Final Project

**Google Play Store Apps**

ALY6015- Intermediate Analytics [CRN: 80797] Northeastern University

Submitted to Prof. Valeriy Shevchenko

Date of Submission – 17 May 2019

**Group Epsilon Members**

Maryam Heidari

Thi Thai Ngan Le

Raviteja Vakkanthula

Findlee Odili-Obi

# Abstract

In this paper, we want to find which application is successful in Google play store, in order to guide investors in making better choices of their money when investing in applications. For that, we focused on the application variables, like average of rating, number of reviews, number of Installs, price, Type and category. We placed the dataset in clusters based on the variables in order to find which application counts as a successful application, so that investors can compare the new application and find if it is going to be successful or not. Moreover, we came up with a model which predicts the number of Installs base on those variables. Our analysis will provide solid understanding that can be implemented on both newly designed applications and existing applications.

**Keywords**

Google Play Store

Successful Application

Clustering

Linear Regression

Table of Contents

[Abstract 2](#_Toc8930105)

[Introduction 4](#_Toc8930106)

[Background and History 4](#_Toc8930107)

[Dataset 4](#_Toc8930108)

[Method used 5](#_Toc8930109)

[Research questions 6](#_Toc8930110)

[Analysis 6](#_Toc8930111)

[Data Cleaning 6](#_Toc8930112)

[Exploratory Data Analysis 10](#_Toc8930113)

[Data Mining Clustering Method 14](#_Toc8930114)

[Linear Regression 19](#_Toc8930115)

[Conclusion 25](#_Toc8930116)

[References 27](#_Toc8930117)

[Appendix 28](#_Toc8930118)

[R Codes: 28](#_Toc8930119)

# Introduction

## Background and History

With the robust development of technological devices, the app market achieves a dramatic growth. According to App Developer Magazine, “there are over 300 app stores worldwide today and the number is still growing” (Alex Makarevich, 2018). In the App market, Google Play and Apple App Store are giants followed by Window Store, Amazon Appstore and BlackBerry World. The statistic from Statista of the first quarter in 2019 indicates that Google Play store takes the lead in the number of apps available with 2.1 million (Statista, 2019). It is also stated by Google that “2 billion monthly active devices on Android” (Ben Popper, 2017). In general, Google App Store is considered as a giant in the dynamic app market which attracts many app developers. Hence, it is necessary to indicate the most trending app in Google Play Store.

There are many elements influencing the success of an application such as overall rating, number of installs, and so forth. The goal of this project is to create a model illustrating the most common application in Google Play Store, and showing the impacts of variables on the successful app. This model can be beneficial to developers who want to create new application on the App Store. Besides, developers of available apps also gain advantageous insights from the model to update their apps with the latest trend. From the Google Play Store aspect, the best app is regarded as their best product. Their business strategies can be affected by the most successful application gained by the predictive model.

## Dataset

The dataset we have used in this project was retrieved, “Google Play Store Apps” from Kaggle which contained 10841 observation and 13 variables. This information is scraped from the Google Play Store. The Play Store apps data has enormous potential to drive app-making businesses to success. The variables which is important for our project are:

1. App: Application name. It is a Factor variable
2. Category: Category the app belongs to. It is a categorical variable with 33 categories.
3. Rating: Overall user rating of the app. It is a continuous variable
4. Reviews: Number of user reviews for the app. It is a continuous variable
5. Installs: Number of user downloads/installs for the app
6. Type: paid or Free. It is a categorical variable with 2 category which are paid or free
7. price: Price of the app. It is a continuous variable

## Method used

Data mining is the main method used in this project. We used the clustering technique of data mining. The clustering technique was used because the training sample is not known in our dataset, so it counts as an unsupervised learning technique.

We want to identify and analyze the correlation between variables; therefore, it is better to apply Regression Analysis technique.

There are many variables involved in the model, therefore we decided to apply Multiple Linear Regression technique. Here is the formula of the linear regression;

Where, y is the dependent variable (the successful application in the Google Play Store)

xi1, xi2, …, xip are independent variables (rating, number of installs, and so forth).

β0 is y-intercept (the estimated value of y without the impact of other variables)

β1, β2, …, βp are coefficients corresponding to independent variables

ε is residuals or model’s error.

Here are some assumptions made to produce the model;

The applications which does not have the rating value and have number of reviews less than 10, will have values of rating is equal to zero.

## Research questions

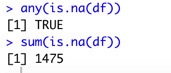
With the help of this analysis, we were able to answer some questions

* Which application is going to be successful in future and worth to investigate on it?
* Do the variables in our dataset impact the number of Installs for any application in Play store?

# Analysis

## Data Cleaning

Before working on the data, it is necessary to check and clean the null values in the dataset which can cause errors when running functions. Here is the total information about the null in the dataset.

****

**Picture 1**

A screenshot of a cell phone

Description automatically generatedIt can be seen that there are 1475 null values in the dataset. This number can be illustrated by the chart below using plot\_missing in package “DataExplorer”.

**Figure 1. Deploy of Null in the Dataset.**

The most missing are in the Rating column, which constitutes 13.6% values of the column. The rule of clearing null data in this part is that the null ratings with reviews that are less than 10 is removed, while those null ratings with reviews that are more than 10 will be replaced with zero.

Following this rule, the original data will be extracted into two sections using the subset() function in R. The first section will include data with non-null values of rating, while the other section has null ratings with number of reviews more than 10. After that, two data sections are joined using rbind() function. By combining the two sections, we now have no null values and the dataset does not include null rating having reviews less than 10. The null rating remained in the combined dataset has been changed to zero and the dataset is now clean without any null values in the Rating column. Moreover, because the rating cannot be more than 5, we wrote a code to eliminate those rows with ratings more than 5.

After replacing the NAN with zero in the rating column, we still have one NAN, so we use the na.omit() function to delete that row.

Furthermore, in order to be certain of no NAN values we wrote a code to delete the duplicate rows, so all the values are unique.

Moreover, we use box plot in order to check if we have outliner data or not.

A screenshot of a cell phone

Description automatically generated

A close up of a mans face

Description automatically generatedA screenshot of a cell phone

Description automatically generated**Figure 2: Box plot for Install**

**Figure 3: Box plot for Reviews**

A close up of a person

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**Figure 4: Box plot for Rating Figure 5: Box plot for Price**

As you can see in all of these box plots, we have a lot of outliners. Because the most outliner is for reviews, we wrote a code based on that in order to make a dataset clear and more normal.

Basically, we run the summary function and delete those rows which have review less than the minimum (which is equal to 1) as shown in picture 2 below;



**Picture 2**

And the result is:

A close up of a mans face

Description automatically generated

**Figure 6: box plot for Review**

As a result of this, the box plot for Reviews got better. However, it still needs cleaning, but because those applications which have a lot of reviews are so important for our model, we do not delete the upper outliners and decided to keep them.

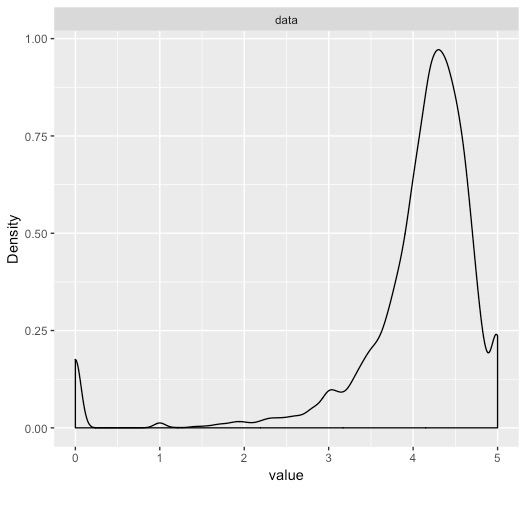
As a result of this above steps, now we have a dataset with 7518 rows instead of 10841.

## Exploratory Data Analysis

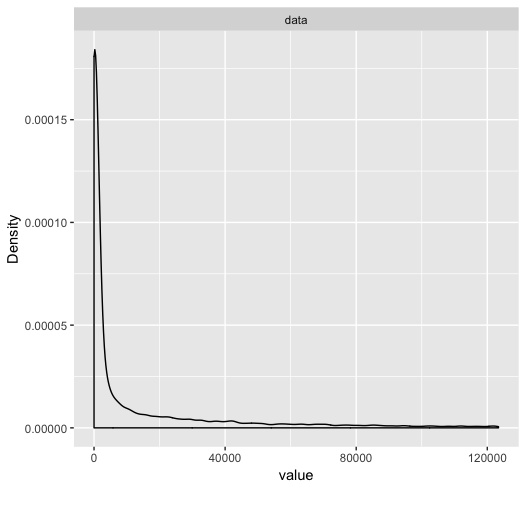
Now, we have a clean dataset which we can use for our analysis, but before that we need to understand our dataset in a better way. For understanding the distribution of data, we are going to draw the histogram plot for Continuous variables, and bar plot for categorical variables.

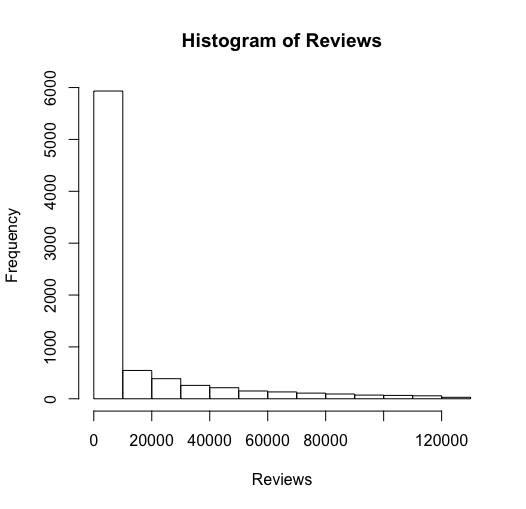
A screenshot of a cell phone

Description automatically generated

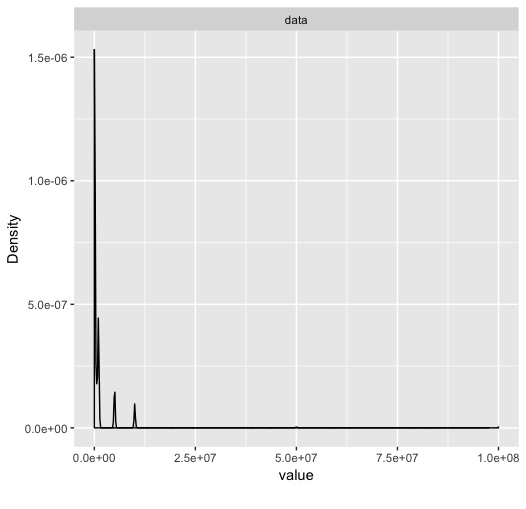
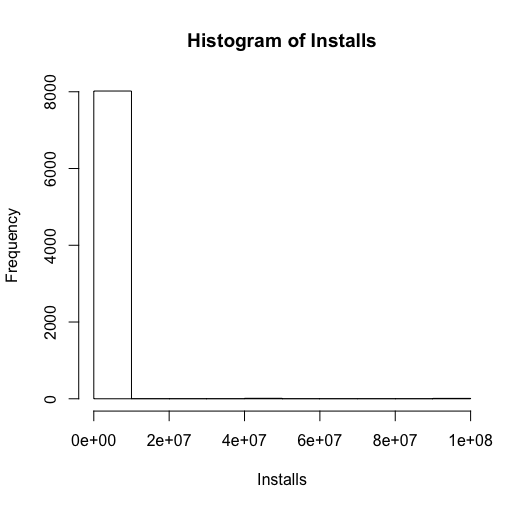


**Figure 7: Histogram for Rating Figure 8: Density plot for Rating**

From the above plots, the Rating does not have a normal distribution, instead it has bimodal distribution with skewed to the left. The reason that we have peak in zero is because we replace the missing data in rating by zero. And we can say expect of those which have zero rate, other application mostly has rate between 4 to 5.

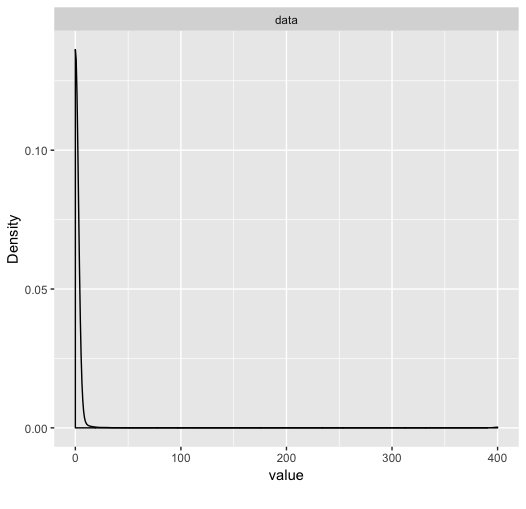
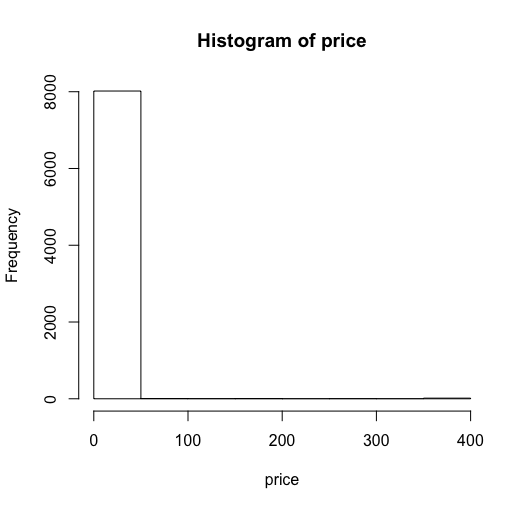


**Figure 9: Histogram for Reviews Figure 10: Density plot for Reviews**

As you can see in the plot, the reviews do not have normal distribution and it has skewed to the right because we have a lot of outliner reviews which shows we have some outstanding application with a huge popularity. In the same time, most of the application does not have reviews more than 20 thousand.

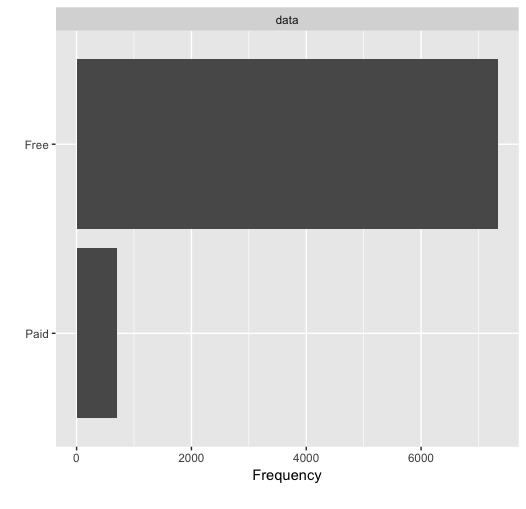
**Figure 11: Histogram for Rating Figure 12: Density plot for Rating**

Installs doesn't have normal distribution and it has skewed to the right.

As you can see the install and reviews have same shape of histogram and frequency. This result is predictable because the application should be installed first so the user can write the review about it. So, these we can say there is a positive correlation between install and reviews and the number of reviews dependent on the number of installs. expect those outstanding application, most of the application got installs less than 20,000 thousand. 

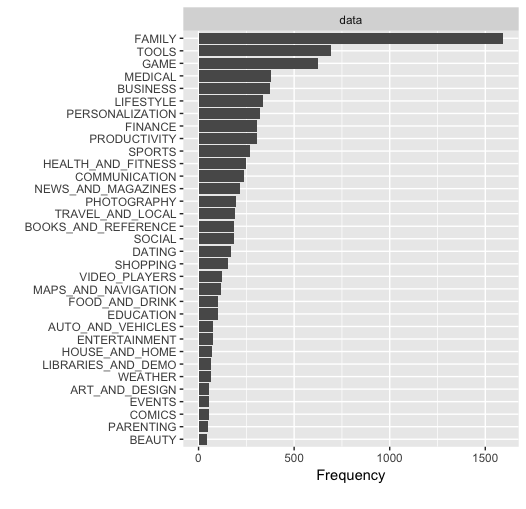
**Figure 13: Histogram for Rating Figure 14: Density plot for Rating**

Price doesn't have normal distribution and it has skewed to the right.



**Figure 15: Bar plot for Type**

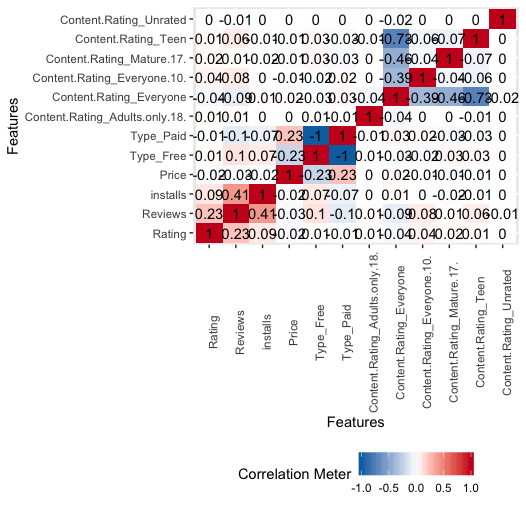
Based on plot, most of the application in google store are free. These is the reason that we do not have normal distribution with skewed to the left for price, because most of the application are free.



**Figure 16: Bar plot for Categories**

As you can see in the box plot of category, the most popular category for application in google play store is family. And by huge difference, the Tool category get the second place. And the popularity of beauty, parenting, comics and events application are almost zero.

Finally, we draw a correlation plot for all of the variables in our dataset.



**Figure 17: Correlation plot of the dataset**

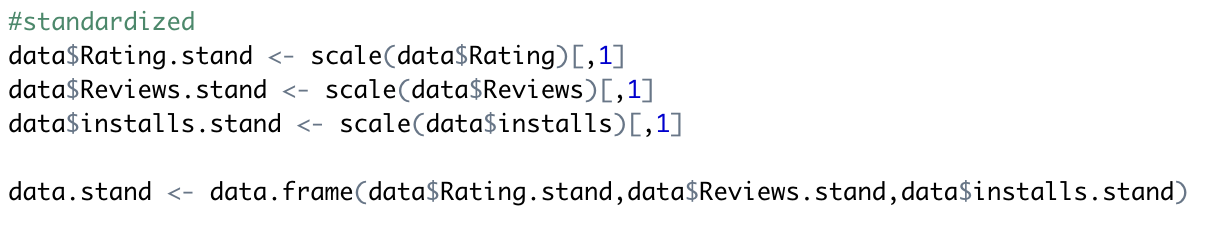
Base on this plot, we have most positive correlation between price and type free, and most negative correlation between paid and free and after that it is between free and paid.

As you can see in this plot, the most positive correlation is between install and reviews, and the most negative correlation between the variables which is important for us, is between price and type free. But in general, the most negative correlation is between content.rating\_Teen and content.rating\_everyone.

## Data Mining Clustering Method

Now, we understand the dataset in a better way, we are going to make a clustering in order to find out which group of application have better chance of become successful in google play store.

We are going to do the clustering based on three parameters: Install, Rating and Reviews.

Since these variables have different range, the first step for clustering is make our parameters standard. Because the clustering method compare values based on their similarity and dissimilarity to sing them in to the cluster.

**Picture 3: Performing standardizations**

Now, we have standard dataset and each value is between -1 and 1.

The method that we going to use for clustering is k-mean method. since we do not know how many clusters we should have, the next step is calculating the optimum number of clusters. There are three general methods to do this task, but we decided to use Elbow method, because it is most common method for finding the optimum number of clusters.

A close up of a map

Description automatically generated

**Figure 18: Plot for Optimal number of Cluster**

Based on this plot, the best number for cluster is 6.

Now we know how many clusters we need; we can use the k-mean method for clustering. And the result is:

A close up of a map

Description automatically generated

**Figure 19: Plot for Clusters**

For better understanding the clusters, we make a table based on the cluster and categories:

|  | **1** | **2** | **3** | **4** | **5** | **6** |
| --- | --- | --- | --- | --- | --- | --- |
| ART\_AND\_DESIGN | 2 | 0 | 28 | 4 | 1 | 22 |
| AUTO\_AND\_VEHICLES | 8 | 1 | 28 | 4 | 1 | 29 |
| BEAUTY | 1 | 1 | 16 | 3 | 6 | 21 |
| BOOKS\_AND\_REFERENCE | 16 | 5 | 74 | 18 | 6 | 38 |
| BUSINESS | 46 | 6 | 90 | 25 | 8 | 88 |
| COMICS | 10 | 0 | 24 | 4 | 1 | 16 |
| COMMUNICATION | 22 | 16 | 35 | 37 | 7 | 90 |
| DATING | 34 | 2 | 30 | 17 | 7 | 62 |
| EDUCATION | 1 | 5 | 33 | 23 | 1 | 43 |
| ENTERTAINMENT | 2 | 10 | 9 | 18 | 0 | 38 |
| EVENTS | 1 | 0 | 29 | 2 | 6 | 13 |
| FAMILY | 146 | 52 | 518 | 178 | 48 | 602 |
| FINANCE | 34 | 12 | 95 | 34 | 10 | 114 |
| FOOD\_AND\_DRINK | 10 | 8 | 28 | 20 | 4 | 30 |
| GAME | 36 | 70 | 163 | 113 | 10 | 262 |
| HEALTH\_AND\_FITNESS | 31 | 17 | 89 | 33 | 5 | 49 |
| HOUSE\_AND\_HOME | 4 | 1 | 11 | 13 | 4 | 34 |
| LIBRARIES\_AND\_DEMO | 3 | 3 | 21 | 3 | 13 | 33 |
| LIFESTYLE | 51 | 10 | 105 | 20 | 8 | 104 |
| MAPS\_AND\_NAVIGATION | 20 | 5 | 19 | 22 | 3 | 47 |
| MEDICAL | 36 | 1 | 139 | 11 | 8 | 112 |
| NEWS\_AND\_MAGAZINES | 25 | 3 | 54 | 29 | 14 | 76 |
| PARENTING | 3 | 0 | 27 | 2 | 10 | 17 |
| PERSONALIZATION | 11 | 12 | 114 | 38 | 10 | 97 |
| PHOTOGRAPHY | 29 | 24 | 37 | 44 | 1 | 67 |
| PRODUCTIVITY | 28 | 17 | 66 | 39 | 3 | 102 |
| SHOPPING | 9 | 16 | 41 | 32 | 6 | 54 |
| SOCIAL | 12 | 3 | 51 | 32 | 6 | 62 |
| SPORTS | 19 | 19 | 82 | 43 | 10 | 72 |
| TOOLS | 108 | 44 | 133 | 65 | 19 | 287 |
| TRAVEL\_AND\_LOCAL | 25 | 9 | 41 | 28 | 5 | 69 |
| VIDEO\_PLAYERS | 25 | 12 | 22 | 12 | 4 | 49 |
| WEATHER | 3 | 3 | 20 | 10 | 2 | 26 |

**Table 1: Cluster based on Categories**

The most successful application is the Family category and it belongs in cluster 6.

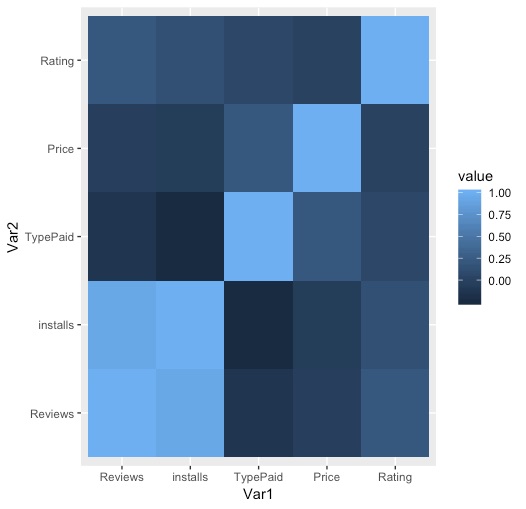
In cluster 6, the center of standard Rating is 0.0224, the center of standard Reviews is -0.3674 and the center of standard Installs is – 0.1963.

## Linear Regression

In order to check whether the variables such as Rating, Reviews, Price, Category and Content Rating in our dataset have any impact on the total number of installs per application in play store we have used the Linear Regression technique.

In this part, we are going to describe how we performed the Linear Regression technique based on our data using R.To begin, since we have the different levels of the data as predictor variables, we have to normalize the variables to the range between 0 and 1, so we can perform the linear regression operations and fit the model accurately. After normalizing the variables that are needed to develop our model accordingly, the values of each variable fell between the range of 0 and 1 (as predicted) for accurate analysis and to make a good fit model.

Once we have normalized the required variables, we set the seed for the data so as to use the constant samples for multiple executions. After that, we partitioned the data into the train sample and test sample with probabilistic ratio accordingly. Based on the outputs, we observed that the train sample and test sample have partitioned from the main data frame accordingly with 80 and 20 probability ratios.

Furthermore**,** we plot the correlation matrix accordingly in order to find out if there are any patterns or relations between the variables present in our dataset. Below you can see the result of performing the correlation matrix.

**Figure 20: Correlation Plot**

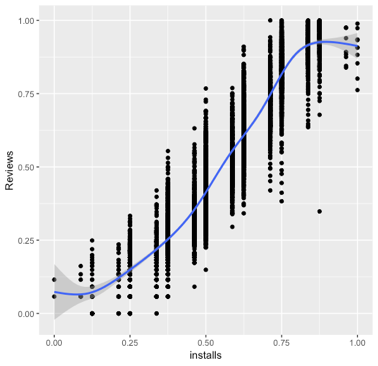
Based on the above graph, we can observe that there is a positive correlation between the variables Reviews and Installs, and we have very slight positive correlation between other variables such as Type and Price, and Rating and Reviews. So, based on this we can define that if we perform the Exploratory Data Analysis (EDA) on variables such as Reviews, Rating, Price and Type with the respective response variable install, we will observe that there are better patterns or some useful information that can be drawn from the analysis.

A picture containing wall

Description automatically generatedWe also performed the exploratory data analysis for the response variable with the respective model predictors to find out some information or any patterns. Below, you can see the results of EDA on respective predictors vs response variables;

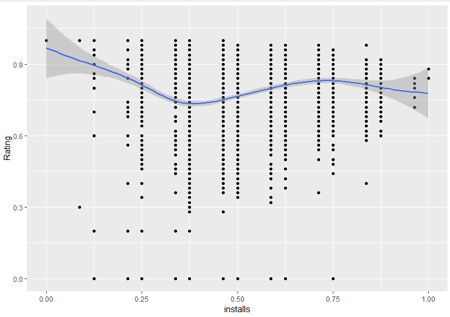
**Figure 21: Installs Vs Price**

This result above tells us that as the number of installs increases there was a very slight reduce in the price, and this suggests that there may be some relationship between the response variable and the predictor variable price.



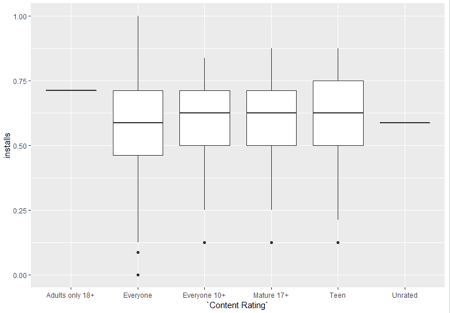
**Figure 22: Installs Vs Reviews**

The above graph tells us that as the number of reviews increases there was a very steep increase in the installs, and this suggests strongly that there may be some relationship between the response variable installs and the predictor variable reviews.



**Figure 23: Installs vs Rating**

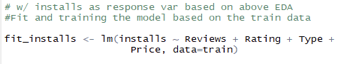
Based on the above graph, we can observe that as the number of ratings is high there was a slight fluctuation in the number of installs, and this suggests that there might have some relationship between these the response variable installs and the predictor variable rating.



**Figure 24: Installs Vs Content Rating**

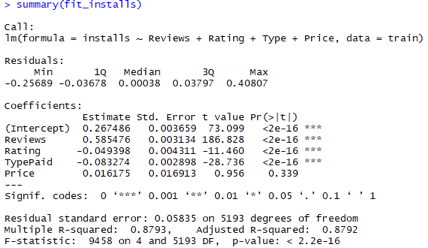
This shows that there is no relationship between the response variable installs and the model predictor content rating. As we can see, there was no drastic changes with respect to both the variables.

Based on these analyses, we have noticed that variables such as reviews, ratings, price and type are the most important variables in order to predict the impact on the number of installs in terms of any application. Hence, we have defined the above-mentioned variables as final model predictors in order to fit the linear regression model with respect to the response variable. Below are the R commands we have used in order to fit and develop a model based on the train data.



**Picture 4: Linear Regression R code**

The results from fitting the linear model with respect to the train data are summarized below;



**Picture 5: Summary of Linear Regression Model**

Based on the above outputs, it can be observed that we have our Adjusted R-squared value as 0.8792. As we know, if the adjusted R-squared value is close to 1 and the P-value is less than the significant level value, it means that the model has the best fit with very accurate predictions. Which means our model has a good fit, because it is close to 1. Knowing that, we have our model as;

**Installs = 0.585 Reviews- 0.049 Rating-0.083 TypePaid+0.016 Price + e,**

After, developing the model based on the train sample, we now need to test the model by implementing it on the test sample as well. To check the model accuracy and prediction, we used Root Mean square error (RMSE), which defines the model predictions that if the RMSE value is less or close to 0, then the model developed is accurate.

**Picture 6: RMSE of Train sample and Test sample**

The above outputs show that the RMSE value of model developed with respect to both the train data and test data is almost similar, and also it is very close to zero. Which means that the linear model we have developed have the best fit, and have drawn the conclusion that the number of installs of an application in the play store has some good impact based on the other variables such as reviews, rating, price and the type of an application in play store.

# Conclusion

Based on our analysis on this dataset, we were able to find out answers to our analysis questions;

1. Which application is going to be successful in future and worth to investigate on it?
2. Do the variables in our dataset has impact the number of Installs for any application in Google Play store?

To answer those questions, we worked with variables that we think are the most important, like the average of rating, number of reviews, number of Installs, price, type of application ( which is free and paid) and category of application.

First and foremost, we made sure that we were going to use a clean dataset for our analysis method. Since our dataset had some missing values, it counted as a dirty dataset. So, we used some data cleaning methods to replace and remove missing data and outliers. Since we have a lot of outstanding applications that have huge number of reviews and installs, we decided to not delete those application data because they are really important and were considered successful. Since our main objective in this project is to look for successful applications, that is why we cannot delete them.

After cleaning , we used Data mining - Clustering method - in order to put the applications in different classes based on similarity and dissimilarity. But, before that, we had to standardize the variables because each of them has different ranges and we cannot compare them to each other. For finding the number of clusters that we needed, we used the Elbow method. As a result, the optimum number of clusters is 6. After clustering the dataset, we extracted them base on categories to see which category and cluster have the most members, because any application that falls into those classes counts as a successful application. Base on the clustering results, most of the application fell under the Family category - cluster 6, Family category - cluster 3, and Tools category – cluster 6.

Furthermore, we used linear regression to see if Reviews, Ratings, Type, Price, and Content Rating have any impact on Installs. To do this, we first normalized the variables and then divided the dataset into two samples: Train sample and Test sample. Then we plot the correlation between variables in Train sample to see if they have any impact on each other.

Based on the result from the correlation plot, we can observe that there is a positive correlation between the variables Reviews and Installs and we have very slight positive correlation between other variables such as Type and Price, Rating and Reviews. Finally, we use the lm() function to draw out our prediction and coefficients between the main variables and Installs, and the result model from the Train sample has its RMSE of 0.058 which is so close to zero. And just to be sure, we tested the result model on the Test sample and the RMSE is 0.057. This shows that the model we have is good.

# References

Alex Makarevich. (2018). *Alternative Google Play app stores to consider*. Retrieved from <https://appdevelopermagazine.com/alternative-google-play-app-stores-to-consider/>

Ben Popper. (2017). Google announces over 2 billion monthly active devices on Android. Retrieved from <https://www.theverge.com/2017/5/17/15654454/android-reaches-2-billion-monthly-active-users>

Statista. (2019). *Number of apps available in leading app stores as of 1st quarter 2019*. Retrieved from <https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/>

# Appendix

## R Codes:

##Retrieving the Data to the Data frame

install.packages("readr")

library(readr)

df <- read\_csv(file.choose(),col\_names = TRUE)

###Checking for missing Data

any(is.na(df))

sum(is.na(df))

install.packages("DataExplorer")

library(DataExplorer)

plot\_missing(df)

###Cleaning dataset

##cleaning Rating column

#extract observations having RATING not NaN

df1 <- subset(df, df$Rating != "NaN")

#extract observations having RATING NaN and REVIEW more than 10

df2 <- subset(df, df$Rating == "NaN" & df$Reviews > 10)

#Binding both the data frames together

data <- rbind(df1,df2)

#remove NaN from AppData3, then using data as the final dataset

data[["Rating"]][is.na(data[["Rating"]])] <- 0

# Since the rating cannot be more than 5, we have deleted the rows that have rating more than 5

data<- data[!(data$Rating>5),]

##cleaning other columns missing value

data <- na.omit(data)

##checking the count of NA

sum(is.na(data))

##checking the Duplicates and removing them accordingly

data <- unique(data)

##checking the dataset

str(data)

nrow(data)

ncol(data)

##checking for outliers

library(ggplot2)

boxplot(log(data$installs))

boxplot(data$installs)

boxplot(log(data$Reviews))

boxplot(data$Reviews)

boxplot(log(data$Rating))

boxplot(data$Rating)

boxplot(data$Price)

#delete outliers (outlier less than lower limit)

summary(data$Reviews)

data<- data[!(data$Reviews<1),]

outliers.reviews <- boxplot(data$Reviews, plot=FALSE)$out

data <- data[-which(data$Reviews %in% outliers.reviews),]

#checking the outliers again

boxplot(log(data$Reviews))

boxplot(data$Reviews)

###EDA

str(data)

nrow(data)

ncol(data)

##plot for continuous parameters

#install.packages("DataExplorer")

library(DataExplorer)

hist(data$Rating, main = "Histogram of Rating", xlab = "Rating")

plot\_density(data$Rating)

hist(data$Reviews,main = "Histogram of Reviews", xlab ="Reviews")

plot\_density(data$Reviews)

hist(data$installs,main = "Histogram of Installs", xlab ="Installs")

plot\_density(data$installs)

hist(data$Price,main = "Histogram of price",xlab ="price")

plot\_density(data$Price)

##Plot for categorical parameters

plot\_bar(data$Category)

plot\_bar(data$Type)

##correlation plot for all of the parameters that we going to use in model

plot\_correlation(data)

###CLUSTERING

##standardized

data$Rating.stand <- scale(data$Rating)[,1]

data$Reviews.stand <- scale(data$Reviews)[,1]

data$installs.stand <- scale(data$installs)[,1]

data.stand <- data.frame(data$Rating.stand,data$Reviews.stand,data$installs.stand)

##Determining number of clusters

install.packages("factoextra")

library(factoextra)

install.packages("NbClust")

library(NbClust)

fviz\_nbclust(data.stand, kmeans, method = "wss") +labs(subtitle = "Elbow method")

##k-means cluster analysis

result <- kmeans(data.stand,6)

attributes(result)

##plot clusters

library(ggplot2)

install.packages("cluster")

library(cluster)

fviz\_cluster(result,data = data.stand, geom = "point", stand = FALSE,frame.type = "norm")+theme\_bw()

##To check the clusters with datapoints individually

table(data$Category,result$cluster)

##Information about Clusters

result$centers

###LINEAR REGRESSION for predicting the impact of other variables on the number of installs

install.packages("modelr")

library(modelr) #required for Finding RMSE for model

install.packages("reshape2")

library(reshape2) #required for melt\_mat function during correlation matrix

## Normalize data

normalize <- function(vec, log\_=FALSE) {

if (log\_) {

vec <- log(vec)

}

min <- min(vec)

max <- max(vec)

new\_vec <- (vec - min) / (max - min)

return(new\_vec)

}

##Duplicating the dataframe so as to maintain backup of the original data

data1 <- data

##Normalizing the required variables

data1$Rating <- normalize(data$Rating)

data1$Reviews <- normalize(data$Reviews, log\_=TRUE)

data1$installs <- normalize(data$installs, log\_=TRUE)

data1$Price <- normalize(data$Price)

##Specifying seed

set.seed(12)

##Partition of the data

train\_data <- sample(1:nrow(data1), as.integer(nrow(data1) \* 0.8))

test\_data <- setdiff(1:nrow(data1), train\_data)

train <- data1[train\_data, ]

test <- data1[test\_data, ]

summary(train)

train <- as.data.frame(train)

str(train)

##Correlation matrix for train sample

mat <- model.matrix(~ Reviews + installs + Type + Price +

Rating, data=train)

melt\_mat <- melt(cor(mat[, -1]))

ggplot(melt\_mat, aes(x=Var1, y=Var2, fill=value)) + geom\_tile()

##EDA for response variable vs model predictors

#installs v. Price

install.packages("dplyr")

library(dplyr)

train %>%

filter(Price < 50) %>%

ggplot(aes(installs,Price)) +

geom\_point() +

geom\_smooth()

#installs v. Reviews

train %>%

filter(Reviews < 10e+3) %>%

ggplot(aes(installs,Reviews)) +

geom\_point() +

geom\_smooth()

#installs v.Rating

train %>%

ggplot(aes(installs, Rating)) +

geom\_point() +

geom\_smooth()

#installs v.Content Rating

train %>%

ggplot(aes(`Content Rating`,installs)) +

geom\_boxplot()

##Fit and training the model based on the train data

fit\_installs <- lm(installs ~ Reviews + Rating + Type +Price, data=train)

##Summary of the trained/developed model

summary(fit\_installs)

##Retrieving RMSE value to know the model accuracy

rmse(fit\_installs, train)

rmse(fit\_installs,test)