SMU
SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

ISSS 616 Applied Statistical Analysis with R

Google Play Store App Success Prediction Tool

Project Report (Group 8)

Group Members (G1):
Debanjan Datta
Ayushi Shakya
Sarthak Nagapurkar
Ilansurya Ilanchezhiyan
Kruti Chandrashekhar

Introduction

In 2023, it is estimated that the annual number of app downloads worldwide will amount to 299 billion, up from approximately 247 billion global app downloads in 2020.

As of 2023, 3.5 billion apps are available for download on Google Play Store, with the number reaching 1.96 billion for Apple Play Store.

In 2020, global consumer spending on mobile apps reached approximately $143 billion, a significant increase from previous years. By the end of 2023, this number is expected to rise to $201 billion, with projections for 2025 even reaching levels of $270 billion, with a CAGR of 19.5%.

The above statements indicate a meteoric rise of mobile application usage in the recent past and how they will continue to play a major role in the lives of digital consumers and businesses – small, medium, or large.

The global surge in mobile app adoption has reshaped the business landscape, making these applications essential tools for enterprises worldwide. Mobile applications offer an unparalleled platform for businesses to engage with their customer base, building stronger connections through personalized experiences. A survey by Apptentive revealed that 89% of consumers are more likely to stay loyal to a brand with a seamless app experience (Apptentive, 2021). This statistic highlights the crucial role mobile apps play in fostering brand loyalty and customer retention.

Additionally, mobile apps provide a direct channel for communication, allowing businesses to convey updates, promotions, and offers in real-time. Push notifications, in-app messaging, and personalized recommendations strengthen customer relationships, fostering trust and a sense of community.

Mobile applications are powerful revenue generators. In 2020, global consumer spending on mobile apps reached $143 billion, a 20% increase from the previous year (Sensor Tower, 2021). This surge in consumer spending underscores the substantial revenue potential within the mobile app ecosystem.

Furthermore, apps enable businesses to unlock new revenue streams through data-driven insights. By leveraging analytics and user behaviour patterns, companies can fine-tune their offerings to meet the evolving needs and preferences of their audience.

The mobile app industry's unprecedented growth underscores the critical role these applications play in modern business strategies. From augmenting customer engagement to driving revenue growth, the benefits of mobile apps are clear and far-reaching.

Apps have been increasingly improved to meet every form of consumer demand. From banking, delivery, and cab-hailing apps to the ever-popular gaming, dating, and social media apps, mobile applications are used by every individual with a mobile device. Thus, launching a mobile app is a popular strategy for companies and other entities when it comes to creating a digital footprint or expanding an already existing one.

Objective

This project endeavors to construct an interactive R Shiny application designed to offer an assessment of a mobile application's success or failure. The success or failure of an app will be judged based on a unique scoring mechanism, which will combine factors that are present in the data. For the said assessment, predictive modelling techniques that will include a mix of statistical modelling techniques like Logistic Regression as well as Machine Learning techniques such as Decision Tree and Random Forest shall be used to generate the decision of whether the app, given a set of values for its variables, will be a success or failure in the present mobile application market.

Apart from serving as a prediction tool, the tool shall also empower users with data-driven strategies for exploring niche categories or executing market entry and business expansion through app development. By leveraging trends and insights from existing applications within their chosen category, users can make informed decisions and shape their app development endeavors effectively.

The R Shiny app will enable users to delve into critical metrics like ratings, download counts, pricing, and more for existing apps. Additionally, the solution will provide users with the capability to visualize the potential impact of variations in key variables on the projected success of their app. This dynamic functionality equips users with the ability to strike a balance between essential and desirable features, optimizing their app's potential for success and achieving desired outcomes.

While numerous methodologies have been developed by researchers to forecast the success of mobile applications, an interactive, user-friendly solution for examining and estimating application success remains conspicuously absent. This project seeks to fill this gap by offering a practical, hands-on tool that empowers users to assess the preliminary success estimate of their mobile application ventures with unprecedented ease and precision.

Data Sources

For analysis, the dataset was sourced from Kaggle. https://www.kaggle.com/datasets/gauthamp10/google-playstore-apps. The dataset has been obtained through web scraping from Google Play Store in June 2021. It contains application data for 231,944 apps worldwide from various categories. From the data obtained, we identified key app success indicators, and variables that significantly impact app success, and measures that can also be used to compare the popularity of various mobile applications.
The following table includes a list of variables relevant to our project:

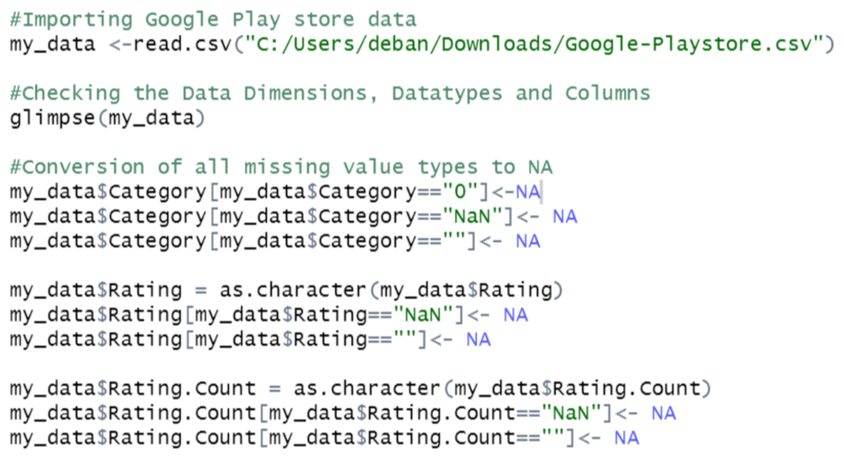
Variable	Description	How was it used
App Name	Name of Mobile Application	To identify an app.
App ID	ID of Mobile Application	To uniquely identify an app
Category	Genre of App	To indicate the genre that the App belongs to.
Rating	Average Rating	A key app success metric
Rating Count	Number of Ratings	Jointly used with Rating to measure app success.
Maximum Installs	Maximum count of installs	A measure of the exact number of times that the App has been installed. This is the variable that we used in our analysis as it gives an exact measure of App Installs.
Free	Boolean variable to denote paid or free app	To indicate if the app is free or not.
Price	App Price	To determine how price impacts installs, ratings, and overall app success.
Size	Size of the application package	We used this to explore relations between size and max installs, size and Avg Rating
Content Rating	The maturity level of the app	To gauge the impact the content rating has on app success.
Ad-supported	Ad support in the app	To compare apps supporting ads vs. not supporting ads.
In-App Purchases	In-app purchases in the app	To compare apps with in-app purchases vs. without in-app purchases
Released	App launch date on Google Play Store	To indicate how long the app has been available to users
Last Updated	The last app update date	To determine the impact of updates on app success.

Data Preparation

1. Data Cleaning

1.1 Missing Value Check

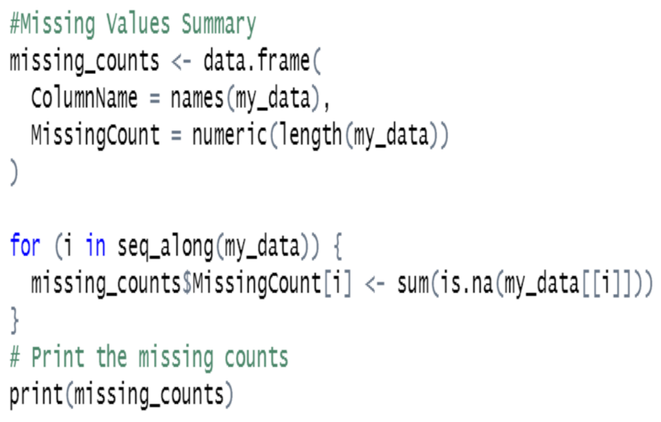
The raw data file was imported into the RStudio environment and then scanned for missing values. It was observed that missing values were not just represented by “NA” values but also by “blanks”. In some variables with char or date datatype, 0’s was treated as missing values along with the NA’s and blanks.



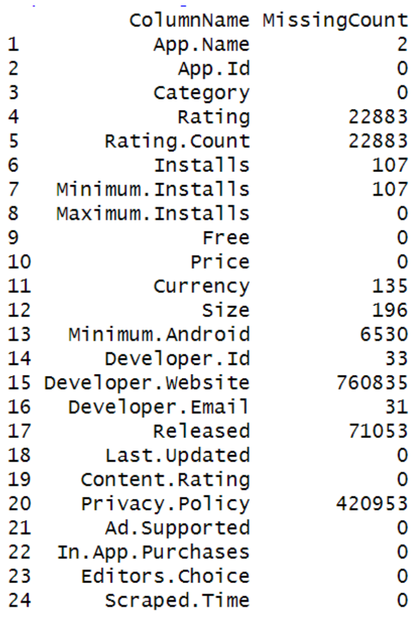
```r
#Importing Google Play store data
my_data <-read.csv("C:/Users/deban/Downloads/Google-Playstore.csv")

#Checking the Data Dimensions, Datatypes and Columns
glimpse(my_data)

#Conversion of all missing value types to NA
my_data$Category[my_data$Category=="0"]<-NA
my_data$Category[my_data$Category=="NaN"]<- NA
my_data$Category[my_data$Category==""]<- NA

my_data$Rating = as.character(my_data$Rating)
my_data$Rating[my_data$Rating=="NaN"]<- NA
my_data$Rating[my_data$Rating==""]<- NA

my_data$Rating.Count = as.character(my_data$Rating.Count)
my_data$Rating.Count[my_data$Rating.Count=="NaN"]<- NA
my_data$Rating.Count[my_data$Rating.Count==""]<- NA
```

The above screenshot does not contain a complete list of variables on which missing value check was implemented. For greater detail, please refer to the code base.

After checking for missing values, the following data frame containing variables and their corresponding missing value count was generated.



```r
#Missing Values Summary
missing_counts <- data.frame(
  ColumnName = names(my_data),
  MissingCount = numeric(length(my_data))
)

for (i in seq_along(my_data)) {
  missing_counts$MissingCount[i] <- sum(is.na(my_data[[i]]))
}
# Print the missing counts
print(missing_counts)
```

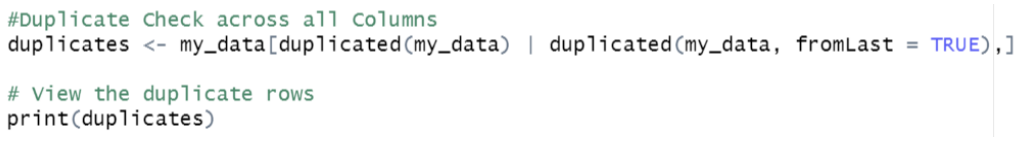


```
          ColumnName MissingCount
1           App.Name            2
2             App.Id            0
3           Category            0
4             Rating        22883
5       Rating.Count        22883
6           Installs          107
7   Minimum.Installs          107
8   Maximum.Installs            0
9               Free            0
10             Price            0
11          Currency          135
12              Size          196
13   Minimum.Android         6530
14      Developer.Id           33
15 Developer.Website       760835
16   Developer.Email           31
17          Released        71053
18      Last.Updated            0
19    Content.Rating            0
20    Privacy.Policy       420953
21      Ad.Supported            0
22  In.App.Purchases            0
23    Editors.Choice            0
24      Scraped.Time            0
```

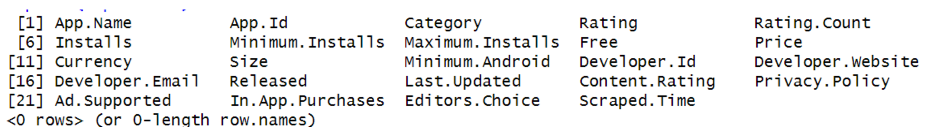
Missing Value Treatment: As seen in the above table, there are many missing values in several variables in the data. There are several variables that won't be used in the final modelling process, so we have removed those variables from the data altogether.

1.2 Duplicate Value Check:

The data was scanned for duplicate across all columns, and we found no duplicates in the dataset.



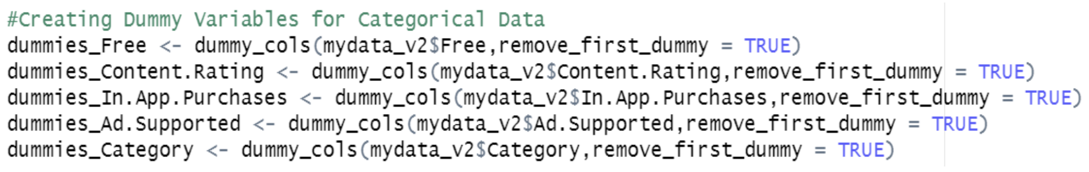
```
#Duplicate Check across all Columns
duplicates <- my_data[duplicated(my_data) | duplicated(my_data, fromLast = TRUE),]

# View the duplicate rows
print(duplicates)
```



```
 [1] App.Name         App.Id            Category          Rating            Rating.Count
 [6] Installs         Minimum.Installs  Maximum.Installs  Free              Price
[11] Currency         Size              Minimum.Android   Developer.Id      Developer.Website
[16] Developer.Email  Released          Last.Updated      Content.Rating    Privacy.Policy
[21] Ad.Supported     In.App.Purchases  Editors.Choice    Scraped.Time
<0 rows> (or 0-length row.names)
```

2. Feature Engineering

Dummy Variable Creation: Owing to the high count of nominal categorical variables in the data, dummy variables were created for such variables.



```
#Creating Dummy Variables for Categorical Data
dummies_Free <- dummy_cols(mydata_v2$Free,remove_first_dummy = TRUE)
dummies_Content.Rating <- dummy_cols(mydata_v2$Content.Rating,remove_first_dummy = TRUE)
dummies_In.App.Purchases <- dummy_cols(mydata_v2$In.App.Purchases,remove_first_dummy = TRUE)
dummies_Ad.Supported <- dummy_cols(mydata_v2$Ad.Supported,remove_first_dummy = TRUE)
dummies_Category <- dummy_cols(mydata_v2$Category,remove_first_dummy = TRUE)
```

If there were 'n' levels in a categorical variable, 'n-1' dummy variables were created. The resultant dummy variables were then joined with the cleaned dataset.

3. Creation of Target Variable

One of the most unique aspects of this tool is how it scores the success measure of an app using a combination of Rating, Rating and Maximum Installs. In this dataset, Maximum Installs is the measure of exact No. of Installs. Rating and Installs are both important factors that the target audience considers before downloading an app. However, they both are very different in nature.

- Rating: It is a subjective measure of app success as it depends on the opinion of the audience and opinion is subjective. Other apps with the same parameters might not be rated so badly. For example, when the CEO of Snapchat Evan Spiegel passed a comment saying that he doesn't want to expand to poor countries like India and Spain, it led to people rating the app badly and the resultant rating dropped to very low levels which wasn't indicative of how the App Category or other parameters performed.
- Installs: They are a much more objective measure of app success. If an app user downloads an app, that person is spending mobile data, storage

space as well as time behind installing that app. It indicates the demand for that app category by that user as well as the demand for other app parameters.

- Rating Count: This is a variable that is closely tied to the Rating variable. The Rating Count variable has different interpretations for different Ratings. If the Rating is high, then a high Rating Count would mean a good thing since the high rating is a result of the high positive Rating Count. However, for low ratings, a high Rating Count changes meaning. It means that the high count of negative ratings has led to the low rating. So, in our score calculation, we need to combine the interaction between Rating and Rating Count.

Scoring Methodology Rationale:

The below table outlines the logic we have used in our analysis for interpreting App success.

App ID	App Rating	Rating Count	Interpretation of App Success
App A	1.5	500	App A is better than App B as it has lesser number of "Low" (less than 3) ratings
App B	1.5	10,000	
App C	4.5	200	App D is better than App C as it has higher number of "High" (greater than or equal to 3) ratings
App D	4.5	50,000	

As indicated above, the interpretation of relationship between Rating and Rating Count is different for High Rated vs Low Rated Apps. So, we divide the Rating variable into two groups – High rating from 3.0-5.0 and Low rating from 1.0-2.9. Next, we create a new measure of Rating that changes the scale of Rating. For ratings above 3, the new variable accepts the rating as is. However, for ratings below 3, the new variable takes (5-rating) for each rating.

App ID	Original Rating	Scaled Rating
App A	Between 3 and 5 (inclusive)	Original Rating
App B	Less than 3	5-Original Rating

App Success score: Based on Scaled Rating, Rating Count and Installs we calculate the score for the app using the equation below:

$$Score = Installs\ [+\ or -]\ (Scaled\ Rating)\ \hat{}\ Rating\ Count$$

The + or – sign in the equation above is based on the Original Rating, if the Original Rating >= 3, then we use + sign in the equation above else we would use – sign.

Transformation of Score: The resultant score has range (-Inf to Inf), so it was required to standardize the scores. Since our score has zeroes and negative values, normalization or standardization would not work. So, we converted the scores using a sigmoid function to a range between 0 and 1. This was our choice of transform as Logistic Regression is based on Sigmoid function too.

Transformed Score Threshold: Finally, using a threshold of 0.5 (If sigmoid score > 0.5, then 1 else 0), the outcome of app success/failure is determined.

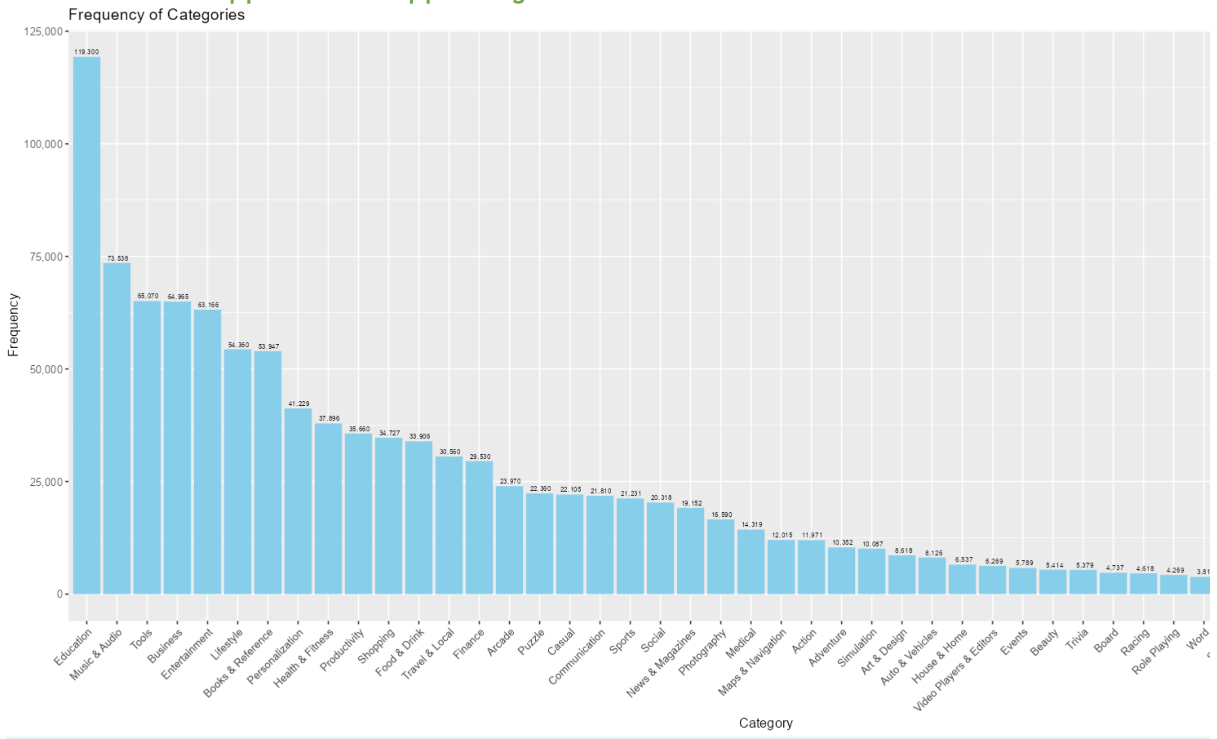
Please refer to the examples listed below for the score and subsequent transformations to arrive at Success/ Failure Decision:

App ID	Original Rating	Scaled Rating	Rating Count	Installs	Score	Sigmoid Transform	Sigmoid Threshold	App Success Indicator
App 1	3.5	3.5	100	55,000	5.50E+04	1.00E+00	1	Success
App 2	1.8	3.2	1,000	700	-3.20E+04	8.77E-14	0	Failure

Exploratory Data Analysis

We performed EDA to analyse and understand trends of apps in the Google play ecosystem.

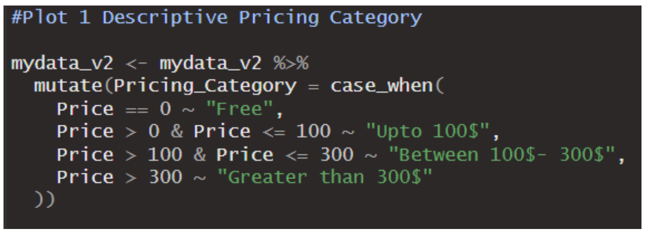
1. Distribution of Apps across App Categories



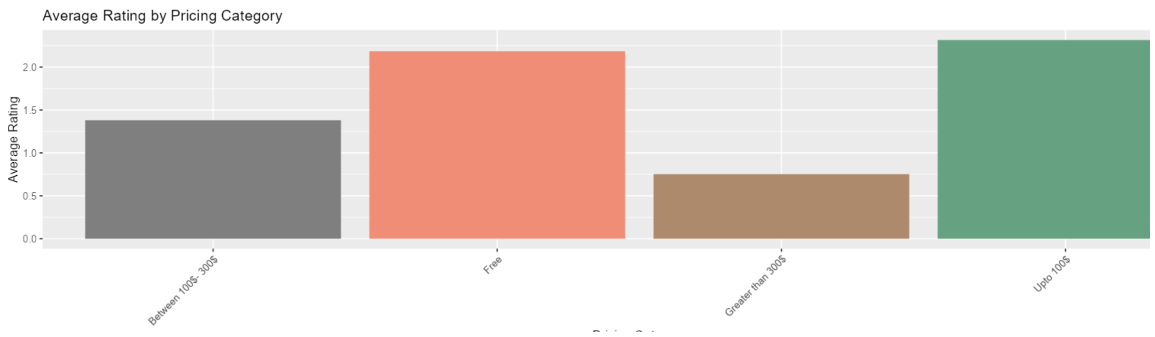
The above figure shows the frequency of different app categories in the data. Since our dataset is large, consisting of over 2 million apps, we can consider this to be an accurate indication of the market share of each app category. 'Education' apps lead the way, followed by 'Music & Audio' and 'Tools'. Apps such as 'Racing', 'Role Playing' and 'Word' bring up the rear. This, however, does not mean that 'Word' category does not lead to app success. It might well be the case that ‘Word’ is a category with very little participation, but it heavily impacts app success. This is the kind of insight that we wish to supply our user with.

2. Average App Rating by Pricing Categories

The price is a continuous variable which was converted to a categorical for the purpose of examining the trends of App Rating by pricing category.



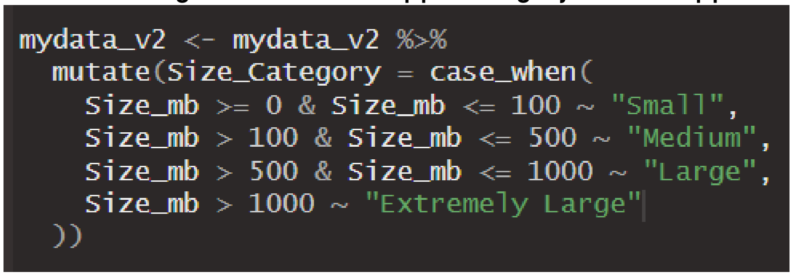
```
#Plot 1 Descriptive Pricing Category

mydata_v2 <- mydata_v2 %>%
  mutate(Pricing_Category = case_when(
    Price == 0 ~ "Free",
    Price > 0 & Price <= 100 ~ "Upto 100$",
    Price > 100 & Price <= 300 ~ "Between 100$- 300$",
    Price > 300 ~ "Greater than 300$"
  ))
```



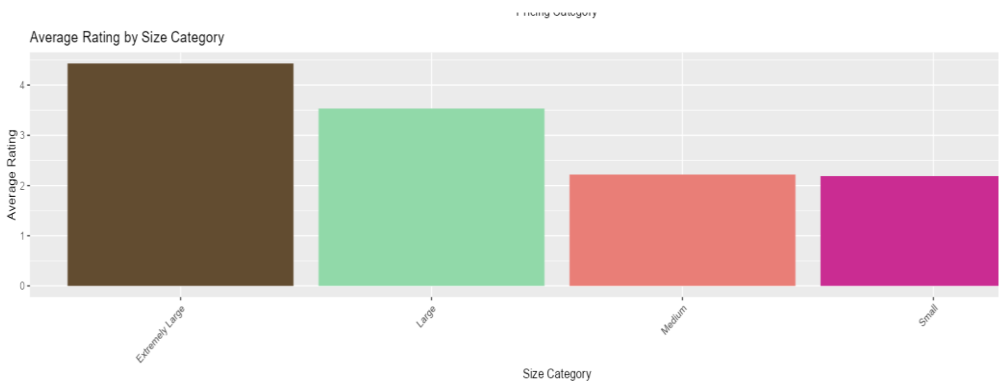
From the above figure, it can be easily judged that Free apps as well as apps with a price limit of $100 have a similar average rating which is also higher than the other categories. Beyond $100, the average rating drops, indicating that after a certain threshold, price has a negative impact on app popularity.

3. Average App Rating by App Size

The size of app is a continuous variable which was converted to a categorical for the purpose of examining the trends of App Rating by size of app.

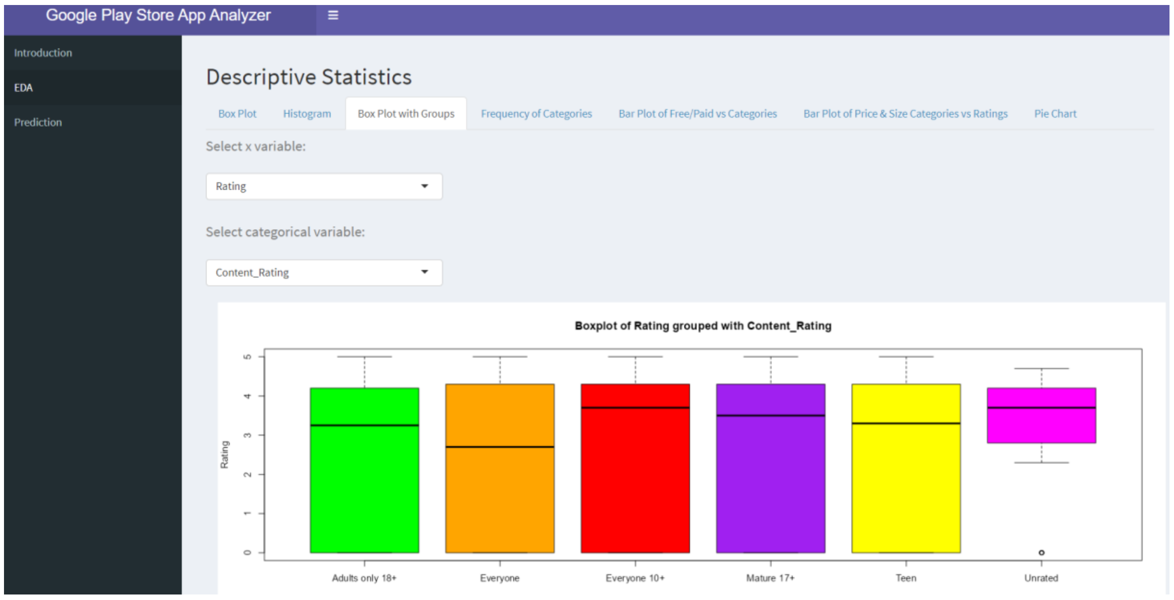


```
mydata_v2 <- mydata_v2 %>%
  mutate(Size_Category = case_when(
    Size_mb >= 0 & Size_mb <= 100 ~ "Small",
    Size_mb > 100 & Size_mb <= 500 ~ "Medium",
    Size_mb > 500 & Size_mb <= 1000 ~ "Large",
    Size_mb > 1000 ~ "Extremely Large"
  ))
```



In the above figure, we can see that the Average Rating increases with the size of the app. This might be because apps with larger size tends to have more features and better graphics. So, app size has a positive relationship with average rating.

Inferential Analysis



Descriptive Statistics

Select x variable: Rating

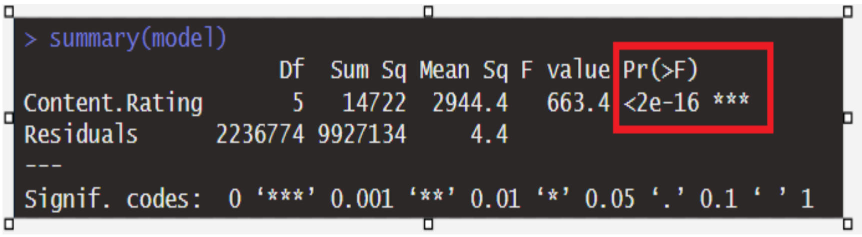
Select categorical variable: Content_Rating

ANOVA Test: To analyse whether the App Rating means is same across Content Rating Groups.

H0: The Mean of App Rating is same across all Content Rating Groups

H1: At least one Content Rating group has a different mean

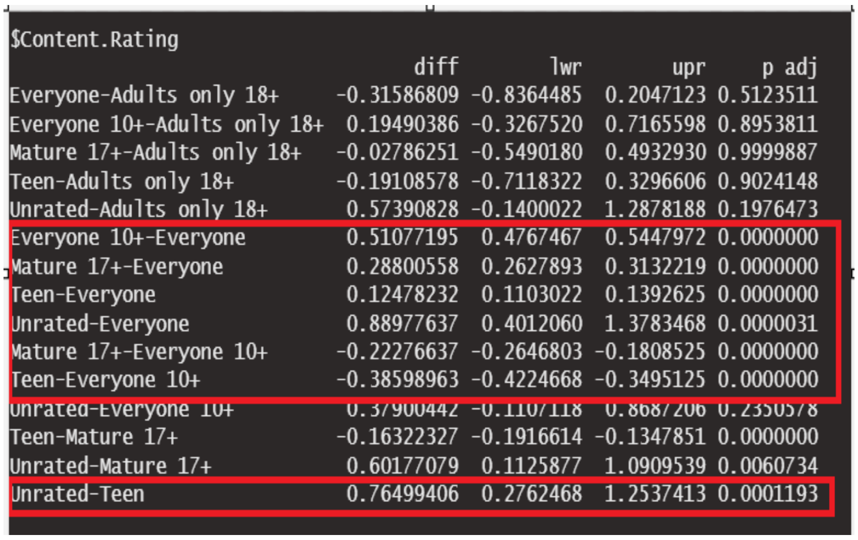
ANOVA Test Results:



```
> summary(model)
                    Df  Sum Sq Mean Sq F value Pr(>F)    
Content.Rating       5   14722  2944.4   663.4 <2e-16 ***
Residuals      2236774 9927134     4.4                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Since the p value of the AOV test is <0.05, we reject H0 and have sufficient evidence to support the statement that at least one population mean is different. To find means that are significantly different from each other, we use the single-step multiple comparison Tukey's HSD Test

Tukey Test Results:

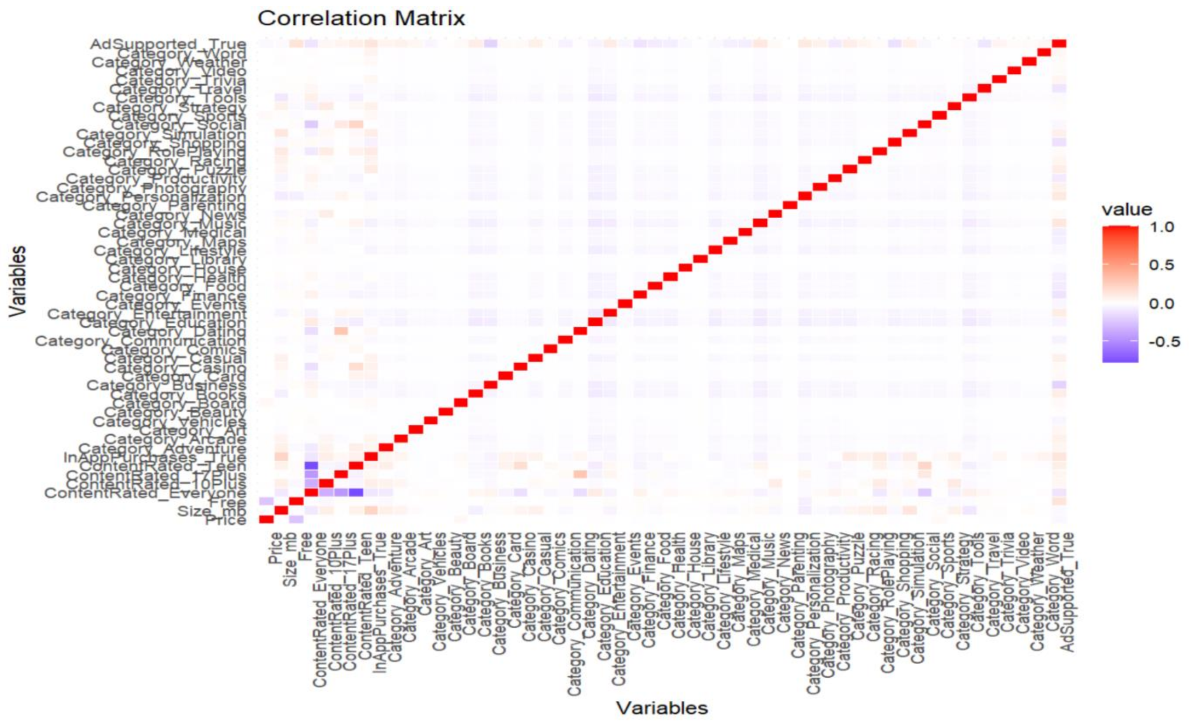


```
$Content.Rating
                                diff        lwr        upr     p adj
Everyone-Adults only 18+     -0.31586809 -0.8364485  0.2047123 0.5123511
Everyone 10+-Adults only 18+  0.19490386 -0.3267520  0.7165598 0.8953811
Mature 17+-Adults only 18+   -0.02786251 -0.5490180  0.4932930 0.9999887
Teen-Adults only 18+         -0.19108578 -0.7118322  0.3296606 0.9024148
Unrated-Adults only 18+       0.57390828 -0.1400022  1.2878188 0.1976473
Everyone 10+-Everyone         0.51077195  0.4767467  0.5447972 0.0000000
Mature 17+-Everyone           0.28800558  0.2627893  0.3132219 0.0000000
Teen-Everyone                 0.12478232  0.1103022  0.1392625 0.0000000
Unrated-Everyone              0.88977637  0.4012060  1.3783468 0.0000031
Mature 17+-Everyone 10+      -0.22276637 -0.2646803 -0.1808525 0.0000000
Teen-Everyone 10+            -0.38598963 -0.4224668 -0.3495125 0.0000000
Unrated-Everyone 10+          0.37900442 -0.1107118  0.8687206 0.2350578
Teen-Mature 17+              -0.16322327 -0.1916614 -0.1347851 0.0000000
Unrated-Mature 17+            0.60177079  0.1125877  1.0909539 0.0060734
Unrated-Teen                  0.76499406  0.2762468  1.2537413 0.0001193
```

From the results, we note that the among the content rating groups the difference in means is statistically significant for Unrated-Teen and Teen-Everyone 10+ groups

Pre-Modelling Check:

We checked for multicollinearity in our dataset as it is essential for regression analysis. It ensures the reliability of coefficient estimates, prevents unstable model outcomes, and maintains predictive accuracy. Identifying and addressing multicollinearity improves the interpretability and overall quality of the regression models.



Predictive Modelling

For our App success measure, we deployed 3 classification modelling methods- Logistic Regression, Random Forest, and Decision Tree. We specifically decided to use these techniques as all 3 are classification methods and are relatively easier to interpret than other algorithms.

Due to computational limitations of R, our model was run for 50,000 records in the dataset while retaining the 53 independent variables (listed below) used in prediction:

```
 [1] "Price"                     "Size_mb"                   "Free"
 [4] "ContentRated_10Plus"       "ContentRated_17Plus"       "ContentRated_Teen"
 [7] "InAppPurchases_True"       "Category_Adventure"        "Category_Arcade"
[10] "Category_Art"              "Category_Vehicles"         "Category_Beauty"
[13] "Category_Board"            "Category_Books"            "Category_Business"
[16] "Category_Card"             "Category_Casino"           "Category_Casual"
[19] "Category_Comics"           "Category_Communication"    "Category_Dating"
[22] "Category_Education"        "Category_Entertainment"    "Category_Events"
[25] "Category_Finance"          "Category_Food"             "Category_Health"
[28] "Category_House"            "Category_Library"          "Category_Lifestyle"
[31] "Category_Maps"             "Category_Medical"          "Category_Music"
[34] "Category_News"             "Category_Parenting"        "Category_Personalization"
[37] "Category_Photography"      "Category_Productivity"     "Category_Puzzle"
[40] "Category_Racing"           "Category_RolePlaying"      "Category_Shopping"
[43] "Category_Simulation"       "Category_Social"           "Category_Sports"
[46] "Category_Strategy"         "Category_Tools"            "Category_Travel"
[49] "Category_Trivia"           "Category_Video"            "Category_Weather"
[52] "Category_Word"             "AdSupported_True"          "flag"
```

Modelling Results and Selection:

Below are the key comparisons across the modelling techniques. This was taken into consideration while evaluating our model selection.

Model Method	Model Method Advantages	Model Method Limitations
Logistic Regression	• High train and test score • High interpretability • Low bias • Low variance as compared to Random Forest	• Training and Test score is not as high as Decision Tree or Random Forest • Bias is higher compared to Decision Tree
Random Forest	• High train and test score • Low variance as compared to Random Forest	• High variance as compared to Logistic Regression and Random Forest • Training score is not as high as Random Forest. • Difficult to interpret • Prone to overfitting • Time complexity • Expensive
Decision Tree	• High train and test score • Low bias • Low variance as compared to Random Forest	• Training score is not as high as Random Forest • Bias is higher compared to Logistic Regression • Difficult to interpret. • Uses a single tree, so output is not as robust as Random Forest

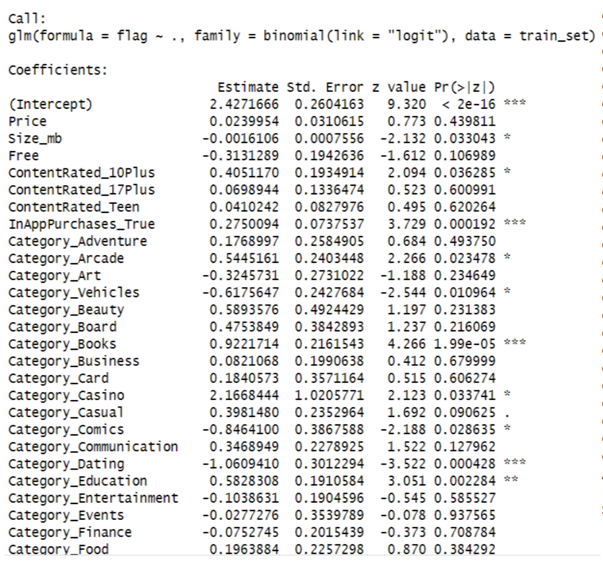
The below table lists the Train and Test scores across the Modelling Techniques-

Modelling Methods	Logistic Regression	Random Forest	Decision Tree
Train Score	96.43%	96.47%	96.43%
Test Score	96.57%	96.57%	96.57%

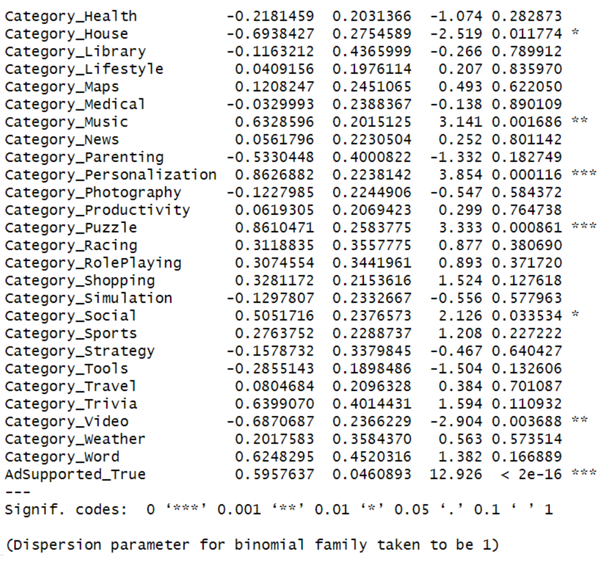
As indicated by the results above, Logistic Regression has the least variance between F1 Train and Test scores when compared with other two Modelling methods. Additionally, Logistic Regression has the highest interpretability which led us to choose this as the final modelling technique for our app success prediction solution.

Interpretation of Output

The Logistic Regression model generated the coefficients of the variables, some of which were significant at varying levels of significance and some of which were insignificant. The complete list of variable coefficients and their corresponding standard errors and p-values are shown in the figures below.



```
Call:
glm(formula = flag ~ ., family = binomial(link = "logit"), data = train_set)

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)    
(Intercept)             2.4271666  0.2604163   9.320  < 2e-16 ***
Price                   0.0239954  0.0310615   0.773 0.439811    
Size_mb                -0.0016106  0.0007556  -2.132 0.033043 *  
Free                   -0.3131289  0.1942636  -1.612 0.106989    
ContentRated_10Plus     0.4051170  0.1934914   2.094 0.036285 *  
ContentRated_17Plus     0.0698944  0.1336474   0.523 0.600991    
ContentRated_Teen       0.0410242  0.0827976   0.495 0.620264    
InAppPurchases_True     0.2750094  0.0737537   3.729 0.000192 ***
Category_Adventure      0.1768997  0.2584905   0.684 0.493750    
Category_Arcade         0.5445161  0.2403448   2.266 0.023478 *  
Category_Art           -0.3245731  0.2731022  -1.188 0.234649    
Category_Vehicles      -0.6175647  0.2427684  -2.544 0.010964 *  
Category_Beauty         0.5893576  0.4924429   1.197 0.231383    
Category_Board          0.4753849  0.3842893   1.237 0.216069    
Category_Books          0.9221714  0.2161543   4.266 1.99e-05 ***
Category_Business       0.0821068  0.1990638   0.412 0.679999    
Category_Card           0.1840573  0.3571164   0.515 0.606274    
Category_Casino         2.1668444  1.0205771   2.123 0.033741 *  
Category_Casual         0.3981480  0.2352964   1.692 0.090625 .  
Category_Comics        -0.8464100  0.3867588  -2.188 0.028635 *  
Category_Communication  0.3468949  0.2278925   1.522 0.127962    
Category_Dating        -1.0609410  0.3012294  -3.522 0.000428 ***
Category_Education      0.5828308  0.1910584   3.051 0.002284 ** 
Category_Entertainment -0.1038631  0.1904596  -0.545 0.585527    
Category_Events        -0.0277276  0.3539789  -0.078 0.937565    
Category_Finance       -0.0752745  0.2015439  -0.373 0.708784    
Category_Food           0.1963884  0.2257298   0.870 0.384292    
```



```
Category_Health         -0.2181459  0.2031366  -1.074 0.282873    
Category_House          -0.6938427  0.2754589  -2.519 0.011774 *  
Category_Library        -0.1163212  0.4365999  -0.266 0.789912    
Category_Lifestyle       0.0409156  0.1976114   0.207 0.835970    
Category_Maps            0.1208247  0.2451065   0.493 0.622050    
Category_Medical        -0.0329993  0.2388367  -0.138 0.890109    
Category_Music           0.6328596  0.2015125   3.141 0.001686 ** 
Category_News            0.0561796  0.2230504   0.252 0.801142    
Category_Parenting      -0.5330448  0.4000822  -1.332 0.182749    
Category_Personalization 0.8626882  0.2238142   3.854 0.000116 ***
Category_Photography    -0.1227985  0.2244906  -0.547 0.584372    
Category_Productivity    0.0619305  0.2069423   0.299 0.764738    
Category_Puzzle          0.8610471  0.2583775   3.333 0.000861 ***
Category_Racing          0.3118835  0.3557775   0.877 0.380690    
Category_RolePlaying     0.3074554  0.3441961   0.893 0.371720    
Category_Shopping        0.3281172  0.2153616   1.524 0.127618    
Category_Simulation     -0.1297807  0.2332667  -0.556 0.577963    
Category_Social          0.5051716  0.2376573   2.126 0.033534 *  
Category_Sports          0.2763752  0.2288737   1.208 0.227222    
Category_Strategy       -0.1578732  0.3379845  -0.467 0.640427    
Category_Tools          -0.2855143  0.1898486  -1.504 0.132606    
Category_Travel          0.0804684  0.2096328   0.384 0.701087    
Category_Trivia          0.6399070  0.4014431   1.594 0.110932    
Category_Video          -0.6870687  0.2366229  -2.904 0.003688 ** 
Category_Weather         0.2017583  0.3584370   0.563 0.573514    
Category_Word            0.6248295  0.4520316   1.382 0.166889    
AdSupported_True         0.5957637  0.0460893  12.926  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
```

Interpretation of Coefficients

- 19 variables in the model are statistically significant at 5% level of significance. The rest are statistically insignificant.
- For categorical variables, the coefficient value gives an estimate of how much the log-odds ratio of App success changes when the variable changes from 0 to 1, keeping other variables constant.
- For continuous variables, the coefficient value gives an estimate of how much the log-odds ratio of App success changes by when the variable increases by 1 unit.

Using Size_Mb as an example, the output was interpreted as follows.

H0: There is no association between App success and Size_mb

H1: There is an association between App success and Size_mb

The p-value for Size_mb is 0.033, which is less than the 0.05 level at 95% confidence level. Thus, we have sufficient statistical significance to reject H0 and infer that there is an association between Size_mb and App success. Similar hypothesis tests can be implemented for all the other variables.

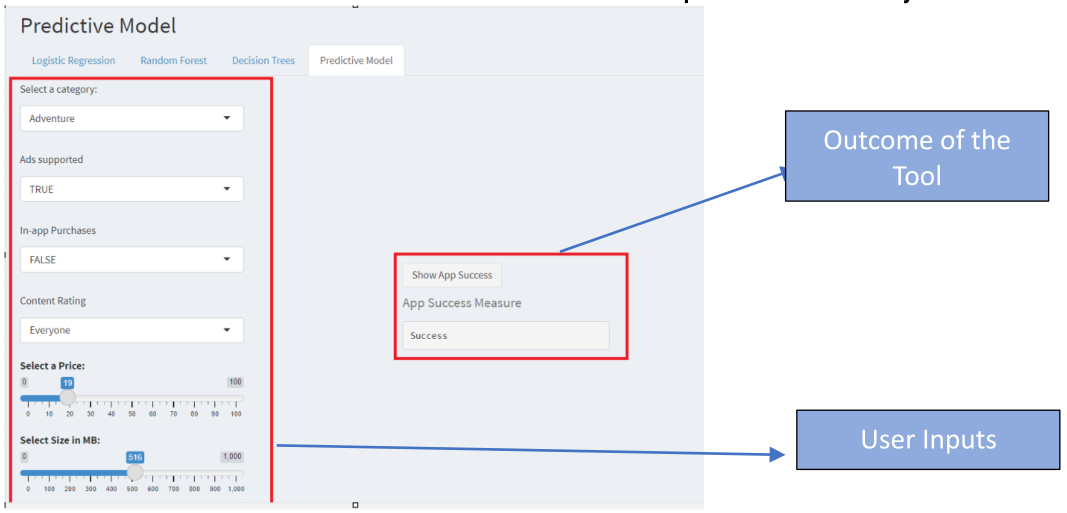
We have decided not to remove the statistically insignificant variables but to keep them as the coefficient gives us quantitative and qualitative information about their impact on App success. This information would help the user understand the variables that strongly impact the target variable in a positive or negative way. This is especially important to maintain transparency. For example, for variables – category_Dating and Category_Books, we see very different behaviour. For category_Dating, the coefficient is negative, which indicates that when this variable changes from 0 to 1, the app success odds ratio drops, which is surprising because Dating apps are very popular. However, the reason behind this could be the saturation in the Dating app market and the lack of opportunities for new players.

Statistical significance is just one aspect of model selection. It's important to consider the broader context, including theory, domain knowledge, and the purpose of the model. One should always use your best judgment and consult with domain experts when deciding whether to include or exclude variables from your model.

R Shiny Application

We have prepared an R Shiny App with 3 Tabs:

1. Introduction page which explains the user the intent of the solution and how they can use it.
2. Exploratory Data Analysis- This includes both univariate and bivariate, where multiple visualization techniques are employed to display the presence and absence of relationship between different categorical and continuous variables. The user can navigate between different options to
3. Predictive Modelling: The final tab is an interactive representation of the Google Play Store App Success Prediction Tool, where the user inputs the values of the parameters, and the model generated the output for that set of parameters. Our model results are run in base R code and then imported to R shiny.



Select a category: Adventure

Ads supported: TRUE

In-app Purchases: FALSE

Content Rating: Everyone

Select a Price: 19

Select Size in MB: 516

App Success Measure: Success

Conclusion & Further Scope

The Google Play App Success Prediction Tool aims to give the user an indicative idea of the parameters that might maximise chances of app success given the current market.

The predictions made by the tool should be well supplemented by practical domain knowledge as well as business requirements and strategies.

Further Scope

- Hyperparameter Tuning is a technique that could improve the model F1 score and pump up the predictive capabilities of the model.
- Algorithms such as XGBoost, Support Vector Machines as well as Machine Learning techniques such as Neural Networks would give more robust output.
- Unable to use the full extent of data due to computational limitations. Would get better estimates if we were able to use all data instead of taking a sample.
- In our analysis, owing to limitations of data, several variables such as Reviews that could improve the robust-ness of the model could not implemented.