PES UNIVERSITY

Hosur Road, Electronic City-560100, Bangalore, INDIA

R Programming
PROJECT REPORT
on

"Google play store Analysis"

Submitted in partial fulfillment of the requirements for the III Semester SPECIAL TOPIC on PROGRAMMING WITH R (UE18CS208D)

Bachelor of Engineering
IN
COMPUTER SCIENCE AND ENGINEERING

For the Academic year
2019-2020

BY

Akash Yadav
PES2201800415

Sanjith S Jadhav
PES2201800318

Under the Guidance of

Dr. Pooja Agarwal
Professor, Dept. of CSE
PESU-EC Campus, Bengaluru-560100

Department of Computer Science and Engineering

PES UNIVERSITY

Hosur Road, Electronic City-560100, Bangalore, INDIA

PES UNIVERSITY

Hosur Road, Electronic City-560100, Bangalore, INDIA

Department of Computer Science and Engineering

CERTIFICATE

Certified that the project work entitled "Google Play Store Analysis" is a bonafide work carried out by Students bearing SRNs: PES2201800415 and PES2201800318, students of PES University, Electronic City, Bangalore in partial fulfillment for the requirement of Bachelor of Engineering III Semester special topic on Programming with R (UE18CS208D) during the year 2019-2020. It is certified that all corrections/suggestions indicated for assessment have been incorporated and the project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said course.

Signatures:

____

Project Guide
Dr. Pooja Agarwal
Professor, Dept. of CSE
PES University- EC Campus,
Bengaluru

ABSTRACT

The intention of compiling this project is to study and to visualize the effect of various factors on each other. These factors are related to the app downloads on the Google play store. We also try to build a linear model which would be able to predict the effect of one variable on the outcome of another. Through this predictive model we also try to find the relation between the two chosen variables. The two chosen variables here are Ratings of the apps and the number of Reviews made by the user. The broad objectives are:

1. Categorize apps based on their popularity.
2. To conclude which kind of apps have the highest revenue.
3. Categorize the number of downloads in each genre of apps.
4. To find, if there is a relation between the number of app reviews and their ratings.

TABLE OF CONTENTS

1. INTRODUCTION

With apps being downloaded in millions, it becomes important for the app developers to keep track of what is affecting the number of downloads that are taking place. There are various factors which affect the number of downloads like ratings, reviews, size etc. Through this project we desire to graphically analyse the co-dependency of these factors on each other and also to find a relation, if any, between these factors. This project is divided into two sub-groups , Firstly , Visualisation through graphical methods and lastly , to build a predictive model which can predict the affect one factor has on another one. The two factors chosen here are Ratings of the app and the number of Reviews made by the Users. So our project basically revolves around these two objectives.

2. LITERATURE SURVEY

<u>Literature Survey 2. 1: A Longitudinal Study of App Permission Usage across the Google Play Store</u>

<u>Cite:</u> J. H. Saltzer and M. D. Schroeder. The protection of information in computer systems. Proceedings of the IEEE, 63(9):1278–1308, 1975 N. Viennot, E. Garcia, and J. Nieh. A Measurement Study of Google Play. In The 2014 ACM International Conference on Measurement and Modeling of
Computer Systems, SIGMETRICS ’14, pages 221–233, New York, NY, USA, 2014. ACM

<u>Reference:</u>
https://pdfs.semanticscholar.org/2ae8/2b0593124d236ea9f5202d48760ce69c644e.pdf

<u>Abstract:</u> Although there are over 1,600,000 third-party Android apps in the Google Play Store ans very little is known about how their individual permissions requirements have evolved over time. This work focuses exclusively on dangerous permissions, i.e., those permissions that guard access to sensitive user data. The permissions required by an app usually relate to the provision of the app’s functionality, but some apps are intentionally over-privileged to facilitate greater access to a user’s personal data.

<u>Method:</u> By taking snapshots of the Google Play Store over a one year period, we characterize changes in the number and type of dangerous permissions required by Android apps when they are updated, to gain a greater understanding of the evolution of permission usage in the Android app ecosystem. We built our own crawler that retrieved full app data (including permission usage) from the Google Play Store, by leveraging the list of apps from the GPSC project. We developed a cloud-based crawler, with geographically distributed worker nodes fetching app data and returning it to a command and control server.

<u>Perks:</u> Since Android 6.0, users are no longer forced to accept (or reject) permissions in their entirety at install-time, and can now accept (or reject)
permissions individually at run-time. This offers added control, but many users continue to accept permission requests blindly due to conditioning or lack of understanding.

<u>Faults:</u> We found that very popular apps (those with in excess of one million downloads) used more permissions than those with less downloads. We also discovered that apps were more likely to add new permissions than remove permissions, overall pointing to a potential erosion in privacy and security. Our data showed that more popular apps were more likely to add new permissions. This is worrying, because it means that a large segment of users in the

Android ecosystem are potentially exposed to additional privacy/security risks simply as a result of using and updating popular apps.

Literature Survey 2.2 - Google play store app ranking prediction

Abstract: This paper study consists of different machine learning algorithms used to predict an app rating on Google play store utilizing real-time dataset of more than 10,000 play store apps. These results are obtained by collection cleansing training and testing data to evaluate each regression model furthermore alter data to get desired results. Finally after implementation it concluded that linear regression fine tree algorithm provides best app rating prediction results.

Methodology : To apply prediction model and analyse data these basic steps has been followed. 1. Data collection: Find a suitable dataset that fulfil the requirement to apply prediction models. 2. Pre-process data: To make data in correct format some filtration functions have been applied. 3. Explore data: Fix if there is any irrelevant value, sparsity, null, repetition and error. 4. Filter data: Remove extra columns that effects computation time and memory utilization. 5. Distribution data: Divide data into training and testing module. 6. Train data: Train the algorithm with training data until a correct model with minimum errors is obtained. 7. Data Evaluation: Compare the model with the testing data 8. Observe data: Analyse results.

Dataset used in this research work is authentic android platform user data. It consists of more than ten thousand instances collected in 2018. As it says on dataset repository site, it is hypothetical data available for analyst, mobile developer and university researchers who perform algorithm to estimate mobile app performance

Conclusion: In this paper, Machine learning algorithms have been evaluates to predict app ranking on given dataset. The results show that, despite the wide variety of techniques and complex algorithms, which could be improved using a different dataset or adding app features in order to make more accurate predictions, one of the most simple techniques, Fine tree, have provided the best results in making predictions from play store historical data. That concluded that, it could be possible and not a hard task to implement tree algorithm on dataset to predict app ranking, in order to forecast the suitable rating against an app, helping to improve app positioning, manage trends in app store meeting the demand of the app stores optimization and making rating systems more accurate.

Cite: Muhammad Suleman College of Computer Science and Information Systems- CCSIS Institute of Business Management , Ahsan Malik College of Computer Science and Information Systems- CCSIS Institute of Business Management.

References: **https://www.researchgate.net/profile/Syed_Hassan92/publication/331889811**

3. Dataset Description

Name of the dataset: googleplaystore.csv

Size of the dataset: 7.26 MB (13 columns and 10841 rows)

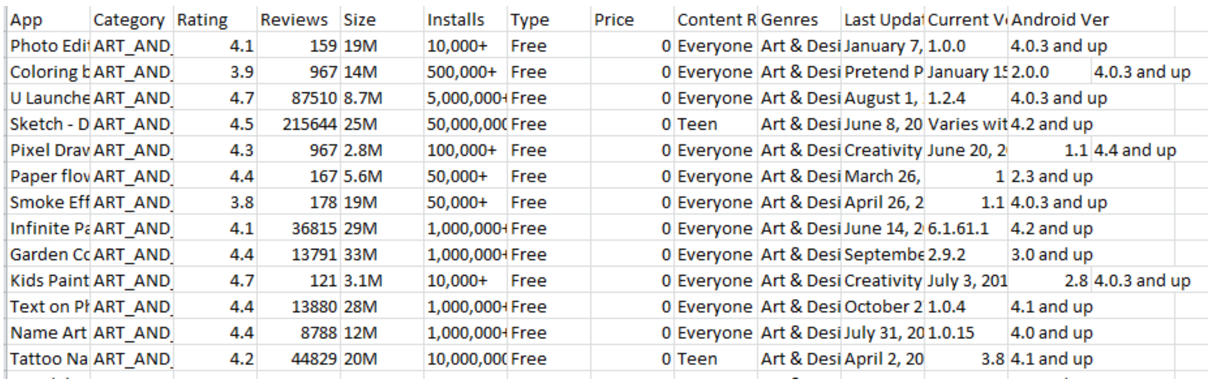
Number of numerical columns: 7

Number of categorical columns: 6

Link to the dataset: https://www.kaggle.com/lava18/google-play-store-apps

Number of Missing and NaN values: 1474 rows with missing values

Dataset Screenshot:-



App	Category	Rating	Reviews	Size	Installs	Type	Price	Content R	Genres	Last Updat	Current V	Android Ver	
Photo Edit	ART_AND_	4.1	159	19M	10,000+	Free	0	Everyone	Art & Desi	January 7,	1.0.0	4.0.3 and up	
Coloring b	ART_AND_	3.9	967	14M	500,000+	Free	0	Everyone	Art & Desi	Pretend P	January 15	2.0.0	4.0.3 and up
U Launche	ART_AND_	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Desi	August 1,	1.2.4	4.0.3 and up	
Sketch - D	ART_AND_	4.5	215644	25M	50,000,000	Free	0	Teen	Art & Desi	June 8, 20	Varies wit	4.2 and up	
Pixel Drav	ART_AND_	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Desi	Creativity	June 20, 2	1.1	4.4 and up
Paper flov	ART_AND_	4.4	167	5.6M	50,000+	Free	0	Everyone	Art & Desi	March 26,	1	2.3 and up	
Smoke Eff	ART_AND_	3.8	178	19M	50,000+	Free	0	Everyone	Art & Desi	April 26, 2	1.1	4.0.3 and up	
Infinite Pa	ART_AND_	4.1	36815	29M	1,000,000+	Free	0	Everyone	Art & Desi	June 14, 2	6.1.61.1	4.2 and up	
Garden Cc	ART_AND_	4.4	13791	33M	1,000,000+	Free	0	Everyone	Art & Desi	Septembe	2.9.2	3.0 and up	
Kids Paint	ART_AND_	4.7	121	3.1M	10,000+	Free	0	Everyone	Art & Desi	Creativity	July 3, 201	2.8	4.0.3 and up
Text on Ph	ART_AND_	4.4	13880	28M	1,000,000+	Free	0	Everyone	Art & Desi	October 2	1.0.4	4.1 and up	
Name Art	ART_AND_	4.4	8788	12M	1,000,000+	Free	0	Everyone	Art & Desi	July 31, 20	1.0.15	4.0 and up	
Tattoo Na	ART_AND_	4.2	44829	20M	10,000,000	Free	0	Teen	Art & Desi	April 2, 20	3.8	4.1 and up	

Figure 3.1. Snapshot of the dataset

3.2. Variable description (of the relevant variables, i.e., the variables which are useful to the model):

App: The name of the App as present in the Google Play Store

Category: The category to which the App belongs to.

Rating: The average rating given by the users to App.

Reviews: The total number of reviews for that App.

Size: The size of the App.

Installs: The total number of Installs of that App.

Type: Weather the App is free or paid.

Price: The price of the App.

Content Rating: To what division of people is the App developed for.

Genres: To what genres the App belongs to.

Last Updated: The last date a given App was updated.

Current Version: The current version on which the App is running.

Android Version: The range of Android Versions which the App supports.

4. Data Cleaning

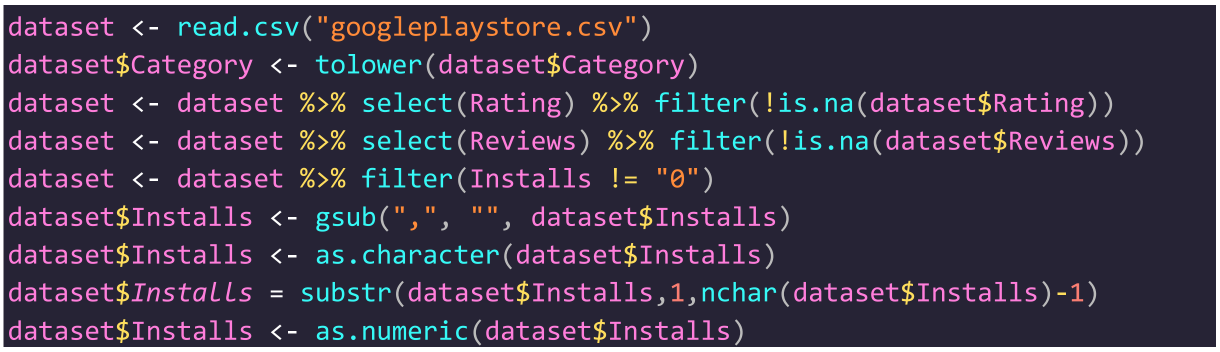
4.1. Cleaning and pre-processing of the dataset:

The following were the steps involved in the dataset cleaning step:-

- We converted each of the category names to lower case so that it could easily be read.
- The first step that we took was to remove all the NA values present in the rating column. It was found that there were nearly 1474 NA values present in this column.
- Next step was that we removed the rows which had Apps that were not installed even once or which did not have any reviews.
- It was found that the number of Installs columns had a lot of commas in it. Therefore these had to be removed to convert it to an integer.

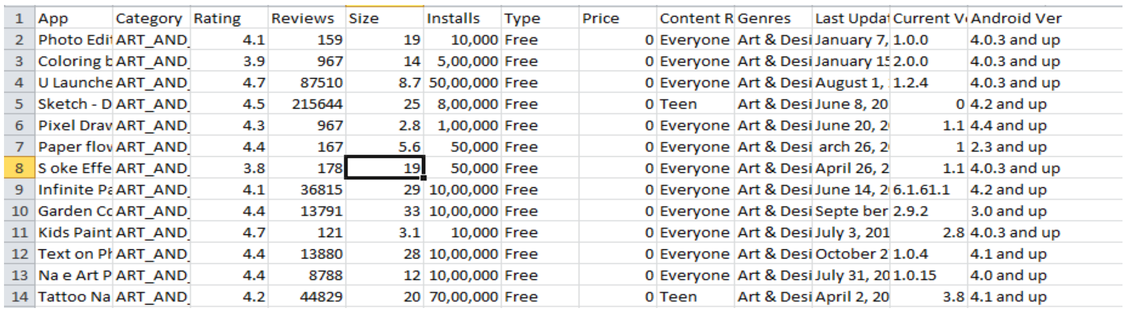
The above steps only consist of removing of the rows. Interpolation of rows was not acceptable as this would bring a discrepancy in the visualization of the data. On plotting the boxplot before and after data cleaning ensured that the outliers were gone.

The following is the code used for data cleaning:-



```
dataset <- read.csv("googleplaystore.csv")
dataset$Category <- tolower(dataset$Category)
dataset <- dataset %>% select(Rating) %>% filter(!is.na(dataset$Rating))
dataset <- dataset %>% select(Reviews) %>% filter(!is.na(dataset$Reviews))
dataset <- dataset %>% filter(Installs != "0")
dataset$Installs <- gsub(",", "", dataset$Installs)
dataset$Installs <- as.character(dataset$Installs)
dataset$Installs = substr(dataset$Installs,1,nchar(dataset$Installs)-1)
dataset$Installs <- as.numeric(dataset$Installs)
```

Figure 4.1. Cleaning the dataset



1	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content R	Genres	Last Updat	Current V	Android Ver
2	Photo Edit	ART_AND_	4.1	159	19	10,000	Free	0	Everyone	Art & Desi	January 7,	1.0.0	4.0.3 and up
3	Coloring b	ART_AND_	3.9	967	14	5,00,000	Free	0	Everyone	Art & Desi	January 15	2.0.0	4.0.3 and up
4	U Launche	ART_AND_	4.7	87510	8.7	50,00,000	Free	0	Everyone	Art & Desi	August 1,	1.2.4	4.0.3 and up
5	Sketch - D	ART_AND_	4.5	215644	25	8,00,000	Free	0	Teen	Art & Desi	June 8, 20	0	4.2 and up
6	Pixel Drav	ART_AND_	4.3	967	2.8	1,00,000	Free	0	Everyone	Art & Desi	June 20, 2	1.1	4.4 and up
7	Paper flov	ART_AND_	4.4	167	5.6	50,000	Free	0	Everyone	Art & Desi	arch 26, 2	1	2.3 and up
8	S oke Effe	ART_AND_	3.8	178	19	50,000	Free	0	Everyone	Art & Desi	April 26, 2	1.1	4.0.3 and up
9	Infinite Pa	ART_AND_	4.1	36815	29	10,00,000	Free	0	Everyone	Art & Desi	June 14, 2	6.1.61.1	4.2 and up
10	Garden Cc	ART_AND_	4.4	13791	33	10,00,000	Free	0	Everyone	Art & Desi	Septe ber	2.9.2	3.0 and up
11	Kids Paint	ART_AND_	4.7	121	3.1	10,000	Free	0	Everyone	Art & Desi	July 3, 201	2.8	4.0.3 and up
12	Text on Pl	ART_AND_	4.4	13880	28	10,00,000	Free	0	Everyone	Art & Desi	October 2	1.0.4	4.1 and up
13	Na e Art P	ART_AND_	4.4	8788	12	10,00,000	Free	0	Everyone	Art & Desi	July 31, 20	1.0.15	4.0 and up
14	Tattoo Na	ART_AND_	4.2	44829	20	70,00,000	Free	0	Teen	Art & Desi	April 2, 20	3.8	4.1 and up

Figure 4.2. Cleaned Dataset

5. Normalization

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. This could be done either by ranging the values between 0 and 1(normalization) or between -1 and +1(standardization).

We have implemented normalization to our dataset by the following code:-

```
normalize(dataset, method = "standardize", range = c(0, 1), margin = 1L,
on.constant = "quiet")
```

6. Graph Visualization

Graphs help us understand a dataset better by giving us pictorial representations of it.
It can help us better understand the problem statement and come up with a better solution.
Here are a few examples:

1. Most Installed Apps Category (Installs more than a billion)

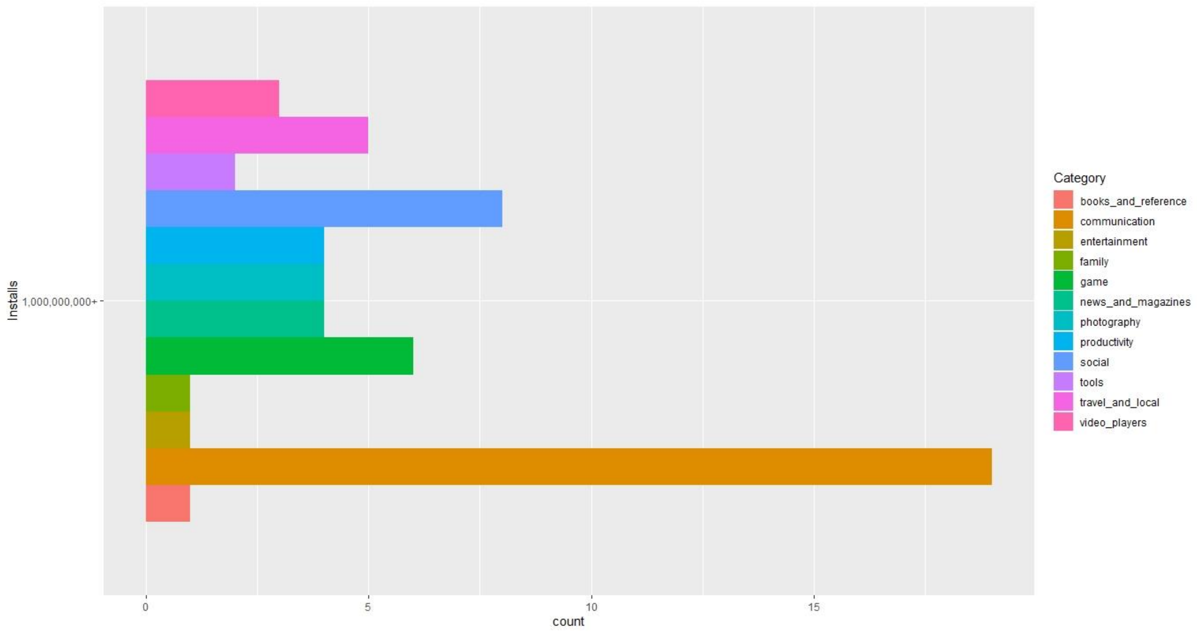
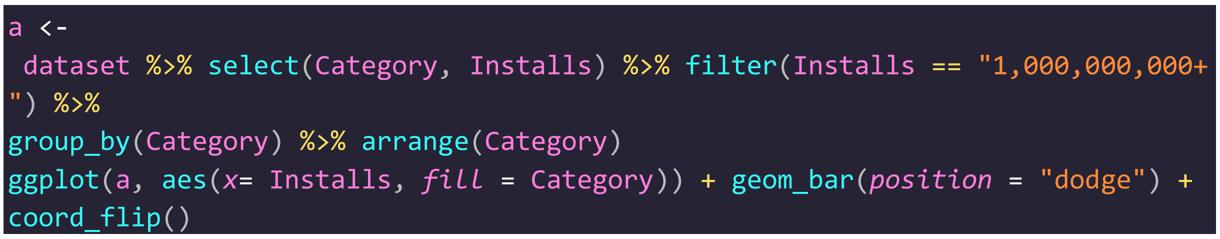


Figure 6.1: A graphs for installs vs count

The above tells us that communication; social and game apps are the most installed by over one billion people.

The code for the above graph is:-



```
a <-
 dataset %>% select(Category, Installs) %>% filter(Installs == "1,000,000,000+
") %>%
group_by(Category) %>% arrange(Category)
ggplot(a, aes(x= Installs, fill = Category)) + geom_bar(position = "dodge") +
coord_flip()
```

Figure 6.2 : Code for the above graph

2. How many apps are included in each category?

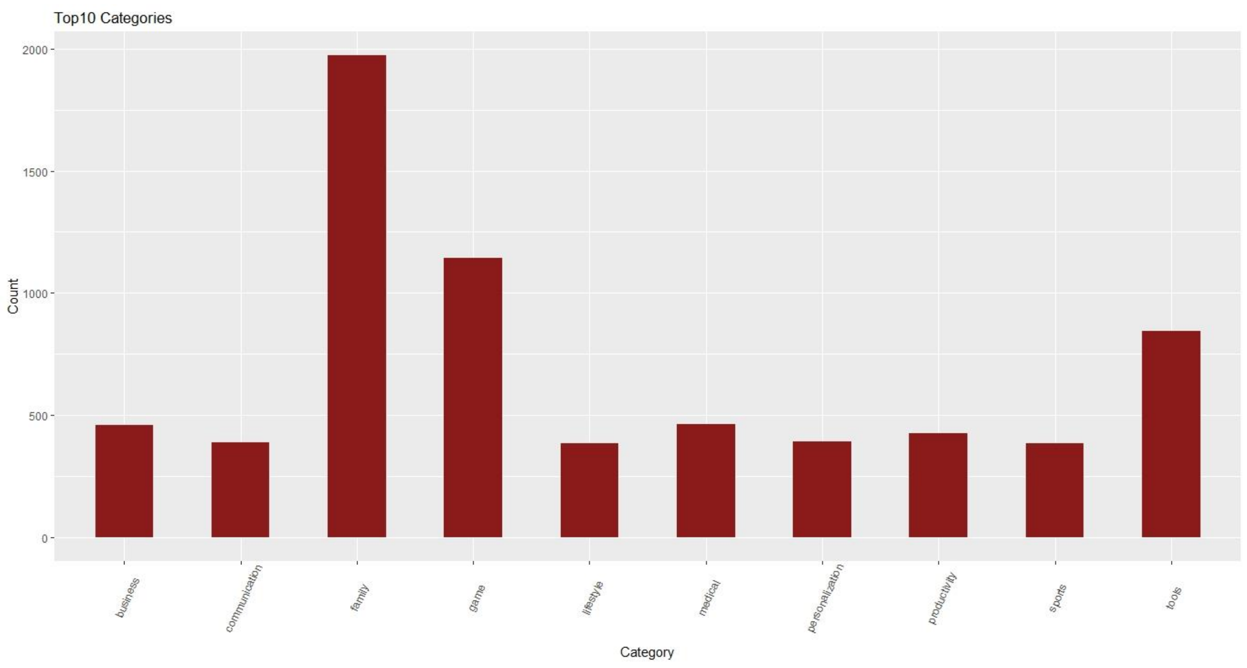


Figure 6.3 : A graph for count vs category

The above graph made us infer that Apps related to Family are the most frequent category followed by apps related to games.

The code for the above graph is:-

```
c <-
 dataset %>% group_by(Category) %>% summarize(Count = n()) %>% arrange(desc(Co
unt))
ggplot(c, aes(x = Category, y = Count)) + geom_bar(stat="identity", width=.5,
 fill="firebrick4") +
  labs(title = "Top10 Categories") + theme(axis.text.x = element_text(angle=65
, vjust=0.6))
```

Fig 6.4: Code for the above graph

3. Top 10 Paid Apps?

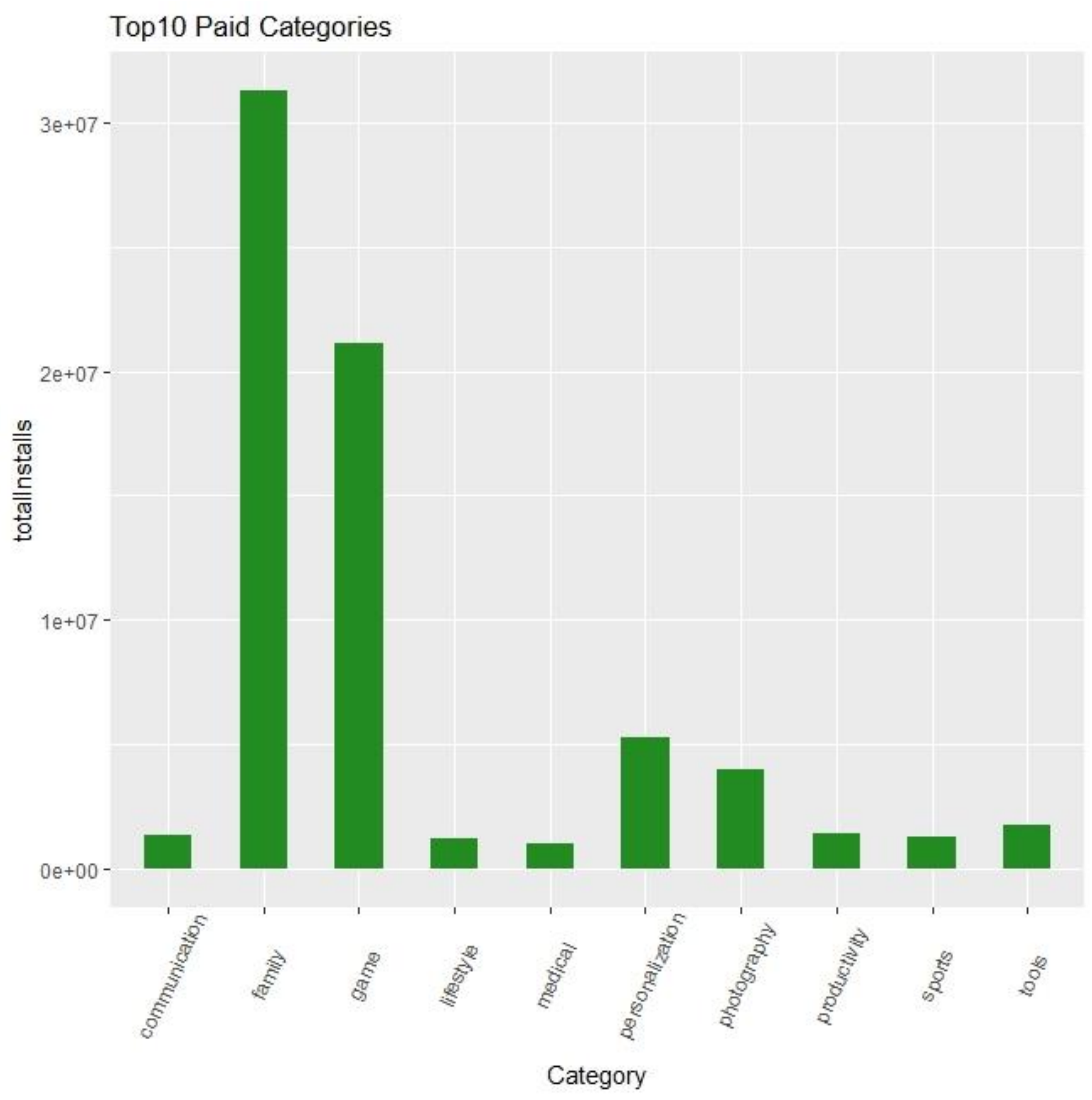
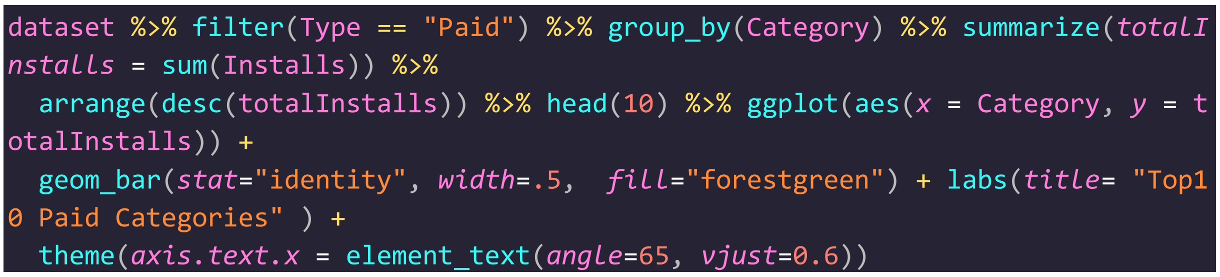


Figure 6.5 : A graph for installs vs category

Family Apps has the highest income as it has the most number of installs.

The code for the above graph is:-



```
dataset %>% filter(Type == "Paid") %>% group_by(Category) %>% summarize(totalI
nstalls = sum(Installs)) %>%
  arrange(desc(totalInstalls)) %>% head(10) %>% ggplot(aes(x = Category, y = t
otalInstalls)) +
  geom_bar(stat="identity", width=.5,  fill="forestgreen") + labs(title= "Top1
0 Paid Categories" ) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```

Fig 6.6: Code for the above graph

4. Average Rating across Apps

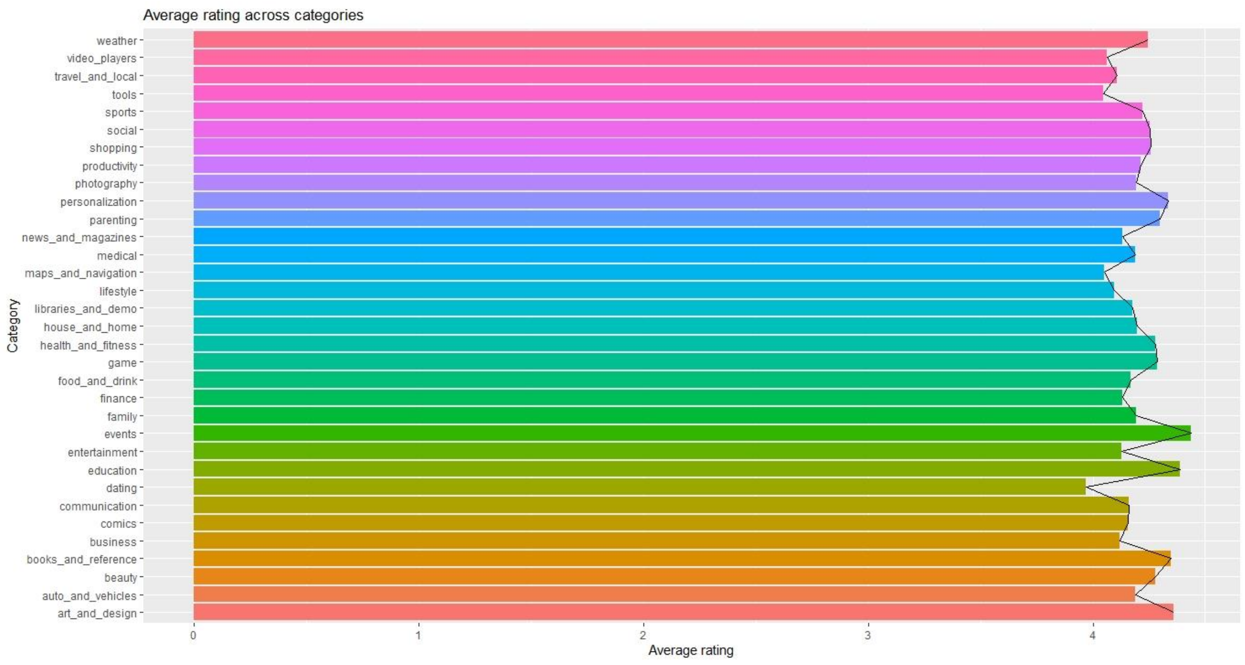


Figure 6.7 : A graph for installs vs category

Categories generally have a similar mean rating, however, Events, personalization, education and art applications all tend to have a minutely higher rating.

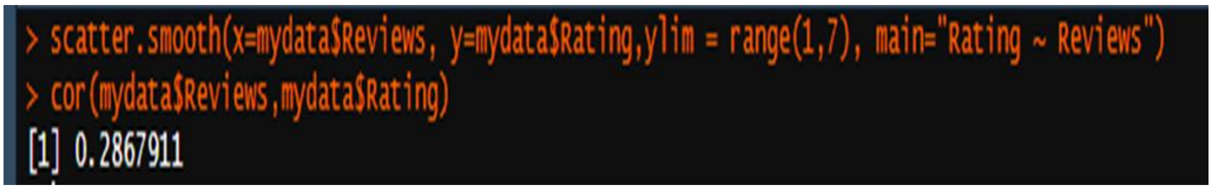
The code for the above graph is:-

```
dataset %>%  group_by(Category) %>%  filter(!is.na(Rating), Category!='1.9') %
>%
summarise(meanRating = mean(Rating)) %>% ggplot(mapping = aes(x = Category, y
= meanRating)) +
geom_col(aes(fill = Category)) + geom_line(group = 1) +  coord_flip() +
ggtitle("Average rating across categories") + ylab("Average rating") + guides(
fill=FALSE)
```

Fig 6.8: Code for the above graph

7. Correlation/ Regression/ Classification

Linear regression is one of the most commonly used predictive modelling techniques. The aim of linear regression is to find a mathematical equation for a continuous response variable Y as a function of one or more X variable(s). So that you can use this regression model to predict the Y when only the X is known.
Now that the data has been pre-processed we will find the co-relation between the two variables i.e. Ratings and the Reviews column.
We will use the command ‘cor’ to do this.

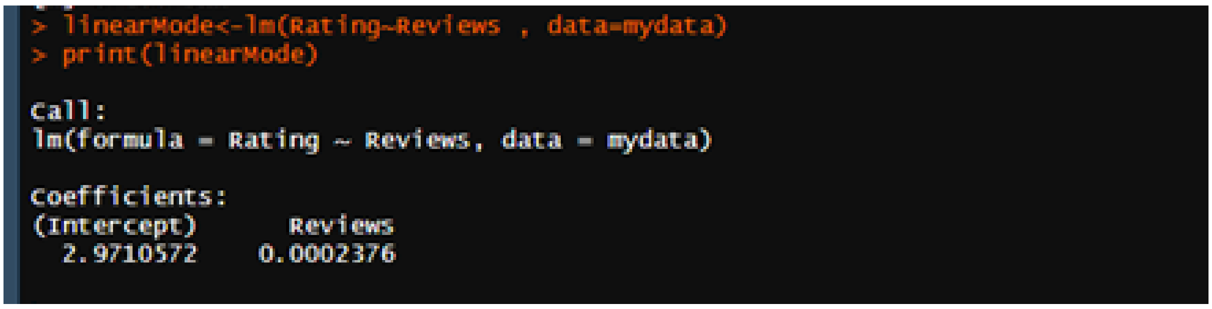


```
> scatter.smooth(x=mydata$Reviews, y=mydata$Rating,ylim = range(1,7), main="Rating ~ Reviews")
> cor(mydata$Reviews,mydata$Rating)
[1] 0.2867911
```

Fig : 7.1 Code for co-relation

From this we conclude that there is positive co-relation between the two variables. Though, it is not a strong one.

We now build the regression model , and the command is given below ,

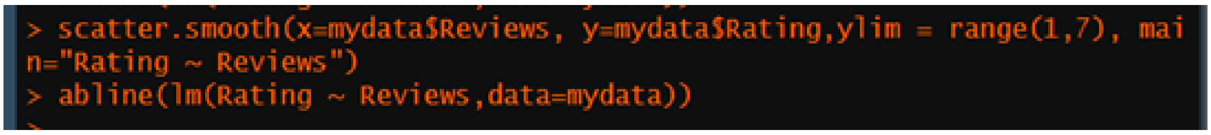


```
> linearMode<-lm(Rating~Reviews , data=mydata)
> print(linearMode)

Call:
lm(formula = Rating ~ Reviews, data = mydata)

Coefficients:
(Intercept)      Reviews  
  2.9710572    0.0002376  
```

Fig:7.2 Code for building the linear model

- By building the linear regression model, we have established the relationship between the predictor and response in the form of a mathematical formula.
- The relationship can be given by ,
- ***Rating = 2.971 + 0.0002376*review***

Now we build the regression plot for our model , the code is also given alongside .



```
> scatter.smooth(x=mydata$Reviews, y=mydata$Rating,ylim = range(1,7), mai
n="Rating ~ Reviews")
> abline(lm(Rating ~ Reviews,data=mydata))
```

Fig: 7.3 Code for plotting the linear regression model

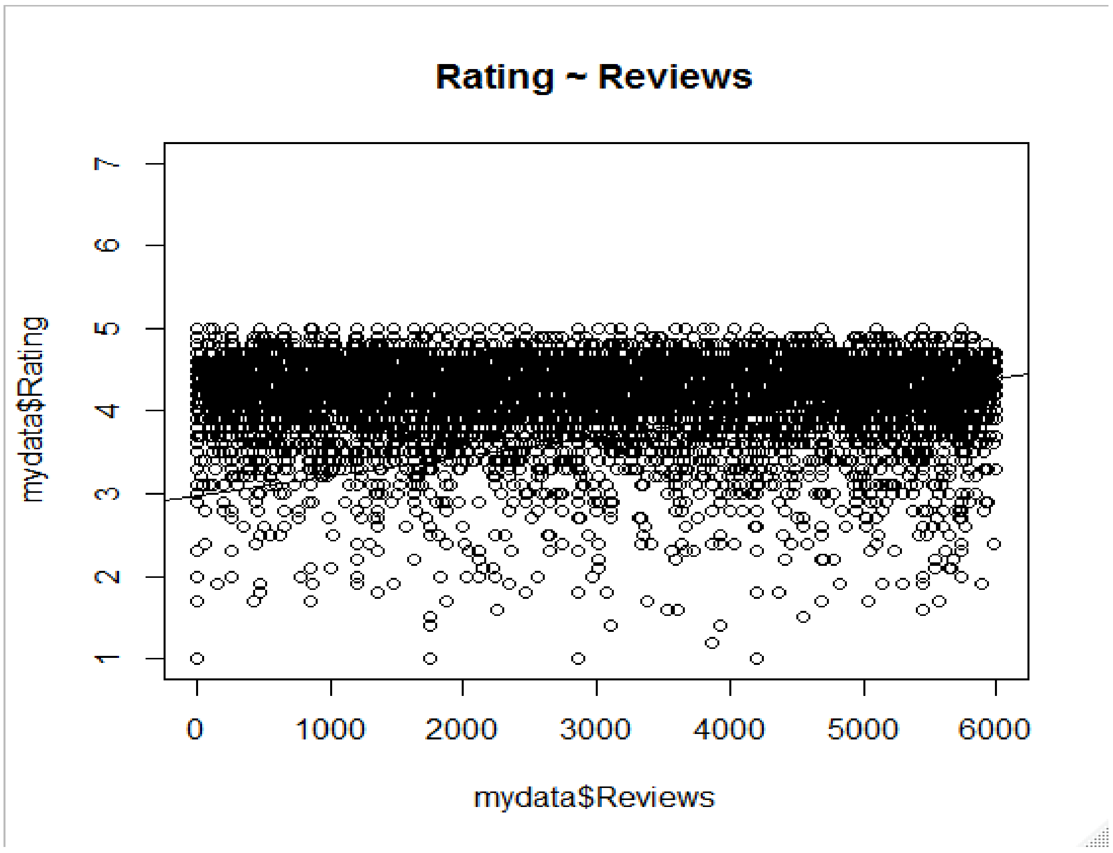
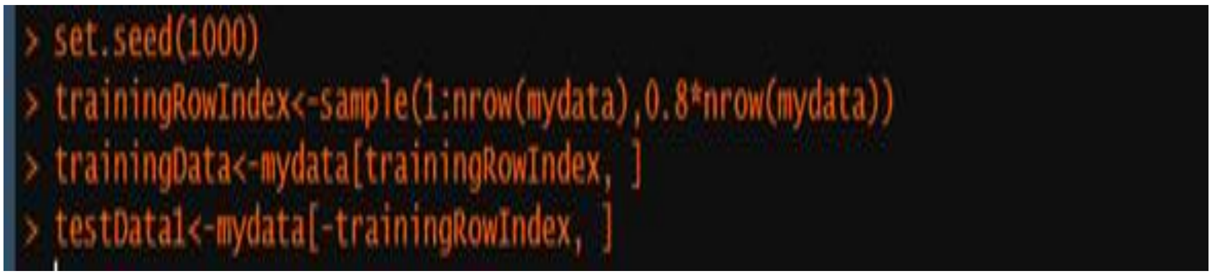


Fig:7.4 Graph showing the linear plot.

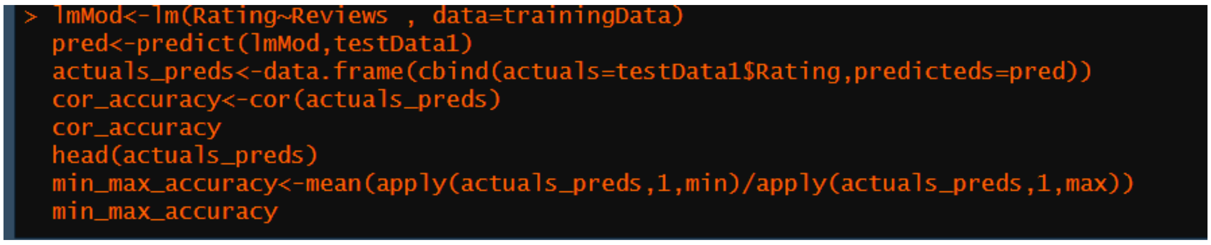
We'll now build a predictive model , The preferred practice to predicting is to split your dataset into a 80:20 sample (training : test), then, build the model on the 80% sample and then use the model thus built to predict the dependent variable on test data(the left out 20% data). The code is given below ,



```
> set.seed(1000)
> trainingRowIndex<-sample(1:nrow(mydata),0.8*nrow(mydata))
> trainingData<-mydata[trainingRowIndex, ]
> testData1<-mydata[-trainingRowIndex, ]
```

Fig:7.5 code for training a model

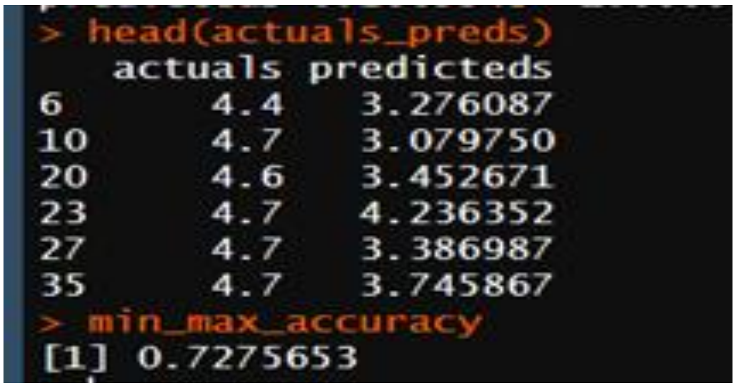
Now, using this model we'll predict the accuracy of our model , and the code for that is given below ,



```
> lmMod<-lm(Rating~Reviews , data=trainingData)
  pred<-predict(lmMod,testData1)
  actuals_preds<-data.frame(cbind(actuals=testData1$Rating,predicteds=pred))
  cor_accuracy<-cor(actuals_preds)
  cor_accuracy
  head(actuals_preds)
  min_max_accuracy<-mean(apply(actuals_preds,1,min)/apply(actuals_preds,1,max))
  min_max_accuracy
```

Fig: 7.6 Code for predicting the accuracy.

Here , we have used min-max accuracy technique for prediction , the result of this prediction is as follows ,



```
> head(actuals_preds)
   actuals predicteds
6      4.4   3.276087
10     4.7   3.079750
20     4.6   3.452671
23     4.7   4.236352
27     4.7   3.386987
35     4.7   3.745867
> min_max_accuracy
[1] 0.7275653
```

Fig: 7.7 Showing the accuracy

Now we can conclude that our model has an accuracy of 72.75% . Along with that even the predicted values are also shown alongside. So, the predictive model has been built.

8. Hypothesis Testing

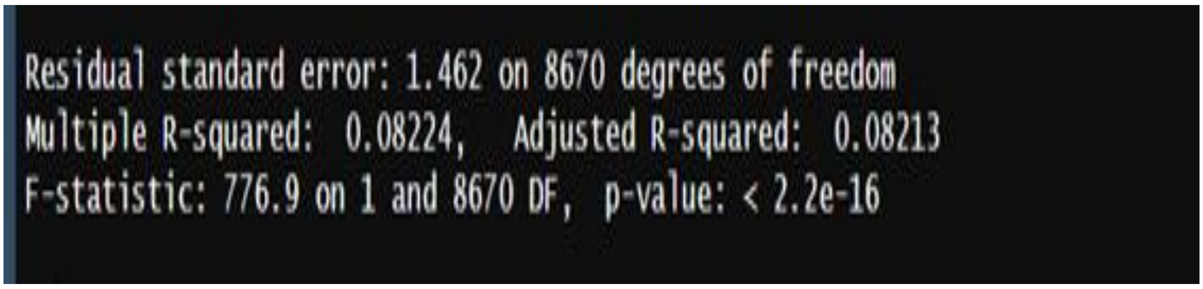
Finally, we performed a hypothesis test on the model. Hypothesis testing is a statistical method that is used in making statistical decisions using experimental data.

For our problem statement, the null hypothesis is,

Null Hypothesis: The rating of the app does not depend on the number of reviews.

Alternate Hypothesis: The rating of the app depends on the number of reviews.

The p_value obtained here is less than 0.05 so we can conclude by rejecting the null hypothesis and thereby claiming that there is a relationship between the app ratings and the reviews.



```
Residual standard error: 1.462 on 8670 degrees of freedom
Multiple R-squared:  0.08224,	Adjusted R-squared:  0.08213
F-statistic: 776.9 on 1 and 8670 DF,  p-value: < 2.2e-16
```

Fig:8.1 It gives us the p_value.

9.Conclusion

As seen above we can conclude by saying that with the increase in the number of reviews the rating of the app becomes higher. Even our hypothesis testing supports the claim. Similarly we can do this testing for various other variables and find a relation between them. Through the data visualisation we can conclude that , Apps related to Family are the most frequently downloaded category followed by apps related to games and Family Apps have the highest income as it has the most number of installs. We can now finally conclude by saying that every factor in our dataset is equally important and has the capability to make an impact on the values of other factors. These predictions and visualization can help the app developers in developing and promoting their apps to get the desired result.

10. Future Scope

Since this was the first experience we had with R programming language, we decided to begin by taking up a simple model to show the relation between the different parameters of our dataset. Our main idea was to show a relationship between the different parameters of our dataset and the predictor variable . Although we did manage to show a desired relationship between the different values, there is still scope to improve and build on this current idea. This analysis when done with a proper dataset , can greatly help the app developers in building their apps and marketing it in a right way . This was a greatly beneficial project through which we learned a lot of new things and are sure that we will use all these concepts in building more efficient project in the future.