Exploratory Data Analysis on Google play store application

INTRODUCTION

Google Play store application is a digital distribution service operated and developed by Google. It also serves as a digital media store, offering music, books, movies, and television programs. Google Play store applications are either chargeable or free of cost. Anyone can download directly from their Android device or through the Google Play website. The Google Play store had over 82 billion app downloads in 2016 and has reached over 3.5 million apps published in 2017.

Google Play was launched on March 6, 2012, bringing together Android Market, Google Music, and the Google eBook store under one product. The services included in Google Play are Google Play Books, Google Play Games, and Google Play Music.

To perform this analysis we have downloaded data set from one of the popular website Kaggle (https://www.kaggle.com/lava18/google-play-store-apps). This dataset contains more than 10K rows and 13 columns information like App, Category, Rating, Reviews, Size, Installs, Type, Price, Content Rating, Genres, Last Updated, Current Ver. and Android Ver. Above dataset consist of 33 category of app's like art and design, beauty, books and reference, etc. As usual app rating ranges between1 to 5. Based on initial stage of analysis it becomes clear that some of these factors definitely have a role to play in an app's rating and performance.

Software used: Python and SPSS

QUESTIONS AND FINDINGS

In Google play store, which category of application got highest and lowest rating?

In Google play store, which category of application priced high and free?

In Google play store, which category of application received most and least review?
Which parameters affect the ratings the most?

Predict the User Rating?

Table 1: About data

Variable	Detailed description
App	The name of the application
Category	The category to which the app belongs
Rating	User rating of the app when scraping was done
Reviews	Number of user reviews for the app
Size	Size of the app
Installs	Number of user downloads/installs for the app
Type	A binary variable which dentoes whether a
Price	The cost of the app in US Dollars
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). For

Variable	Detailed description
	eg, a musical family game will belong to Music, Game, Family genres.
Last Updated	Date when the app was last updated on Play Store
Current Version	Current version of the app available on Play Store
Android Version	Min required Android version

Data Preparation & Exploratory Analysis

The first step in the data analytics is to understand the data and business problem. Then try to get a summary of data using describe function, which will give us a basic picture of the data set. Later we need to detect and then delete the duplicate values and outlier's in the given dataset. Otherwise it is difficult to build a sophisticated model. Here we have used drop duplicates function to remove duplicate values and we used box plot to detect outlier in the dataset. Afterword's, we have used replace function to replace the values that make sense to the dataset.

Later, we have created scatter plot categories Vs rating. It shows that event application got high rating with mean 4.40 out of 5, then education and art and design apps received similar rating, which stands in the second place. Books and reference apps stands in the third place. However, tools, video players, maps and navigation and dating apps received least rating. Concerned companies need's to take care of these least rated apps to make it more customers friendly and the results are graphically shown in Figure 1.

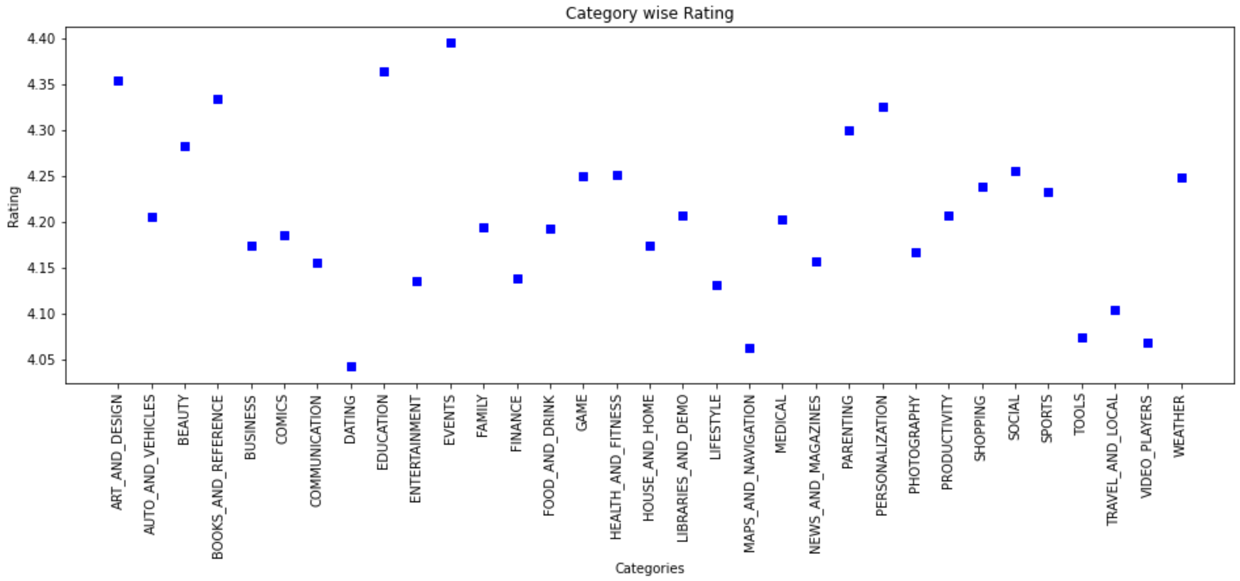

Figure 1

Similarly, in order to study about category wise pricing we have created a line graph as shown in Figure 2. It indicates that customers need to pay more money to finance related apps and also it indicates that many customers interested in paid apps then free apps, it is may be because of customer believed that paid apps are more secure then the free apps for doing financial transaction. Moreover, we can see that family, lifestyle and medical apps in the second, third and fourth position in terms of price. Remaining apps are free and some apps are priced to a little amount.

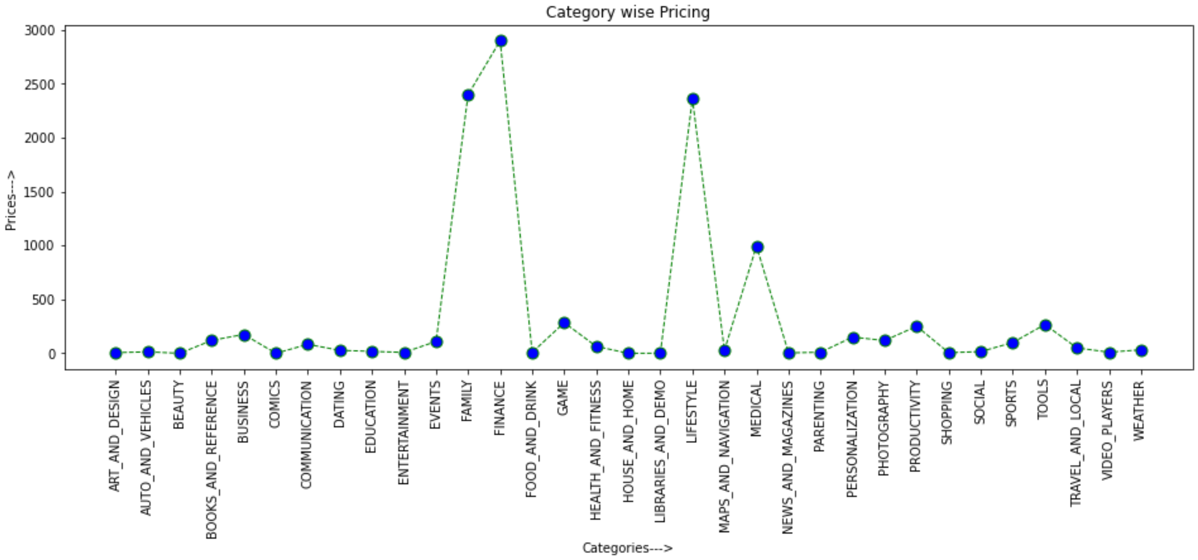

Figure 2

In Figure 3, we can see that family related apps received more reviews, we can also see the same picture in figure 1, it received highest rating but which is not true always. Then, gaming apps received second highest reviews and tools application stands in the third position. Business, communication, lifestyle, medical, personalization, photography and productivity apps received almost similar reviews but art and design, auto and vehicles, beauty apps, etc..., failed to get enough reviews.

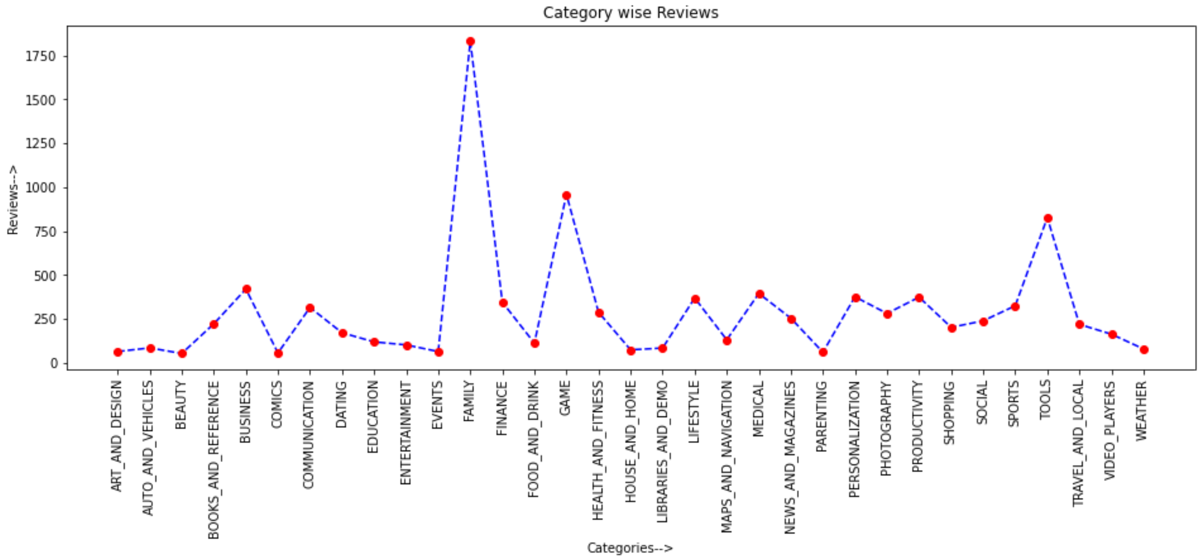

Figure 3

Correlation Analysis

A correlation matrix was then created to find the relationship between the numerical variables namely Rating, Reviews, Size, Installs, and Price. This correlation plot can be seen in Figure 4.

Table 2. Correlation matrix

	Rating	Reviews	Size	Installs	Price
Rating	1				
Reviews	0.050	1			
Size	0.046	0.179	1		
Installs	0.034	0.625	0.134	1	
Price	-0.019	-0.008	-0.022	-0.009	1

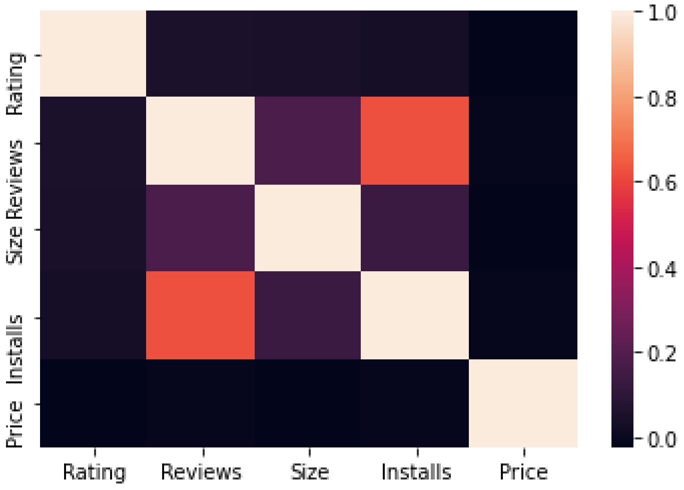

Figure 4

An interesting point that could be inferred from the above analysis was that the variables Installs and Reviews are fairly correlated which makes sense as because the usage of the app increases the number of reviews also increases.

Hypothesis testing

H_0: *Google play store application reviews, size, installs and price will not significantly impact on rating performance.*

H_1: *Google play store application reviews, size, installs and price will significantly impact on rating performance.*

The impact of Google play store application reviews, size, installs and price on their rating performance was studied with the help of linear regression analysis. The descriptive is provided in Table 3 and a summary of the regression analysis is provided in Table 4. Before running the regression model we have applied the Z score transformation technique. This technique helps to bring data from different sources onto the same scale.

Table 3. Descriptive Statistics

	Mean	Std. Deviation	N
Rating	-0.017	1.034	8432
Reviews	-0.052	0.511	8432
Size	0.000	1.000	8432
Installs	-0.075	0.462	8432
Price	0.008	1.070	8432

Table 4. Model Summary

				Change Statistics				
R	R Square	Adjusted R Square	Std. Error of the Estimate	R Square Change	F Change	df1	df2	Sig. F Change
.073	**0.005**	**0.005**	1.03108529	0.005	11.241	4	8427	0.000

Table 5. Model Coefficients

Model	Unstandardized Coefficients		Standardized Coefficients	t	Sig.
	B	Std. Error	Beta		
(Constant)	-0.011	0.011		-0.959	0.337
Reviews	0.106	0.028	0.052	3.825	0.000
Size	0.036	0.011	0.035	3.184	0.001
Installs	0.010	0.030	0.004	0.323	0.746
Price	-0.017	0.010	-0.017	-1.604	0.109

The results of the regression analysis show that Google play store application reviews, size, install and price will significantly impact on rating performance ($p<0.05$). However, it can be observed that the size of the effect is very small. Google play store reviews, size, installs and price could explain only 0.56% variance in the rating ($R^2=0.056$). Though the impact is very small, it has been found to be significant; thereby, accepting the alternative hypothesis "*Google play store application reviews, size, installs and price will significantly impact on rating performance*".

From the coefficients table (Table 5), it can seen that a unit variance in app reviews produces about B=0.106 variations in rating performance. Similarly, a unit variance in app size produces about B=0.0336 variations in rating performance. The regression model is thus given as follows:

Rating = (-0.011) + (0.106* Reviews) + (0.036* Size) + (0.010* Installs) + (-0.017* Price)

Note: Building predictive model using the above model, that does not make sense ($R^2=0.056$).

Python Coding available at (https://github.com/MaheshKulal/Exploratory-Data-Analysis-on-Google-Play-Store-app/blob/master/EDA%20on%20Google%20Play%20Store%20applications.ipynb)

Regression analysis and Z score transformation is done using IBM SPSS software.

Python Coding:

Importing Python Libraries

```
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline
```

```
# Reading the Dataset
data=pd.read_csv("googleplaystore.csv")
## print the top10 records
data.head(10)
## printing shape of dataset with rows and columns
print(data.shape)
data.info()
##Summarizing data sets with statistics
data.describe(include='all')
##pictorial representation of data
data.hist('Rating')
#Displaying the total number off Apps:
tn = len(data)
print ('The number of records in the data set is: ', tn)
#Removing duplicate Apps and filling Null values from the dataset:
data.drop_duplicates(subset='App', inplace=True)
dpv = len(data)
print ('The number of unique App records in the data set is: ', dpv)
##Outlier detection using boxplot
data.boxplot('Rating')
##Summing up of all missing values in the dataset
```

```
data.isnull().sum()

##Outlier detection in the Rating column

data[data.Rating>5]

##Deleting the perticular row that contain outlier

data.drop([10472],inplace=True)

##Visulalising the dataset

data.boxplot("Rating")

##Ploting Histogram for Rating Column

data.hist("Rating")

##Replacing ? with NaN (Not a Number)

data.replace('?','NaN')

data.isnull().sum()

data.info()

print(data["Type"].mode())

print(data['Current Ver'].mode())

print(data['Android Ver'].mode())

##Replacing the missing value with median

data['Rating'].fillna(data['Rating'].median(), inplace=True)

##Checking the missing values

data.isnull().sum()

##Replacing the $ sign that contained in string
```

```
data["Price"]=data["Price"].apply(lambda x:str(x).replace("$","")if '$'in str(x)else str(x))

data["Size"]=data["Size"].apply(lambda x:str(x).replace("M","")if '$'in str(x)else str(x))

data['Size'] = data['Size'].apply(lambda a: str(a).replace('M', ") if 'M' in str(a) else a)

data['Size'] = data['Size'].apply(lambda a: str(a).replace(',', ") if ',' in str(a) else a)

data['Size'] = data['Size'].apply(lambda a: str(a).replace('+', ") if '+' in str(a) else a)

data['Size'] = data['Size'].apply(lambda a: str(a).replace('Varies with device', 'NaN') if 'Varies with device' in str(a) else a)

##Replacing the missing value with median

data['Size'].fillna(data['Size'].median(), inplace=True)

##Converting string data to float

data["Price"]=data['Price'].apply(lambda x :float(x))

data['Size'] = data['Size'].apply(lambda x : float(x))

##Converting object type data to integer

data["Reviews"]=pd.to_numeric(data["Reviews"],errors="coerce")

##Same as the above

data["Installs"]=data["Installs"].apply(lambda x:str(x).replace("+","")if '+'in str(x)else str(x))

data["Installs"]=data["Installs"].apply(lambda x:str(x).replace(",","")if ','in str(x)else str(x))

data["Installs"]=data['Installs'].apply(lambda x :float(x))

data.info()

data.describe()

##Grouping the different data based on categories

gr_ct=data.groupby('Category')
```

```
x = gr_ct['Rating'].mean()

y = gr_ct['Price'].sum()

z=gr_ct['Reviews'].count()

print(x)

print(y)

print(z)

##Plot shows the comparision of different categories w.r.t rating

plt.figure(figsize=(16,5))

plt.plot(x,'bs', color='b')

plt.xticks(rotation=90)

plt.title('Category wise Rating')

plt.xlabel('Categories')

plt.ylabel('Rating')

plt.show()

##Plot shows the comparision of different categories w.r.t pricing

plt.figure(figsize=(16,5))

plt.plot(y,color='green', linestyle='dashed', linewidth = 1,

        marker='o', markerfacecolor='blue', markersize=9)

plt.xticks(rotation=90)

plt.title('Category wise Pricing')

plt.xlabel('Categories--->')
plt.ylabel('Prices--->')

plt.show()

##Plot shows the comparision of different categories w.r.t reviews

plt.figure(figsize=(16,5))

plt.plot(z,'r--', color='b')

plt.plot(z,'ro', color='r')

plt.xticks(rotation=90)

plt.title('Category wise Reviews')

plt.xlabel('Categories-->')

plt.ylabel('Reviews-->')

plt.show()

#Correlation Heatmap for numerical variables

sns.heatmap(data.corr())

data.corr()
```