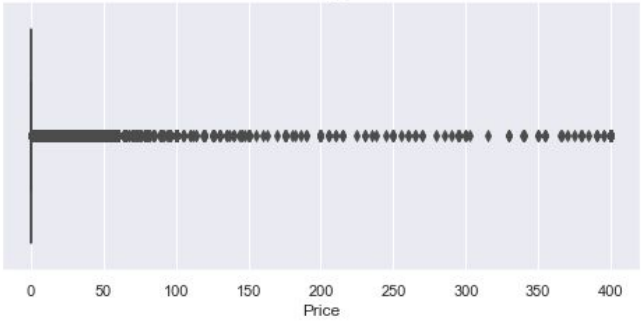
Rating:3-5

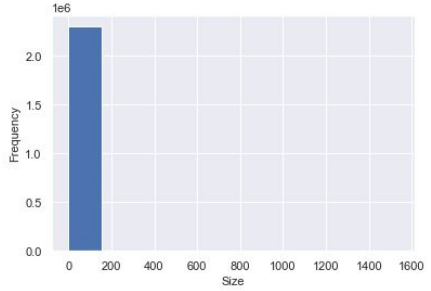
Most of the app rating around 3-5.
More apps are rated in 4 and 5.
Fewer app rated in 3

Boxplot









We can visualize that Size and Price have many outliers. Rating of the app is all in 0-5. Size of the app is between 0-200

Market size

Medical

With the highest avg cost. Expensive

0.7

Role Playing

Second expensive app to use

0.3

Sports

Third highest avg. cost

0.22

05

Data Modeling

Predictive Data Models

Linear Regression

We took a sample of 5000

	Rating	Rating Count	Installs	Minimum Installs	Maximum Installs	Free	Price	Size	Released	Last Updated	Ad Supported	In App Purchases	Editors Choice
Rating	1.000000	0.033276	0.042816	0.042816	0.053526	-0.005463	-0.003815	0.043620	-0.218391	0.001133	0.187189	0.148028	0.028121
Rating Count	0.033276	1.000000	0.981027	0.981027	0.965062	0.003613	-0.002131	0.016971	-0.065143	0.024918	0.023715	0.073844	0.099198
Installs	0.042816	0.981027	1.000000	1.000000	0.985788	0.006486	-0.003579	0.019508	-0.076640	0.029793	0.033809	0.088912	0.080152
Minimum Installs	0.042816	0.981027	1.000000	1.000000	0.985788	0.006486	-0.003579	0.019508	-0.076640	0.029793	0.033809	0.088912	0.080152
Maximum Installs	0.053526	0.965062	0.985788	0.985788	1.000000	0.007957	-0.004433	0.027814	-0.082249	0.034851	0.040928	0.103653	0.125441
Free	-0.005463	0.003613	0.006486	0.006486	0.007957	1.000000	-0.540051	-0.012552	0.135230	0.113847	0.124352	-0.000274	0.004184
Price	-0.003815	-0.002131	-0.003579	-0.003579	-0.004433	-0.540051	1.000000	0.008533	-0.092524	-0.095402	-0.072244	0.004254	-0.002260
Size	0.043620	0.016971	0.019508	0.019508	0.027814	-0.012552	0.008533	1.000000	0.121120	0.208603	-0.065443	0.183279	0.042573
Released	-0.218391	-0.065143	-0.076640	-0.076640	-0.082249	0.135230	-0.092524	0.121120	1.000000	0.513638	0.034420	-0.040730	-0.025655
Last Updated	0.001133	0.024918	0.029793	0.029793	0.034851	0.113847	-0.095402	0.208603	0.513638	1.000000	-0.016103	0.072850	0.024031
Ad Supported	0.187189	0.023715	0.033809	0.033809	0.040928	0.124352	-0.072244	-0.065443	0.034420	-0.016103	1.000000	0.124298	0.014420
In App Purchases	0.148028	0.073844	0.088912	0.088912	0.103653	-0.000274	0.004254	0.183279	-0.040730	0.072850	0.124298	1.000000	0.093682
Editors Choice	0.028121	0.099198	0.080152	0.080152	0.125441	0.004184	-0.002260	0.042573	-0.025655	0.024031	0.014420	0.093682	1.000000

Regression Model

Target Variable: Maximum Installs

OLS Regression Results

Dep. Variable:	Maximum Installs	R-squared:	0.972
Model:	OLS	Adj. R-squared:	0.972
Method:	Least Squares	F-statistic:	5.782e+04
Date:	Sat, 25 Jun 2022	Prob (F-statistic):	0.00
Time:	17:05:27	Log-Likelihood:	-70783.
No. Observations:	5000	AIC:	1.416e+05
Df Residuals:	4996	BIC:	1.416e+05
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	9.687e+04	7029.758	13.781	0.000	8.31e+04	1.11e+05
Minimum Installs	2.107e+06	2.49e+04	84.703	0.000	2.06e+06	2.16e+06
Rating	1.059e+04	2306.126	4.592	0.000	6068.101	1.51e+04
Rating Count	-1.043e+05	2.49e+04	-4.196	0.000	-1.53e+05	-5.56e+04

Omnibus:	12892.307	Durbin-Watson:	2.007
Prob(Omnibus):	0.000	Jarque-Bera (JB):	313302538.267
Skew:	29.023	Prob(JB):	0.00
Kurtosis:	1227.942	Cond. No.	22.8

We can observe that maximum installs has a high correlation with rating count, installs and minimum installs

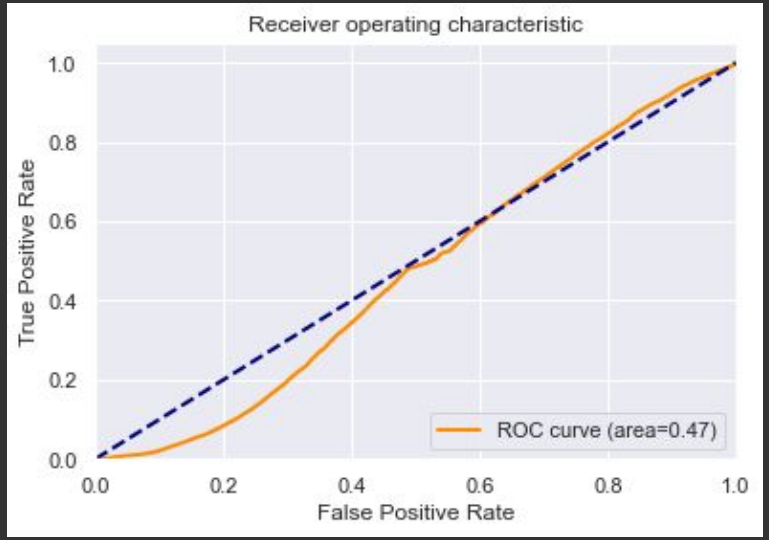
Logistic Regression

Free.True	Minimum Installs	Maximum Installs	Rating Count	Installs	Price	Size
ID						
0	1	1.000000e-09	1.244026e-09	0.000000e+00	10	0.0	0.006508
1	1	5.000000e-07	6.354484e-07	4.619019e-07	5000	0.0	0.001886
2	1	5.000000e-09	4.810233e-09	0.000000e+00	50	0.0	0.002407
3	1	1.000000e-09	1.575766e-09	3.608608e-08	10	0.0	0.001170
4	1	1.000000e-08	3.964296e-08	0.000000e+00	100		

Logistic Regression

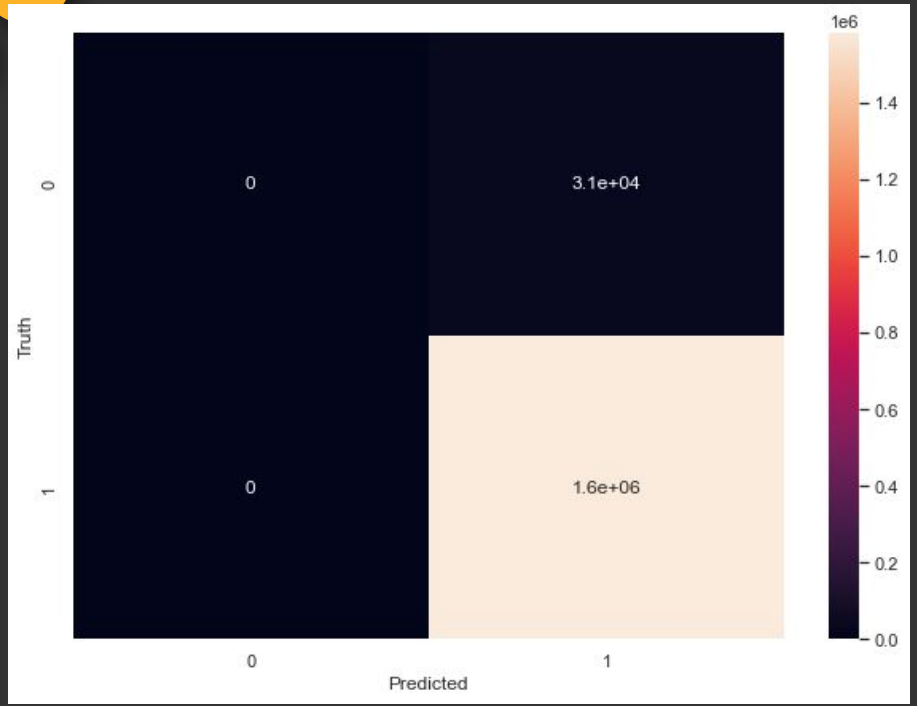
	accuracy	sensitivity	specificity
Logistic Regression	0.980627	0.0	1.0

Accuracy Table



ROC Curve

Logistic Regression



We created this Logistic regression model to predict whether the App is Free or Paid.

We can notice that the Logistic regression accuracy is 0.98. So we can use this Logistic regression estimates the probability of an Free app occurring.

06

Conclusion

Summarizing Data Results & Recommendations

Conclusion

Through the data preprocessing, visualization and modeling we developed a Linear regression model that identifies the most critical input variables that result in more downloads, higher ratings, and which applications are more likely to receive favorable or negative feedback. We can combine those analysis result to provide a better PlayStore which contains more higher ratings app with high quality.

We can also noticed that maximum installs is high correlation with rating count, installs and minimum installs. Those data modeling can help Google PlayStore better integrate platform, launch more high-quality APP.

Thanks!

Do you have any questions?