Google Play Store Data Analysis Report

Suheng Yao

2024-11-18

Abstract

The main purpose of this report is about prediction of rating scores of the apps in google play store and find out the important variables that can influence the rating score. The method used are multiple linear regression model and linear mixed effect model. After testing for the model results, the multiple linear regression gets the better performance, and the important variables are number of reviews, price, number of installs, size, category and the time the app was last updated. In the discussion section, the report had a literature review, comparing the approach used in this report with methods used by other researchers. Also, some biases within the dataset used is mentioned in the discussion section.

Introduction

Google Play Store has always been the most popular and largest app store for Android phone users across the world. Since it is pre-installed on supporting Android devices, and the operative system holds over 70 percent of the global market, Google Play Store becomes the default app hub for most of the Android users worldwide. Up until the second quarter of 2024, there are 2.26 million apps available in the store[1].

In this project, the main question of interest is: What are the important factors that can affect the apps ratings? The reasons why I am interested in this question are divided into two part: first of all, since I use Google Play to install apps every day, I am always curious about what kind of apps can have good ratings scores and why some apps I find easy to use get low rating score. Secondly, by doing this analysis, if I want to build apps to publish on Google Play in the future, then I will know which essential factors to focus on or the popular fields to go into.

This project report will be divided into four parts: data cleaning section will talk about data cleaning process; EDA section will include the graph of variables of interest and initial understanding of the data; methods section will talk about model selection, model building and model output results; discussion section will further analyze those findings and talk about the next steps for the analysis; appendix is the last part, which will raw R model output and references.

Data Cleaning

```
## Rows: 10,841
## Columns: 13
## $ App            <chr> "Photo Editor & Candy Camera & Grid & ScrapBook", "Colo~
## $ Category       <chr> "ART_AND_DESIGN", "ART_AND_DESIGN", "ART_AND_DESIGN", "~
## $ Rating         <dbl> 4.1, 3.9, 4.7, 4.5, 4.3, 4.4, 3.8, 4.1, 4.4, 4.7, 4.4, ~
## $ Reviews        <chr> "159", "967", "87510", "215644", "967", "167", "178", "~
## $ Size           <chr> "19M", "14M", "8.7M", "25M", "2.8M", "5.6M", "19M", "29~
## $ Installs       <chr> "10,000+", "500,000+", "5,000,000+", "50,000,000+", "10~
## $ Type           <chr> "Free", "Free", "Free", "Free", "Free", "Free", "Free",~
## $ Price          <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", ~
## $ Content.Rating <chr> "Everyone", "Everyone", "Everyone", "Teen", "Everyone",~
## $ Genres         <chr> "Art & Design", "Art & Design;Pretend Play", "Art & Des~
## $ Last.Updated   <chr> "January 7, 2018", "January 15, 2018", "August 1, 2018"~
## $ Current.Ver    <chr> "1.0.0", "2.0.0", "1.2.4", "Varies with device", "1.1",~
## $ Android.Ver    <chr> "4.0.3 and up", "4.0.3 and up", "4.0.3 and up", "4.2 an~
```

Looking at the structure of the dataset, there are in total 10,841 observations and 13 variables related to each app. The response variable is **Rating**, which is a continuous variable. By looking at the column names of the data, “Category” and “Genre” seem to have similar data, I can print out the unique values in those two columns:

```
##  [1] "ART_AND_DESIGN"      "AUTO_AND_VEHICLES"   "BEAUTY"
##  [4] "BOOKS_AND_REFERENCE" "BUSINESS"            "COMICS"
##  [7] "COMMUNICATION"       "DATING"              "EDUCATION"
## [10] "ENTERTAINMENT"       "EVENTS"              "FINANCE"
## [13] "FOOD_AND_DRINK"      "HEALTH_AND_FITNESS"  "HOUSE_AND_HOME"
## [16] "LIBRARIES_AND_DEMO"  "LIFESTYLE"           "GAME"
## [19] "FAMILY"              "MEDICAL"             "SOCIAL"
## [22] "SHOPPING"            "PHOTOGRAPHY"         "SPORTS"
## [25] "TRAVEL_AND_LOCAL"    "TOOLS"               "PERSONALIZATION"
## [28] "PRODUCTIVITY"        "PARENTING"           "WEATHER"
## [31] "VIDEO_PLAYERS"       "NEWS_AND_MAGAZINES"  "MAPS_AND_NAVIGATION"
## [34] "1.9"
```

```
##  [1] "Art & Design"                           "Art & Design;Pretend Play"
##  [3] "Art & Design;Creativity"                "Art & Design;Action & Adventure"
##  [5] "Auto & Vehicles"                        "Beauty"
##  [7] "Books & Reference"                      "Business"
##  [9] "Comics"                                 "Comics;Creativity"
## [11] "Communication"                          "Dating"
## [13] "Education;Education"                    "Education"
## [15] "Education;Creativity"                   "Education;Music & Video"
## [17] "Education;Action & Adventure"           "Education;Pretend Play"
## [19] "Education;Brain Games"                  "Entertainment"
## [21] "Entertainment;Music & Video"            "Entertainment;Brain Games"
## [23] "Entertainment;Creativity"               "Events"
## [25] "Finance"                                "Food & Drink"
## [27] "Health & Fitness"                       "House & Home"
## [29] "Libraries & Demo"                       "Lifestyle"
## [31] "Lifestyle;Pretend Play"                 "Adventure;Action & Adventure"
## [33] "Arcade"                                 "Casual"
## [35] "Card"                                   "Casual;Pretend Play"
## [37] "Action"                                 "Strategy"
## [39] "Puzzle"                                 "Sports"
## [41] "Music"                                  "Word"
## [43] "Racing"                                 "Casual;Creativity"
## [45] "Casual;Action & Adventure"              "Simulation"
## [47] "Adventure"                              "Board"
## [49] "Trivia"                                 "Role Playing"
## [51] "Simulation;Education"                   "Action;Action & Adventure"
## [53] "Casual;Brain Games"                "Simulation;Action & Adventure"
## [55] "Educational;Creativity"            "Puzzle;Brain Games"
## [57] "Educational;Education"             "Card;Brain Games"
## [59] "Educational;Brain Games"           "Educational;Pretend Play"
## [61] "Entertainment;Education"           "Casual;Education"
## [63] "Music;Music & Video"               "Racing;Action & Adventure"
## [65] "Arcade;Pretend Play"               "Role Playing;Action & Adventure"
## [67] "Simulation;Pretend Play"           "Puzzle;Creativity"
## [69] "Sports;Action & Adventure"         "Educational;Action & Adventure"
## [71] "Arcade;Action & Adventure"         "Entertainment;Action & Adventure"
## [73] "Puzzle;Action & Adventure"         "Strategy;Action & Adventure"
## [75] "Music & Audio;Music & Video"       "Health & Fitness;Education"
## [77] "Adventure;Education"               "Board;Brain Games"
## [79] "Board;Action & Adventure"          "Board;Pretend Play"
## [81] "Casual;Music & Video"              "Role Playing;Pretend Play"
## [83] "Entertainment;Pretend Play"        "Video Players & Editors;Creativity"
## [85] "Card;Action & Adventure"           "Medical"
## [87] "Social"                            "Shopping"
## [89] "Photography"                       "Travel & Local"
##  [ reached getOption("max.print") -- omitted 30 entries ]
```

Based on the printed results above, "Genre" is just the more detailed classification of "Category", since in this project, I mostly focus on the general groups of apps, I can remove the "Genre" column and change the Category name to lower case.

Also, from the distinct values of category, there is a category called "1.9", which does not make sense. Since there is only one record of data with category "1.9", I can just remove this record from the dataset.

Now, let's check if there are any duplicated apps in the dataset:

```
## [1] 1181
```

From the analysis above, there are 1181 duplicated apps in the dataset, to make further analysis easier, I will just keep the first occurrence of each app record.

Now, I need to check if there are NA values in the dataset:

```
##      App              Category             Rating        Reviews
##  Length:9659        Length:9659        Min.   :1.000   Length:9659
##  Class :character   Class :character   1st Qu.:4.000   Class :character
##  Mode  :character   Mode  :character   Median :4.300   Mode  :character
##                                        Mean   :4.173
##                                        3rd Qu.:4.500
##                                        Max.   :5.000
##                                        NA's   :1463
##      Size             Installs             Type              Price
##  Length:9659        Length:9659        Length:9659        Length:9659
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  Content.Rating     Last.Updated       Current.Ver        Android.Ver
##  Length:9659        Length:9659        Length:9659        Length:9659
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
```

From the result shown above, in the response variable “Rating”, there are 1463 missing values. To maintain the size of the data, I will fill in those missing values using median values of the “Rating” column.

```
##      App              Category            Rating        Reviews
##  Length:9659       Length:9659       Min.   :1.000   Length:9659
##  Class :character   Class :character   1st Qu.:4.000   Class :character
##  Mode  :character   Mode  :character   Median :4.300   Mode  :character
##                                         Mean   :4.192
##                                         3rd Qu.:4.500
##                                         Max.   :5.000
##      Size             Installs            Type             Price
##  Length:9659       Length:9659       Length:9659       Length:9659
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Content.Rating     Last.Updated       Current.Ver        Android.Ver
##  Length:9659       Length:9659       Length:9659       Length:9659
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

Since it makes more sense to talk about installs, reviews, size and price in numerical values, I will change those variables from categorical to numeric:

```
##      App              Category            Rating         Reviews
##  Length:9659       Length:9659       Min.   :1.000   Min.   :       0
##  Class :character   Class :character   1st Qu.:4.000   1st Qu.:      25
##  Mode  :character   Mode  :character   Median :4.300   Median :     967
##                                         Mean   :4.192   Mean   :  216593
##                                         3rd Qu.:4.500   3rd Qu.:   29401
##                                         Max.   :5.000   Max.   :78158306
##
##   Size(in MB)       Installs            Type               Price
##  Min.   :  1.00   Length:9659        Length:9659        Min.   :  0.000
##  1st Qu.:  5.10   Class :character   Class :character   1st Qu.:  0.000
##  Median : 13.00   Mode  :character   Mode  :character   Median :  0.000
##  Mean   : 21.17                                         Mean   :  1.099
##  3rd Qu.: 29.00                                         3rd Qu.:  0.000
##  Max.   :100.00                                         Max.   :400.000
##  NA's   :1541
##  Content.Rating     Last.Updated       Current.Ver        Android.Ver
##  Length:9659       Length:9659       Length:9659       Length:9659
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
```

Since Most of the NA values in Size are related to “Varies with Device” value in the original dataset, and in this project, I want to mostly focus on the app with fixed size, I will just remove those app records with varied app sizes.

Transforming Installs is a more complex matter, I will solve this problem differently. First check the distinct values in the variable “Installs”:

```
##  [1] "10,000+"        "500,000+"       "5,000,000+"     "50,000,000+"
##  [5] "100,000+"       "50,000+"        "1,000,000+"     "10,000,000+"
##  [9] "5,000+"         "100,000,000+"   "1,000+"         "500,000,000+"
## [13] "50+"            "100+"           "500+"           "10+"
## [17] "1+"             "5+"             "1,000,000,000+" "0+"
```

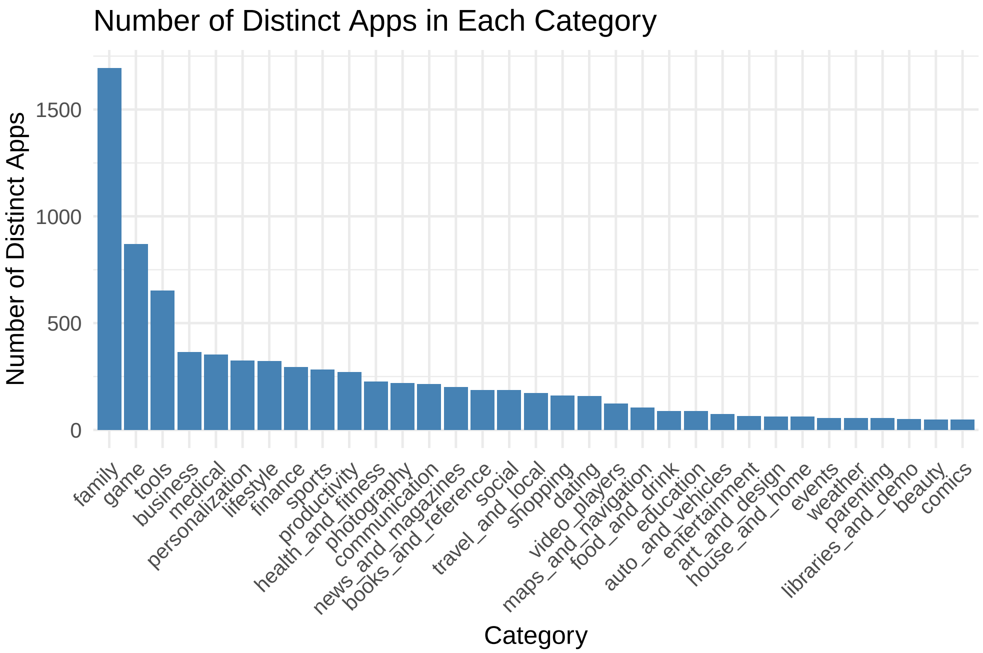
To convert those values into numeric values and avoid duplicate values, take “500+” as an example, I will change it to a value between 500 to 1000 because “1000+” will be on a different level.

```
##      App              Category             Rating         Reviews
##  Length:8118        Length:8118        Min.   :1.000   Min.   :       0
##  Class :character   Class :character   1st Qu.:4.000   1st Qu.:      17
##  Mode  :character   Mode  :character   Median :4.300   Median :     534
##                                        Mean   :4.189   Mean   :  125135
##                                        3rd Qu.:4.500   3rd Qu.:   17068
##                                        Max.   :5.000   Max.   :44891723
##   Size(in MB)       Installs              Type               Price
##  Min.   :  1.00   Min.   :0.000e+00   Length:8118        Min.   :  0.000
##  1st Qu.:  5.10   1st Qu.:2.839e+03   Class :character   1st Qu.:  0.000
##  Median : 13.00   Median :7.671e+04   Mode  :character   Median :  0.000
##  Mean   : 21.17   Mean   :8.849e+06                      Mean   :  1.195
##  3rd Qu.: 29.00   3rd Qu.:2.305e+06                      3rd Qu.:  0.000
##  Max.   :100.00   Max.   :1.307e+09                      Max.   :400.000
##  Content.Rating     Last.Updated       Current.Ver        Android.Ver
##  Length:8118        Length:8118        Length:8118        Length:8118
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

After those data cleaning is done, I will start doing EDA.

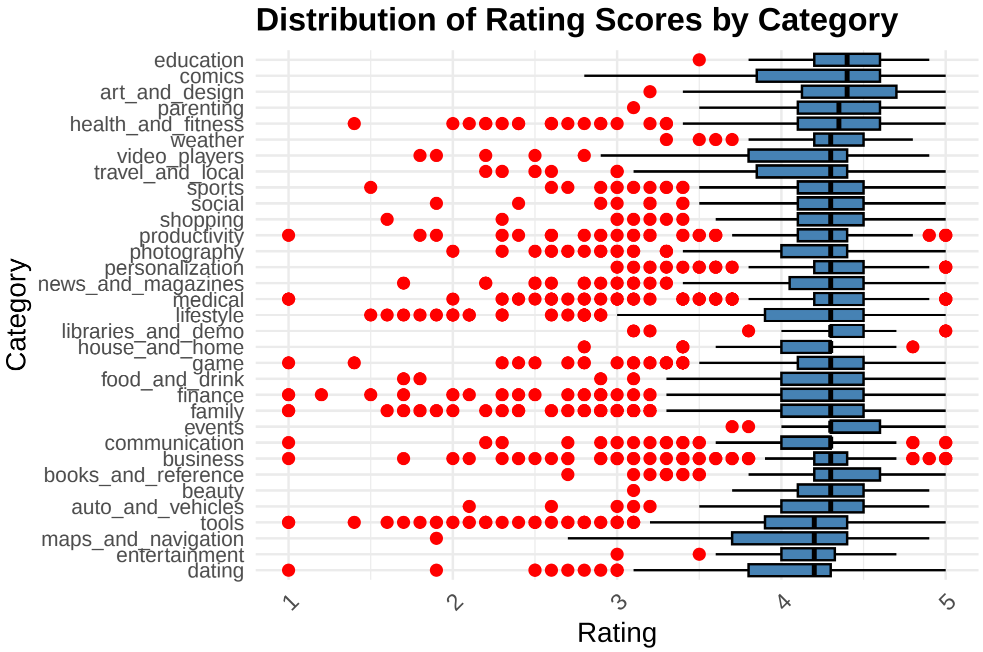
EDA

Step 1: Bar Plot Counting Distinct Apps in Each Category



From the count of distinct apps, it is clear that category family gets the most number of apps, with more than 1500 apps in the category. The second most category is game, which is only half the number of the family category, and the third most is the tools category. The category with the least number of distinct apps is comics, with only less than 250 apps.

Step 2: Box Plot of Distribution of Rating Scores for Each Category

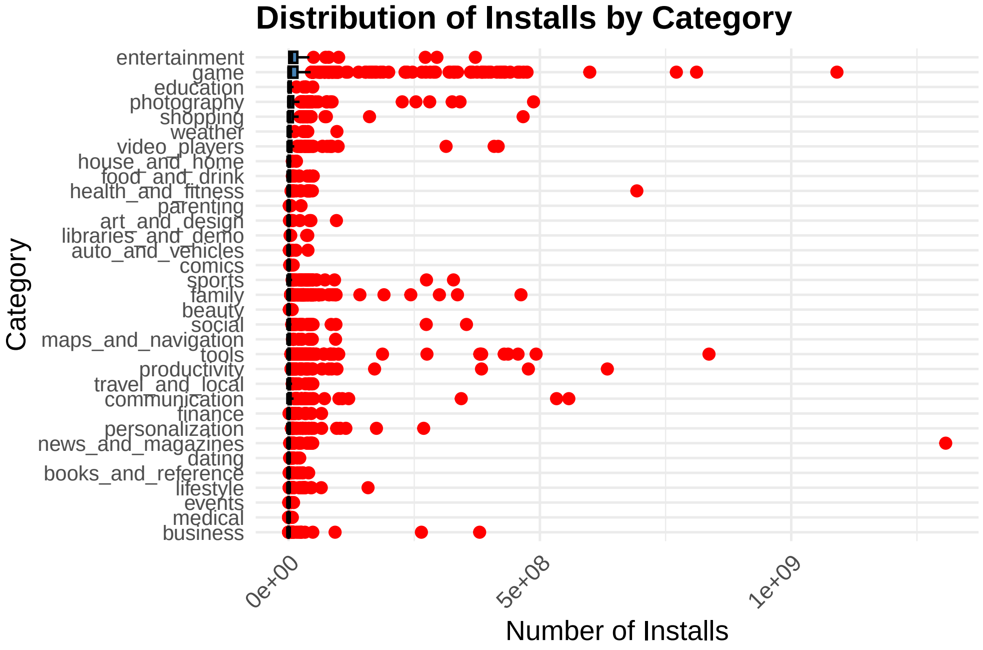


From the box plot for each category above, it is clear that almost all the categories have median value greater than 4. Also, maybe due to less number of apps in comics category, it is the only category without outliers. What's more, the last four categories at bottom: Tools, Maps and Navigation, Entertainment and Dating tend to have lower median values compared to other groups, which may reflect the user dissatisfaction for those app groups.

Table 1: Summary of Installations by Category

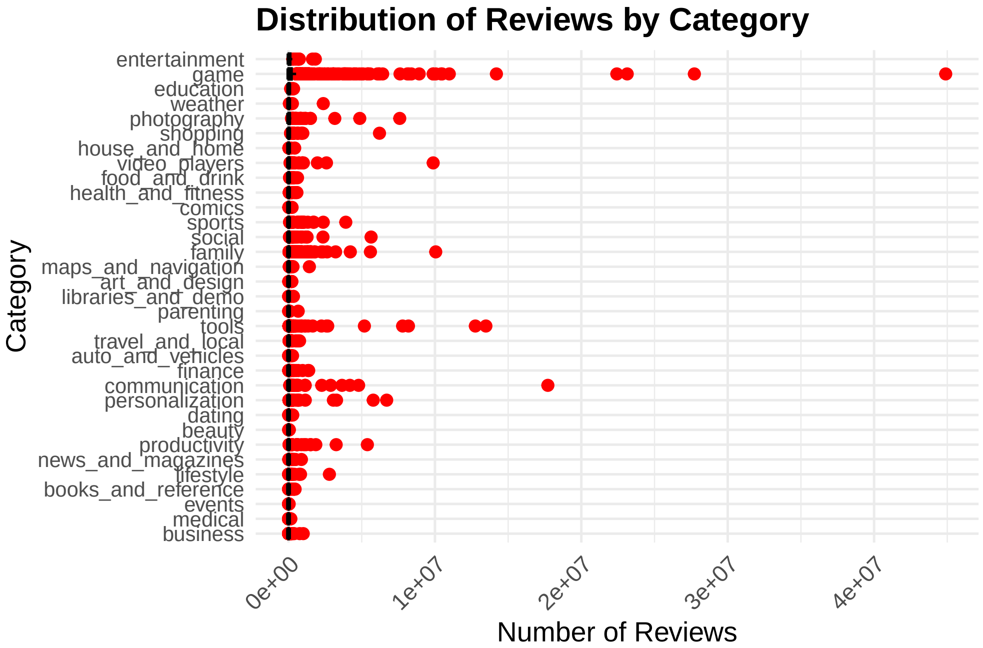
Category	Count	Min Installs	Median Installs	Max Installs
family	1694	0	71920	462319688
game	870	3	2196416	1090875958
tools	651	4	52464	836361665
business	365	0	990	380087219
medical	353	0	3104	7542762
personalization	325	0	30609	268748267

Step 3: Table and Box Plot of Distribution of Number of Installs in Each Category



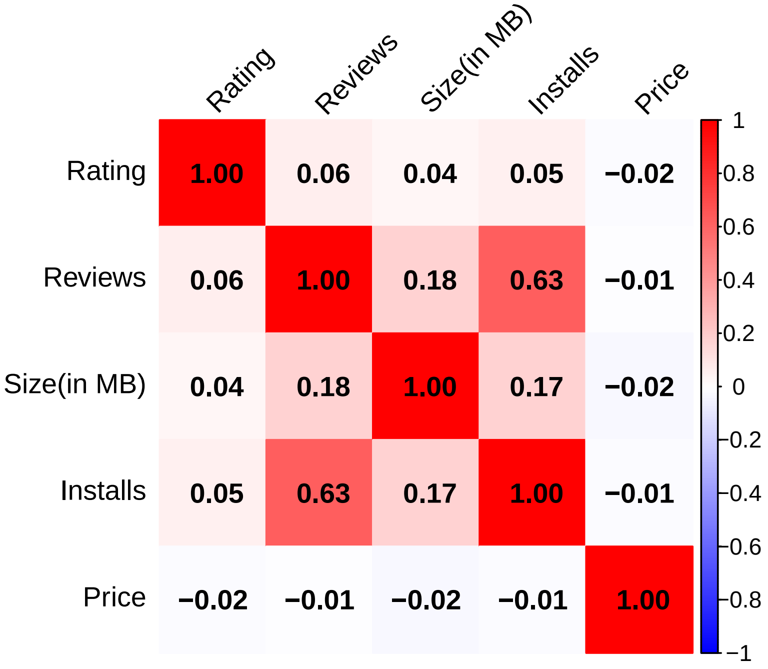
From this boxplot above, in general, games tend to have the more number of installs, and the range of app installs in the game category also tend to be greater than other categories. However, there is one outlier in personalization category, which has the most number of installs across all categories.

Step 4: Distribution of Reviews in Each Category



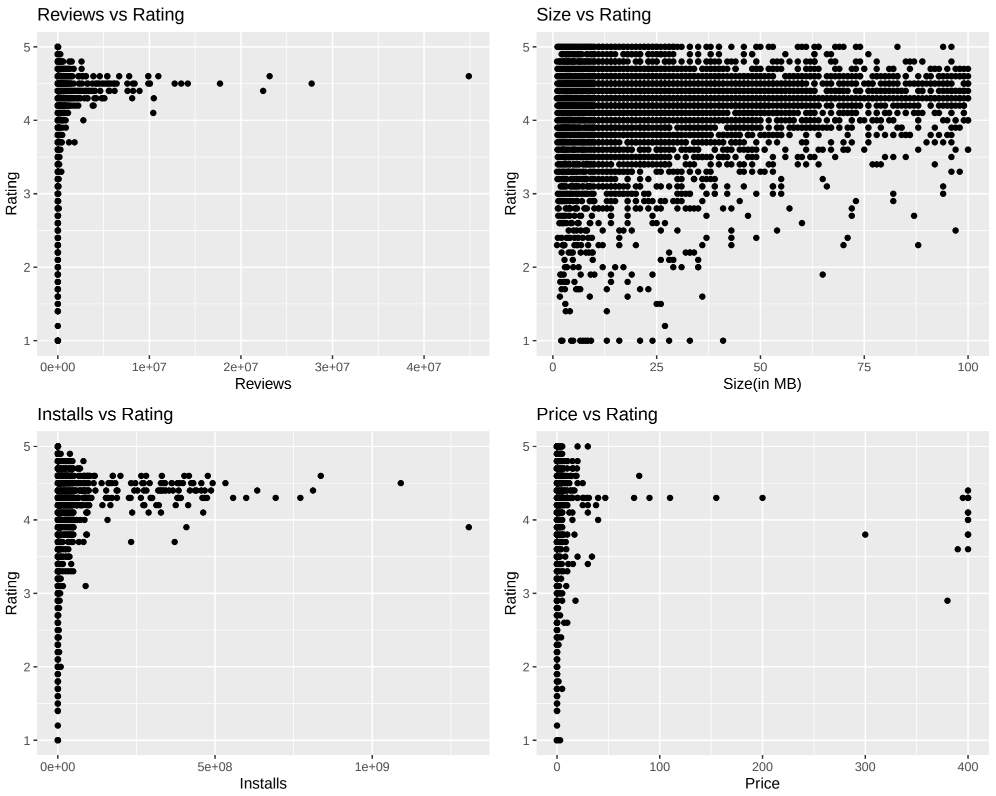
As shown in the box plot above, game category still tend to have more reviews than the other category, with the most number of reviews over $4 * 10^7$.

Step 5: Correlation Analysis of Numeric Variables



From the correlation plot, most of the numerical variables do not have strong correlation with each other, except the installs and review. Their correlation is 0.64, which means that there is positive relationship between those two variables. When there is more installs for the app, it tends to have more reviews.

We can also plot the relationship of those numeric variables with Rating to assess their relationships:



Based on the scatterplot above, all the numeric variables show a similar log relationship with the response variable Rating, indicating that log transformation may be needed to transform those variables.

Step 6: Check Association Between Categorical Variables Using Chi-Square Test

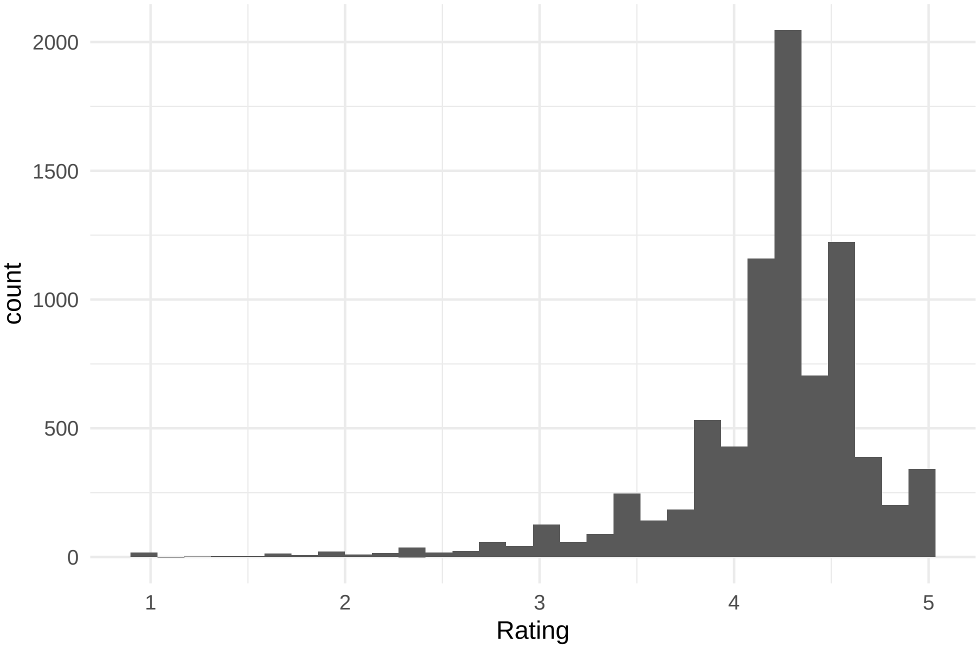
```
##
##  Pearson's Chi-squared test
##
## data:  Category_vs_Type
## X-squared = 246.46, df = 32, p-value < 2.2e-16

##
##  Pearson's Chi-squared test
##
## data:  Category_vs_ContentRating
## X-squared = 4377.1, df = 160, p-value < 2.2e-16

##
##  Pearson's Chi-squared test
##
## data:  Type_vs_ContentRating
## X-squared = 14.532, df = 5, p-value = 0.01256
```

From the chi-square results above, since all the p-value of the tests are less than 0.05, which means that there is association between category and type, category and content rating, type and content rating.

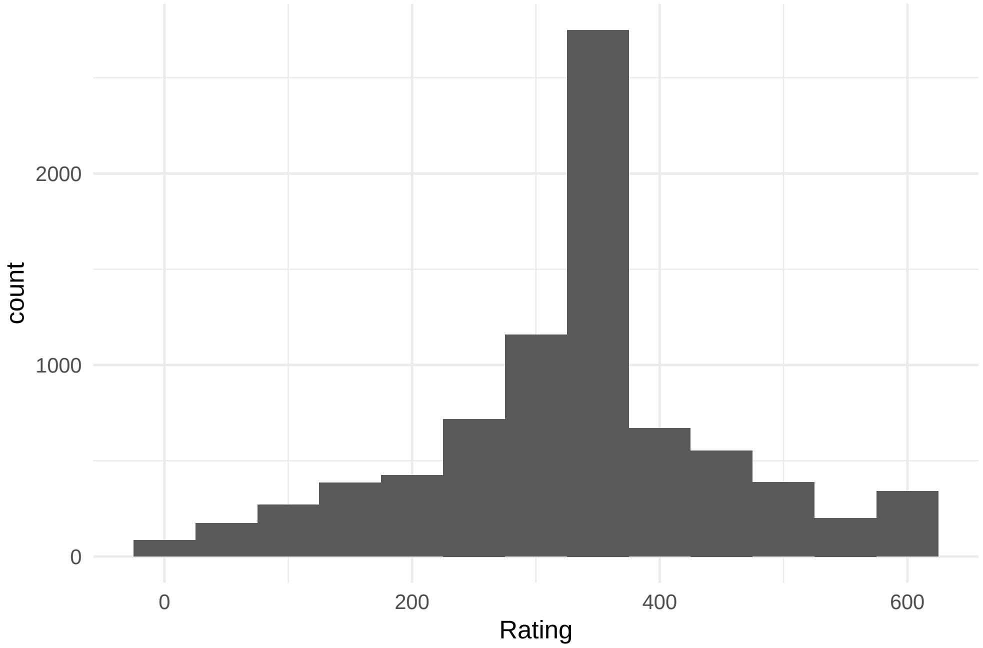
Method

Step 1: Selection of Models

Since the response variable rating is a continuous variable, logistic and multinomial models cannot be used. Also, the response variable is not count data and not discrete, so poisson and negative binomial models cannot be used. Thus, the models that I want to use are linear regression model or linear mixed effect models. First plot the distribution of rating score:



Since the rating score is left-skewed, I want to do some transformation to make it more symmetric. The method I would like to use is power transformation.



After taking rating to the power of 4, the distribution of rating score becomes symmetric. So in the later report, when I refer to the variable rating, it refers to $rating^4$.

Step 2: Start from Null Model

The formula of **null model** is:

$$Rating = \beta_0 \tag{0}$$

The model output is in the Null Model section in Appendix.

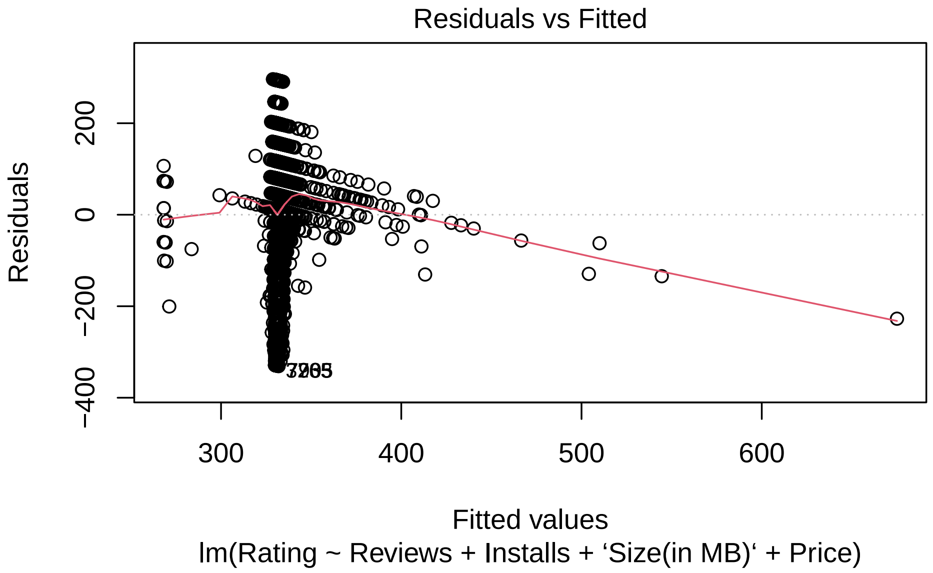
In this null model, the estimate of the intercept is just the overall mean of rating score of all the apps in the whole dataset.

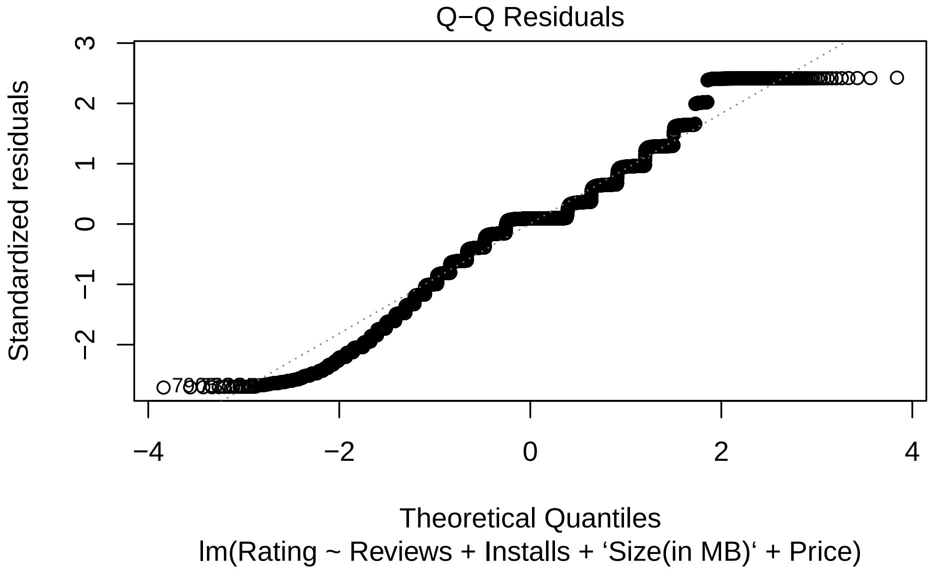
Step 3: Building the Linear Regression Model and Transformation of Data

Building upon the null model, based on the correlation matrix and scatterplot in EDA, I will first add reviews, installs, size and price in its original form into the model:

The formula of **model 1** is:

$$Rating = \beta_0 + \beta_1 \cdot Reviews + \beta_2 \cdot Installs + \beta_3 \cdot Size + \beta_4 \cdot Price \tag{1}$$

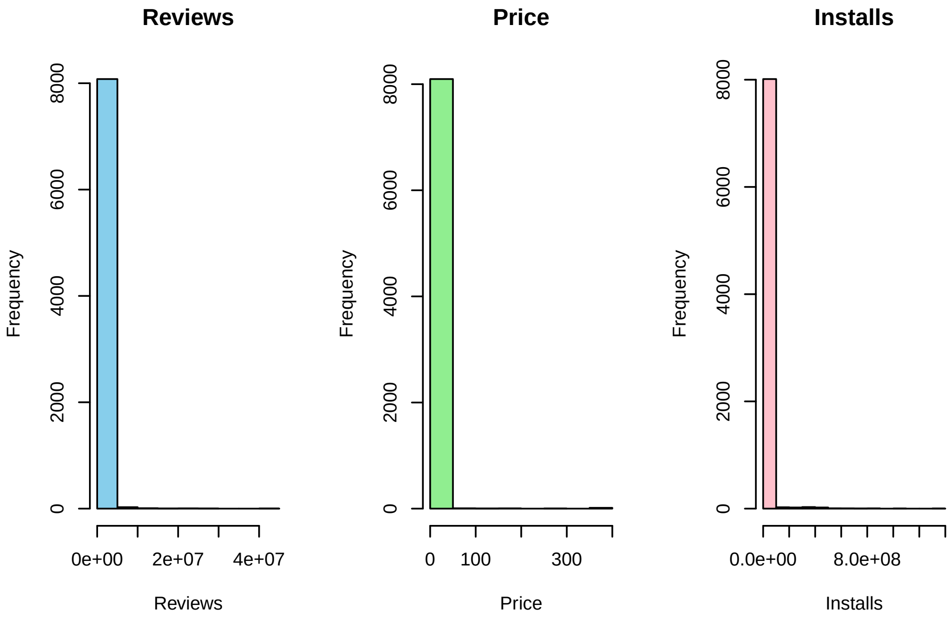




The model output is in the Model 1 section in Appendix.

Based on the model output result, variable **Reviews and Price** have p-value less than 0.05, suggesting that they have statistically significant effect on Rating. Additionally, F-test is less than 0.05, suggesting that there is at least one variable that is statistically significant. However, based on the residuals vs fitted value plot, the points are not randomly scattered around 0, indicating that there is heteroscedasticity problem. Based on the QQ plot, there are some points that deviate from the line, but overall, the normality assumption is not violated. Since the homoscedasticity assumptions of the linear regression models is violated, the model cannot used to fit the dataset, which may explain the low adjusted R-square value.

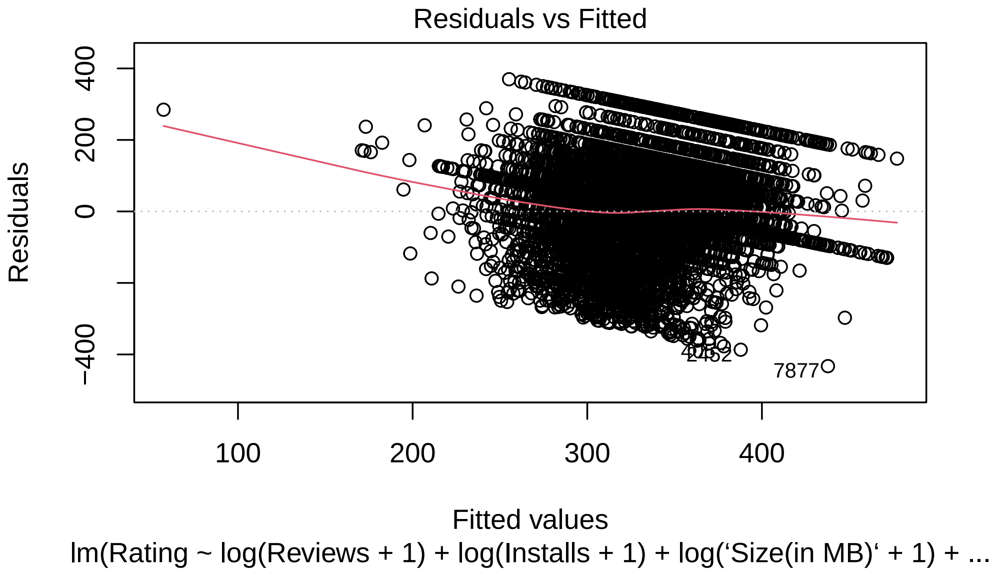
As mentioned in EDA part, one thing I would like to try is doing log transformation on the predictor variable, and based on the residual vs fitted plot, the fitted value is concentrated in a narrow range, and this could be related to the high-skewness or low variability in one of the predictor variables:

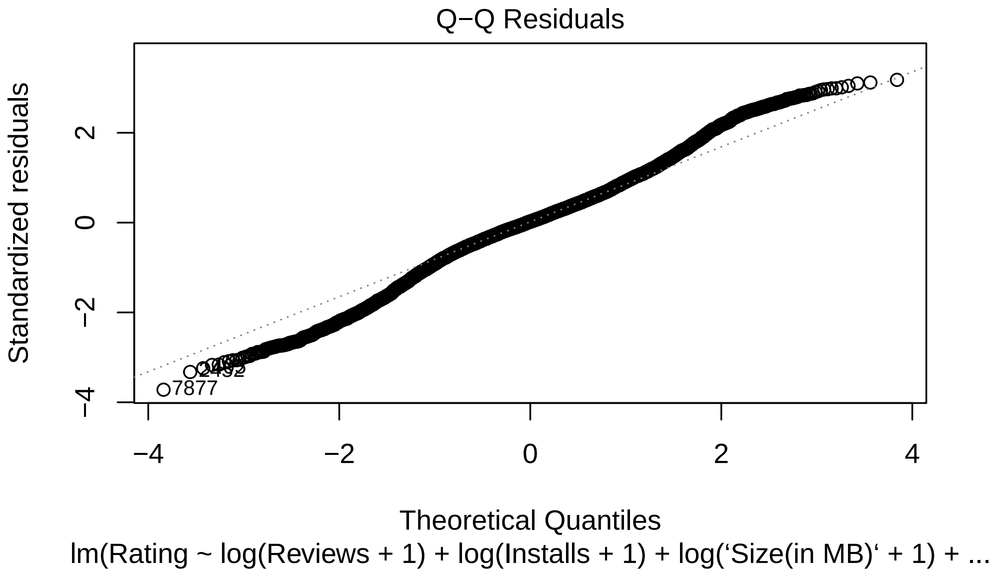


Based on the histograms above, those three numeric variables are all highly skewed, so I could do log transformation on those three variables.

Here is the formula for the **model 2**, model output in Model 2 section in Appendix, the reason to add 1 in log is to avoid some zero value for those variables because log is not defined at 0:

$$\begin{aligned} Rating = \beta_0 + \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ \beta_3 \cdot log(Size + 1) + \beta_4 \cdot log(Price + 1) \end{aligned} \tag{2}$$





After doing the log transformation on predictors, in the residual vs fitted plot, the residuals become more spread around 0, indicating the homoscedasticity assumption is no longer violated, and the QQ plot has less heavy-tail problem, indicating the normality of residual distribution is also no longer. What's more, the adjust r-squared value is ten times the original model, and the residual standard error also decreased, meaning that model 2 is a better model than model 1. I could add more categorical variables into the model and see if it improves the model. Since **log(Size)** is no longer significant in this model, I can consider removing it for the later model.

Here is the formula for **model 3**, model output is in the Model 3 section in Appendix:

$$\begin{aligned} Rating = \beta_0 + \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ \beta_3 \cdot TypePaid + \beta_4 \cdot log(Price + 1) + \\ \beta_5 \cdot Category \end{aligned} \tag{3}$$

In the formula above, because there are too many levels in Category variable, it is written as one variable with one coefficient, but in the actual model, it is treated as 32 different levels(1 level is reference) with 32 coefficients.

In model 3, residual standard error decreased and adjusted R-square increased, meaning that adding category variable increase the model performance. Let's try adding two more categorical variables.

Here is the formula for **model 4**, model output is in the Model 4 section in Appendix:

$$\begin{aligned} Rating = \beta_0 + \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ \beta_3 \cdot TypePaid + \beta_4 \cdot log(Price + 1) + \\ \beta_5 \cdot Category + \beta_6 \cdot Content.Rating + \\ \beta_7 \cdot Last.Updated \end{aligned} \tag{4}$$

Similar to model 3, in the actual model, each level of categorical variable get a coefficient, for the sake of easy writing, they are each treated as one single variable in the formula.

Based on the model4 output result, the residual standard error and adjusted r-squared all get further improvement.

After getting those five models with only fixed effects, I can use MSE to determine which model is the best model and compare it with the null model:

Table 2: Mean Squared Error (MSE) for Models

Model Name	MSE Value
Null Model	14959.46
Model 1	14895.05
Model 2	13542.87
Model 3	13181.08
Model 4	10767.30

```
## Analysis of Variance Table
##
## Model 1: Rating ~ 1
## Model 2: Rating ~ log(Reviews + 1) + log(Installs + 1) + Type + log(Price +
##     1) + Category + Content.Rating + Last.Updated
##   Res.Df       RSS   Df Sum of Sq  Pr(>Chi)
## 1   8117 121440886
## 2   6849  87408925 1268  34031961 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By comparing their MSE and chi-square test between model 4 and null model, model 4 is the best model, which has the lowest MSE, residual standard error and highest adjusted R-squared.

Step 4: Fit the No Pooling, Partial Pooling and Complete Pooling Model

Based on the model fitting result above, some of the categories of apps are statistically significant, also, in the EDA part, each group's variability in rating score is different. Thus, I could try to treat Category variable as a group variable and fit a LMM on the data. The model output for all three models can be found in the appendix section.

Let's start with the **no pooling model**, which is similar to model 4, here is the formula:

$$\begin{aligned} Rating = {} & \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ & \beta_3 \cdot TypePaid + \beta_4 \cdot log(Price + 1) + \\ & \beta_5 \cdot Category + \beta_6 \cdot Content.Rating + \\ & \beta_7 \cdot Last.Updated \end{aligned} \tag{5}$$

In this no pooling model, each category is treated as a separate variable in the model and has its own coefficients. Also, the adjusted R-squared increased a lot compared to model 4.

Now let's fit the **partial pooling model**, here is the formula:

$$\begin{aligned} Rating = {} & \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ & \beta_3 \cdot TypePaid + \beta_4 \cdot log(Price + 1) + \beta_5 \cdot Content.Rating + \\ & \beta_6 \cdot Last.Updated + (1|Category) \end{aligned} \tag{6}$$

Based on the output of the partial pooling model, the variance of app rating score of each Category group is very small compared to overall variance of app rating score across all Category groups. Also, compared with

Table 3: Mean Squared Error (MSE) for Models

Model Name	MSE Value
No Pooling Model	10767.30
Partial Pooling Model	10780.38
Complete Pooling Model	11064.24

no pooling model, the residual standard deviation is similar compared with no pooling model, indicating that the random effect may not be necessary.

Another option is to try **complete pooling** and completely ignore the Category Variable, here is the formula:

$$\begin{aligned} Rating = {} & \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ & \beta_3 \cdot TypePaid + \beta_4 \cdot log(Price + 1) + \beta_5 \cdot Content.Rating + \\ & \beta_6 \cdot Last.Updated \end{aligned} \tag{7}$$

In this model, compared with previous two models, the residual standard error increased, and its adjusted R-squared is lower than the no pooling model, indicating that the model may be worse than the previous two.

To further compare those three model, MSE could be calculated. Based on the MSE table above, the model with no pooling gets the lowest MSE, which means that it is the best model among the three models.

Table 4: Comparison of Model 4 and No Pooling Model

Model Name	AIC Value	MSE Value
No Pooling Model	100947.6	10767.3
Model 4	100947.6	10767.3

Step 5: Compare the No Pooling Model and Model 4

Based on the results above, no pooling model gets very similar result with model 4 for both AIC and MSE because no pooling model is just model 4 removing the intercept term. For easier model interpretation, I would choose model 4 for further analysis, but with high AIC value and lower MSE value, this could indicate the overfitting problem with model 4.

Step 6: Add Interaction Term into Model 4

Since based on the model 4 output, **Type and log(Reviews)** are statistically significant, I could try adding the interaction term between these two, here is the formula for the model, the model output can be found in the appendix section:

$$\begin{aligned}Rating = \beta_1 \cdot log(Reviews + 1) + \beta_2 \cdot log(Installs + 1) + \\ \beta_3 \cdot TypePaid + \beta_4 \cdot log(Price + 1) + \beta_5 \cdot Content.Rating + \\ \beta_6 \cdot Last.Updated + \beta_7 \cdot Category + \beta_8 \cdot log(Reviews + 1) \cdot TypePaid\end{aligned} \tag{8}$$

```
## Analysis of Variance Table
##
## Model 1: Rating ~ log(Installs + 1) + log(Price + 1) + Type * log(Reviews +
##     1) + Content.Rating + Last.Updated
## Model 2: Rating ~ log(Reviews + 1) + log(Installs + 1) + Type + log(Price +
##     1) + Category + Content.Rating + Last.Updated
##   Res.Df      RSS Df Sum of Sq  Pr(>Chi)
## 1   6880 89752403
## 2   6849 87408925 31   2343479 < 2.2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

Based on the anova table result above, the chi-square test shows that model 2, which is the model without interaction term, is better. Thus, the additional interaction term between log(Reviews+1) and Type is not needed.

Step 7: Interpretation of Model 4 Selected

Now, after choosing the best model, I will try to interpret this model:

Based on the model output, the residual’s median is 0, and min, max, first quantile and third quantile are all close to each other, meaning that the residual’s distribution is symmetric.

To interpret the coefficient, since there are many variables, especially in category, content rating and last updated, I will only select one of the level from each variable to interpret.

For the coefficient of reviews, one unit increase in $log(Reviews + 1)$, the ratings(which is the 4th power of original rating score) is expected to increase by 30.91, keeping all other variables constant. This shows the positive correlation between reviews and rating score.

For the coefficient of installs, one unit increase in $log(Installs + 1)$, the ratings is expected to decrease by 30.2, keeping all other variables constant. This shows the negative correlation between installs and rating score.

The interpretation of coefficient of log(price) is similar to previous ones, and it also has negative correlation with the rating scores.

For coefficient of Type, the average rating score for paid app is 20.21 higher compared to free app, keeping all other variables constant.

For category level auto and vehicles, the coefficient is -58.6, which shows that the average rating score for apps belonging to auto and vehicles group is 58.6 lower compared to the apps in arts and design group(reference group), keeping all other variables constant.

For content rating level everyone, the coefficient is -42.1, which shows that the average rating score for apps with content rating as everyone is 42.1 lower compared to the apps with content rating as Adults(reference group), keeping all other variables constant.

For the last updated date April 12th, 2016, the coefficient is 204, which shows that the average rating score for apps that last updated at April 12th, 2016 is 204 higher compared to the apps last updated at April 1st 2016(reference group), keeping all other variables constant.

The residual standard error is 112.9, which is the lowest compared to other models that I fit. The adjust R-square is 0.1482, which indicated that the model can only explain 14.82% variability in the response variable, and although this is the best model I get, the model still does not fit the data well. The F-statistic has p-value less than 0.05, showing that there is at least one variable that is statistically significant, and the model is better than the null model.

The important predictors of rating score according to my model are reviews, installs, type, price, category and last updated date.

Discussion

To validate the result of my model, I found three research paper that related to rating score prediction in Google Play Store.

Based on the paper by S Shashank et.al[2], they tried to predict the rating score using machine learning algorithm. The paper used the same dataset as I do in this report. Compared to my approach, the author do a more detailed EDA. Instead of focusing on the category of apps, they pay more attention on whether the app is free or paid and find out the difference in rating score between free app group and paid app group. In the method part, the authors applied five techniques trying to find the important variables related to rating score: random forest, support vector regression, linear regression, k-nearest neighbors and k-means clustering. As a results, the k-means neighbors achieves the best result, which has 92% prediction accuracy, and the author concludes that **size, type, price, content rating and genre** are variables strongly correlated with the rating score. The author's result is consistent with my result, and those are also statistically significant variables in my linear regression model, except I used category instead of genre and exclude size from my model. However, after reading the paper, I find out there could be bias related to the rating score because higher ratings given by users potentially attract several new users disproportionately, and people tend to only use apps with high rating score, leading to more reviews of the app. Additionally, many people don't like writing reviews for the apps no matter they like the app or not, there are also people writing negative reviews but give very positive rating scores, so for some apps, the rating score may not reflect their true quality.

Another research paper I found was written by Min-Kyo Seo et.al[3]. The main purpose of this paper is to investigate the predictors and main determinants of consumers' ratings of mobile applications in the Google Play Store. The author also tried to extend their model into a sentimental analysis and aim to review polarity and subjectivity on the application rating. In the data preprocessing part, they used sentiment analysis based on the users reviews from Google Play Store and created new variables polarity and subjectivity and merge them into the original Google Play dataset. In the method part, there were four models they used: multiple linear regression, regression tree, random forest tree and neural network. Based on the model result, neural network model gives the lowest RMSE result, and the important variables are **price, installs and reviews**, and polarity and subjectivity of reviews are less critical. Those variables are also included in my model, one difference is that install is positively correlated with ratings based on the author's results, but install is negative correlated with rating score based on my model's results.

The last research paper I found is by Jayanth. P et.al[4], and the main purpose of research paper is to predict the rating score using the comprehensive Google Play Store dataset similar to the one that I used. The method that the author used are: lasso regression model, ridge regression model, gradient boosting, XGBoost and CATBoost. Based on the result, CATBoost method provides the lowest MAE, MSE, RMSE and highest R^2 value, but the top features that the author find that can influence the rating score is different from my results, the author find that **reviews, last updated and android version** are the most critical variables. In my models, android version is not included, and most of the levels of last updated are not statistically significant.

Based on the literature review above, the next step for my analysis could be try to implement tree model or deep learning model to further improve my current linear regression model.

Appendix

Model Output

Null Model

```
##
## Call:
## lm(formula = Rating ~ 1, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -330.77  -75.77   10.11   78.29  293.23
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  331.772      1.358   244.4   <2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 122.3 on 8117 degrees of freedom
```

Model 1

```
##
## Call:
## lm(formula = Rating ~ Reviews + Installs + ‘Size(in MB)‘ + Price,
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -330.94  -74.11   11.12   75.92  296.21
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.300e+02  1.892e+00 174.435  < 2e-16 ***
## Reviews        7.579e-06  1.835e-06   4.131 3.65e-05 ***
## Installs       7.298e-10  3.783e-08   0.019   0.9846
## ‘Size(in MB)‘  4.710e-02  6.313e-02   0.746   0.4557
## Price         -1.549e-01  7.603e-02  -2.037   0.0417 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 122.1 on 8113 degrees of freedom
## Multiple R-squared:  0.004306,   Adjusted R-squared:  0.003815
## F-statistic:  8.77 on 4 and 8113 DF,  p-value: 4.623e-07
```

Model 2

```
##
## Call:
## lm(formula = Rating ~ log(Reviews + 1) + log(Installs + 1) +
##     log(‘Size(in MB)‘ + 1) + log(Price + 1), data = df)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -432.73  -63.36    2.57   67.60  369.79 
## 
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)    
## (Intercept)             473.376      6.443  73.475  < 2e-16 ***
## log(Reviews + 1)         31.007      1.104  28.090  < 2e-16 ***
## log(Installs + 1)       -30.056      1.040 -28.905  < 2e-16 ***
## log(‘Size(in MB)‘ + 1)   -2.168      1.400  -1.549    0.122    
## log(Price + 1)          -11.349      2.646  -4.289 1.81e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 116.4 on 8113 degrees of freedom
## Multiple R-squared:  0.0947, Adjusted R-squared:  0.09425 
## F-statistic: 212.2 on 4 and 8113 DF,  p-value: < 2.2e-16
```

Model 3

```
## 
## Call:
## lm(formula = Rating ~ log(Reviews + 1) + log(Installs + 1) + 
##     Type + log(Price + 1) + Category, data = df)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -424.57  -64.05    3.68   66.99  370.74 
## 
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                  530.504     15.870  33.429  < 2e-16 ***
## log(Reviews + 1)              31.603      1.115  28.339  < 2e-16 ***
## log(Installs + 1)            -30.428      1.060 -28.715  < 2e-16 ***
## TypePaid                      10.507      9.005   1.167 0.243322    
## log(Price + 1)               -17.260      4.608  -3.746 0.000181 ***
## Categoryauto_and_vehicles    -59.329     19.885  -2.984 0.002857 ** 
## Categorybeauty               -13.460     22.262  -0.605 0.545463    
## Categorybooks_and_reference  -29.146     16.896  -1.725 0.084567 .  
## Categorybusiness             -62.444     15.871  -3.934 8.41e-05 ***
## Categorycomics               -63.126     22.274  -2.834 0.004608 ** 
## Categorycommunication        -87.949     16.617  -5.293 1.24e-07 ***
## Categorydating               -99.041     17.271  -5.735 1.01e-08 ***
## Categoryeducation            -14.724     19.153  -0.769 0.442045    
## Categoryentertainment        -84.924     20.554  -4.132 3.64e-05 ***
## Categoryevents                -4.783     21.249  -0.225 0.821929    
## Categoryfamily               -61.238     14.895  -4.111 3.98e-05 ***
## Categoryfinance              -68.062     16.117  -4.223 2.44e-05 ***
## Categoryfood_and_drink       -66.359     19.088  -3.477 0.000511 ***
##  [ reached getOption("max.print") -- omitted 19 rows ]
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 115.1 on 8081 degrees of freedom
## Multiple R-squared:  0.1189, Adjusted R-squared:  0.115
## F-statistic: 30.29 on 36 and 8081 DF,  p-value: < 2.2e-16
```

Model 4

```
##
## Call:
## lm(formula = Rating ~ log(Reviews + 1) + log(Installs + 1) +
##     Type + log(Price + 1) + Category + Content.Rating + Last.Updated,
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -377.83  -56.54    0.00   62.15  376.21
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  500.8226   105.3705   4.753 2.05e-06 ***
## log(Reviews + 1)              30.4907     1.2185  25.024  < 2e-16 ***
## log(Installs + 1)            -29.8140     1.1371 -26.219  < 2e-16 ***
## TypePaid                      20.1210    10.1020   1.992 0.046435 *
## log(Price + 1)               -16.8499     5.0079  -3.365 0.000771 ***
## Categoryauto_and_vehicles    -63.5622    20.2702  -3.136 0.001721 **
## Categorybeauty                -9.8455    22.7118  -0.433 0.664667
## Categorybooks_and_reference  -23.5375    17.7691  -1.325 0.185338
## Categorybusiness             -54.0187    16.2979  -3.314 0.000923 ***
## Categorycomics               -67.8500    22.8681  -2.967 0.003017 **
## Categorycommunication        -79.7491    17.0806  -4.669 3.08e-06 ***
## Categorydating               -87.2004    19.0523  -4.577 4.80e-06 ***
## Categoryeducation             -7.8130    20.1788  -0.387 0.698631
## Categoryentertainment        -85.1093    21.0611  -4.041 5.38e-05 ***
## Categoryevents                -3.3697    21.8255  -0.154 0.877303
## Categoryfamily               -47.9514    15.2607  -3.142 0.001684 **
## Categoryfinance              -68.7832    16.4781  -4.174 3.03e-05 ***
## Categoryfood_and_drink       -71.9431    19.4325  -3.702 0.000215 ***
##  [ reached getOption("max.print") -- omitted 1251 rows ]
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 113 on 6849 degrees of freedom
## Multiple R-squared:  0.2802, Adjusted R-squared:  0.147
## F-statistic: 2.103 on 1268 and 6849 DF,  p-value: < 2.2e-16
```

No Pooling Model

```
##
## Call:
## lm(formula = Rating ~ log(Reviews + 1) + log(Installs + 1) +
##     Type + log(Price + 1) + Category + Content.Rating + Last.Updated -
##     1, data = df)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -377.83  -56.54    0.00   62.15  376.21
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## log(Reviews + 1)                 30.4907     1.2185  25.024  < 2e-16 ***
## log(Installs + 1)               -29.8140     1.1371 -26.219  < 2e-16 ***
## TypeFree                        500.8226   105.3705   4.753 2.05e-06 ***
## TypePaid                        520.9436   105.7750   4.925 8.63e-07 ***
## log(Price + 1)                  -16.8499     5.0079  -3.365 0.000771 ***
## Categoryauto_and_vehicles       -63.5622    20.2702  -3.136 0.001721 **
## Categorybeauty                   -9.8455    22.7118  -0.433 0.664667
## Categorybooks_and_reference     -23.5375    17.7691  -1.325 0.185338
## Categorybusiness                -54.0187    16.2979  -3.314 0.000923 ***
## Categorycomics                  -67.8500    22.8681  -2.967 0.003017 **
## Categorycommunication           -79.7491    17.0806  -4.669 3.08e-06 ***
## Categorydating                  -87.2004    19.0523  -4.577 4.80e-06 ***
## Categoryeducation                -7.8130    20.1788  -0.387 0.698631
## Categoryentertainment           -85.1093    21.0611  -4.041 5.38e-05 ***
## Categoryevents                   -3.3697    21.8255  -0.154 0.877303
## Categoryfamily                  -47.9514    15.2607  -3.142 0.001684 **
## Categoryfinance                 -68.7832    16.4781  -4.174 3.03e-05 ***
## Categoryfood_and_drink          -71.9431    19.4325  -3.702 0.000215 ***
##  [ reached getOption("max.print") -- omitted 1251 rows ]
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 113 on 6849 degrees of freedom
## Multiple R-squared:  0.9139, Adjusted R-squared:  0.8979
## F-statistic: 57.28 on 1269 and 6849 DF,  p-value: < 2.2e-16
```

Partial Pooling Model

```
## Linear mixed model fit by REML [’lmerMod’]
## Formula: Rating ~ log(Reviews + 1) + log(Installs + 1) + Type + log(Price +
##     1) + Content.Rating + Last.Updated + (1 | Category)
##    Data: df
##
## REML criterion at convergence: 86087.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3444 -0.5034  0.0000  0.5463  3.2942
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Category (Intercept)   537.1   23.17
##  Residual             12766.9  112.99
## Number of obs: 8118, groups:  Category, 33
##
## Fixed effects:
```

```
##                                  Estimate Std. Error t value
## (Intercept)                     443.70489  104.22613   4.257
## log(Reviews + 1)                 30.30731    1.21457  24.953
## log(Installs + 1)               -29.64737    1.13393 -26.146
## TypePaid                         20.62964   10.08610   2.045
## log(Price + 1)                  -16.94699    5.00293  -3.387
## Content.RatingEveryone          -37.50581   80.96628  -0.463
## Content.RatingEveryone 10+      -48.38322   81.28689  -0.595
## Content.RatingMature 17+        -60.53890   81.31160  -0.745
## Content.RatingTeen              -52.37760   81.04158  -0.646
## Content.RatingUnrated           -43.30352  153.62251  -0.282
## Last.UpdatedApril 1, 2017        70.95585   86.45859   0.821
## Last.UpdatedApril 1, 2018       155.66030   86.48493   1.800
## Last.UpdatedApril 10, 2013      159.98125  130.86601   1.222
## Last.UpdatedApril 10, 2014       80.16463  130.62316   0.614
## Last.UpdatedApril 10, 2015      -17.07739  103.31374  -0.165
## Last.UpdatedApril 10, 2016       26.36854   92.45550   0.285
## Last.UpdatedApril 10, 2018       89.91618   71.98773   1.249
## Last.UpdatedApril 11, 2011      -70.03763  130.80415  -0.535
## Last.UpdatedApril 11, 2014       54.86489  130.78712   0.419
## Last.UpdatedApril 11, 2016      -55.07405  130.64593  -0.422
## Last.UpdatedApril 11, 2017       69.19457   92.37052   0.749
## Last.UpdatedApril 11, 2018      120.46605   72.07262   1.671
## Last.UpdatedApril 12, 2016      206.95660  103.31623   2.003
## Last.UpdatedApril 12, 2017       63.60227   78.15710   0.814
## Last.UpdatedApril 12, 2018       56.31819   71.66865   0.786
## Last.UpdatedApril 13, 2014      184.07867  130.81576   1.407
## Last.UpdatedApril 13, 2016       27.77678  103.37839   0.269
## Last.UpdatedApril 13, 2017       94.76941  103.40982   0.916
## Last.UpdatedApril 13, 2018       74.69444   72.54841   1.030
## Last.UpdatedApril 14, 2015      -23.85610  130.81892  -0.182
##  [ reached getOption("max.print") -- omitted 1207 rows ]

##
## Correlation matrix not shown by default, as p = 1237 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

Complete Pooling Model

```
##
## Call:
## lm(formula = Rating ~ log(Reviews + 1) + log(Installs + 1) +
##     Type + log(Price + 1) + Content.Rating + Last.Updated, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -388.14  -57.02    0.00   61.80  365.32
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     423.3597   104.9753   4.033 5.57e-05 ***
## log(Reviews + 1)                 30.1785     1.2019  25.109  < 2e-16 ***
## log(Installs + 1)               -29.5168     1.1303 -26.113  < 2e-16 ***
## TypePaid                         25.4767    10.0392   2.538 0.011179 *
## log(Price + 1)                  -18.1123     5.0003  -3.622 0.000294 ***
## Content.RatingEveryone          -29.5574    81.4997  -0.363 0.716863
## Content.RatingEveryone 10+      -35.5101    81.8116  -0.434 0.664268
## Content.RatingMature 17+        -59.9774    81.7392  -0.734 0.463116
## Content.RatingTeen              -40.6458    81.5702  -0.498 0.618294
## Content.RatingUnrated           -39.7409   155.1036  -0.256 0.797787
## Last.UpdatedApril 1, 2017        69.7400    87.2873   0.799 0.424335
## Last.UpdatedApril 1, 2018       172.2741    87.2910   1.974 0.048472 *
## Last.UpdatedApril 10, 2013      156.2501   132.0646   1.183 0.236797
## Last.UpdatedApril 10, 2014       93.9443   131.9277   0.712 0.476434
## Last.UpdatedApril 10, 2015       -4.6742   104.3677  -0.045 0.964279
## Last.UpdatedApril 10, 2016       46.9861    93.3303   0.503 0.614672
## Last.UpdatedApril 10, 2018       99.1002    72.7021   1.363 0.172895
## Last.UpdatedApril 11, 2011      -61.3135   132.1573  -0.464 0.642703
##  [ reached getOption("max.print") -- omitted 1219 rows ]
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 114.3 on 6881 degrees of freedom
## Multiple R-squared:  0.2604, Adjusted R-squared:  0.1275
## F-statistic:  1.96 on 1236 and 6881 DF,  p-value: < 2.2e-16
```

Interaction Model Output

```
##
## Call:
## lm(formula = Rating ~ log(Installs + 1) + log(Price + 1) + Type *
##     log(Reviews + 1) + Content.Rating + Last.Updated, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -389.33  -56.38    0.00   61.70  364.94
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     425.48368  104.94793   4.054 5.09e-05 ***
## log(Installs + 1)               -29.53810    1.13005 -26.139  < 2e-16 ***
## log(Price + 1)                  -17.95499    4.99927  -3.592 0.000331 ***
## TypePaid                          6.72302   13.00546   0.517 0.605216
## log(Reviews + 1)                 30.04344    1.20301  24.973  < 2e-16 ***
## Content.RatingEveryone          -30.12607   81.47562  -0.370 0.711576
## Content.RatingEveryone 10+      -36.74111   81.78875  -0.449 0.653287
## Content.RatingMature 17+        -60.65521   81.71512  -0.742 0.457945
## Content.RatingTeen              -41.32857   81.54620  -0.507 0.612303
## Content.RatingUnrated           -40.74243  155.05763  -0.263 0.792746
## Last.UpdatedApril 1, 2017        69.41948   87.26120   0.796 0.426329
## Last.UpdatedApril 1, 2018       171.46114   87.26553   1.965 0.049475 *
## Last.UpdatedApril 10, 2013      164.21828  132.07163   1.243 0.213762
## Last.UpdatedApril 10, 2014       93.63052  131.88805   0.710 0.477775
## Last.UpdatedApril 10, 2015       -4.21988  104.33648  -0.040 0.967740
## Last.UpdatedApril 10, 2016       46.31893   93.30270   0.496 0.619602
## Last.UpdatedApril 10, 2018       97.11878   72.68544    1.336 0.181544
## Last.UpdatedApril 11, 2011      -60.44879  132.11813   -0.458 0.647300
##  [ reached getOption("max.print") -- omitted 1220 rows ]
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 114.2 on 6880 degrees of freedom
## Multiple R-squared:  0.2609, Adjusted R-squared:  0.1281
## F-statistic: 1.964 on 1237 and 6880 DF,  p-value: < 2.2e-16
```

Reference

[1] J. P, A. Nagam, P. Undavalli, P. P, V. P. K. S and V. K. K. K, “Leveraging CAT Boost for Enhanced Prediction of App Ratings in the Google Play Store,” 2024 Second International Conference on Advances in Information Technology (ICAIT), Chikkamagaluru, Karnataka, India, 2024, pp. 1-6, doi: 10.1109/ICAIT61638.2024.10690600.

[2] Statista (2024) Google Play Store - Statistics & Facts. Available at: https://www.statista.com/topics/9929/google-play-store/#topicOverview (Accessed: 27 November 2024).

[3] S Shashank and Brahma Naidu, “Google play store apps-data analysis and ratings prediction”, International Research Journal of Engineering and Technology, vol. 7.12, pp. 265-274, 2020.

[4] Seo, M.K., Yang, O.S. and Yang, Y.H., 2020. Global Big Data Analysis Exploring the Determinants of Application Ratings: Evidence from the Google Play Store. Journal of Korea Trade, 24(7), pp.1-28.