ST309 Summative

What determines the success of android mobile applications in the google play stores?

Examination Numbers:

10295,15888,16854

Individual Contribution: 100%

Introduction

Over the past decade, the "mobile app economy" (Ghose,2014), which refers to the value created from the development and delivery of software applications, has growth phenomenally. The global app economy is estimated to be worth $6.3 trillion by 2021, up from $1.3 trillion in 2018 (AppAnnie,2019). Over this period the user base will almost double from 3.4 billion people using apps to around 6.3 billion (AppAnnie, 2019).

This rapid market growth has produced no shortage of stories of individual app developers making great fortunes from their apps. However, the majority of developers are still struggling to break even (AppSurvey, 2013). For those unsuccessful app developers, a clear analysis of the characteristics of existing successful apps would provide a useful insight into creating content that users want (Tian,2015). Therefore, the main goal of this project is to identify the characteristics that successful apps share and investigate which of these factors is important for success. This study is relevant for both developers and consumers, since both parties will benefit from a greater alignment of demand and supply in the app market; consumers will have a greater choice of apps they enjoy and developers will reap the financial gains of their work.

In order to analyse the traits of successful apps and the determinants of app success it is necessary to analyse a large dataset of both successful and unsuccessful apps. Once exploratory data analysis (EDA) is conducted, patterns and correlations should be revealed which will allow significant predictors of success to be identified. A model can then be fitted to the data to determine which characteristics are the most important for success.

In this project, we use a dataset of more than 10,800 apps from the Google Play Store. After intial EDA and cleansing, our final dataset had 6738 observations (4103 obs. omitted). We have focused our research on the Google Play Store as it is the largest app market by active users with over 2 billion Android devices (Google, 2019). For the 724,000 developers on the Google Play Store (Statista, 2017), knowing the "DNA" of successful apps will be important for their efforts to build successful apps.

Previous studies using data analysis have found a strong correlation between customer ranking and the rank of app downloads and surprisingly no correlation between price and downloads (Lanza,2012). Apps with high user rating have been shown to use APIs that are less fault-pone and change-prone than APIs used by low rated apps (Bavota, 2015). Moreover, high app churn has been correlated with lower user ratings (Guerrouj,2015). Other studies have analysed the impact of customer reviews using sentiment analysis to show that customer endorsements of service quality increase sales (Elizalde,2014).

Most of the current literature on app success focuses on app rating as a proxy for app success (Lee,2017). However, for this study we define a successful app to be an app with more than 500,000 downloads. It is important to note that traffic is only one of the many factors that determine "success" (other important factors incl. app retention rates or in-app purchases) but due to limited data, we are assuming success is wholly due to traffic. This assumption is reasonable in the sense that ultimately, traffic is needed before other factors can be considered as significant in determining success. Given that our research focus is on success in terms of profitability for app developers, we believe the 500,000 downloads assumption is appropriate (Louis,2013). Additionally, by focusing on free and paid downloads we intend to add to the literature on app success which currently focuses on reviews or free app downloads.

Methodology

Our project has two research questions:

1. What characteristics do successful apps share?
2. What factors are important for app success on the Google Play Store?

We followed The Cross Industry Standard Process for Data Mining (CRISP-DM) (Hesse, 2000) to conduct this project. We used this methodology to define our criteria for success; an app having more than 500,000 downloads. Measuring the success of a software application is difficult as there is neither a universal metric nor a ranking scheme (German,2007). Given the limitations of our data and our business understanding of

what app developers need to be profitable, we believe our assumption is defendable. Subsequently, we investigated 17 factors along eight dimensions to understand app success.

We began our project by cleaning our dataset. We recognised and removed, where appropriate, errors and outliers. We transformed a number of variables which were not numeric into numeric values (eg: category, type, genre). As the majority of our variables were normally distributed, it is safe to assume that the error terms for all variables are also normally distributed so this should not adversely affect our model.

We then conducted Exploratory Data Analysis (EDA) to identify correlations across predictors and other patterns in the dataset. Having defined success, we faced a supervised learning problem and consequently we chose to use classification trees. We partitioned our dataset into training and test data. Using the training data, we identified the characteristics of successful apps. We then predicted which apps in the test data would be successful based on the identified characteristics (Provost,2011). We also used a variety of methods to reduce variance and bias including k-fold cross validation, random forest and boostrap aggregation.

We then transformed our success variable from a categorical variable to a binary variable. This allowed us to apply a stepwise logistic regression model to identify the characteristics that were significant for success (Sheaffer-Jones,2004). Similarly, we tested our model on the test data. By plotting the ROC curves (Raschka,2015) for the logistic regression model and the classification tree model we were able to determine which model was the best predictive model for app success. From this, we determined which explanatory variables were significant for a mobile application's success on the google play store.

Data Description

The dataset we used for our analysis was taken from a publicly available source on Kaggle.

This dataset was web scraped from the Google Play store and contains information on 10,841 individual apps across 13 features. This dataset is appropriate for our research questions since it is large, contains a large number of successful and unsuccessful apps and is fairly granular.

The 13 features available for each observation in the dataset are:

1. **App** - Application name
2. **Category** - Category the app belongs to
3. **Rating** - Overall user rating of the app (as when scraped)
4. **Reviews** - Number of user reviews for the app (as when scraped)
5. **Size**- Size of the app (as when scraped)
6. **Installs** - Number of user downloads/installs for the app (as when scraped)
7. **Type** - Paid or Free
8. **Price** - Price of the app (as when scraped)
9. **Content Rating** - Age group the app is targeted at - Teen / Mature 21+ / Adult
10. **Genres** - An app can belong to multiple genres (apart from its main category). For eg, a musical family game will belong to Music, Game, Family genres.
11. **Last Updated** - Date when the app was last updated on Play Store (as when scraped)
12. **Current Ver** - Current version of the app available on Play Store (as when scraped)
13. **Android Ver** - Min required Android version (as when scraped)

We coded these 13 features into 13 variables across 8 dimensions. We then omitted any observations with missing values. For example, we deleted any observations without ratings. However, most applications without rating have very few installs which intuitively makes sense. By omitting these observations, we

reduced the number of unsuccessful apps in our dataset. This had the effect of making our dataset more evenly balanced between successful and unsuccessful apps. This means that the proportion of successful apps in our dataset is higher than in the real world. We also removed duplicated app observations as well as adjusted our variables accordingly to our exploratory data analysis(removed Android.Ver).

A summary of the final dataset can be seen after the exploratory analysis section

Below are the details of the analysis we have conducted to reach the final dataset:

Data Cleaning Methodology and Justification

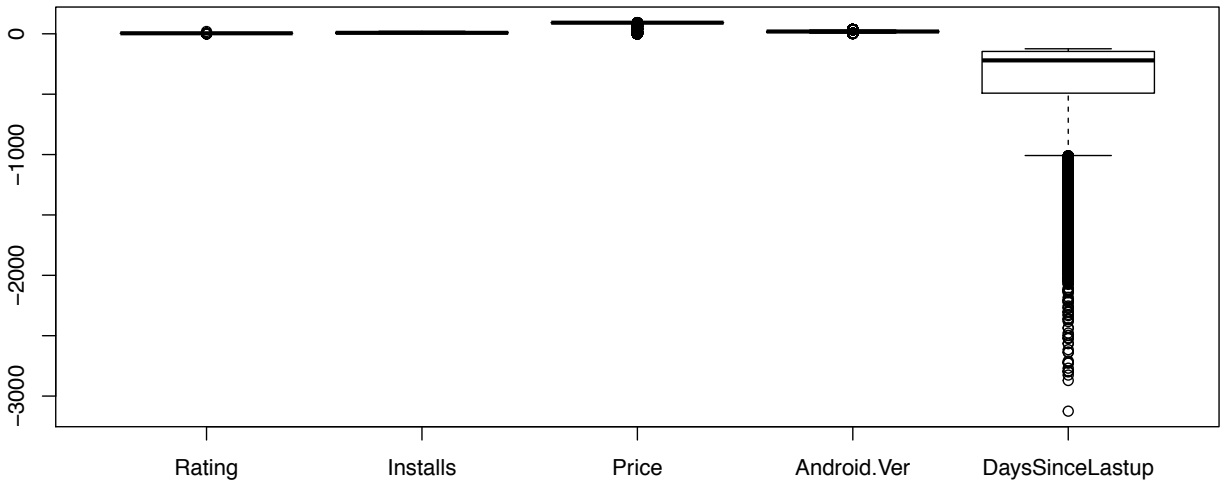
As the raw data file contained several duplicates, NAs, impossible values(as we will see below) and values in classes that cannot be used in modelling, data cleaning is needed. The following steps were taken to get the dataset used in modelling:

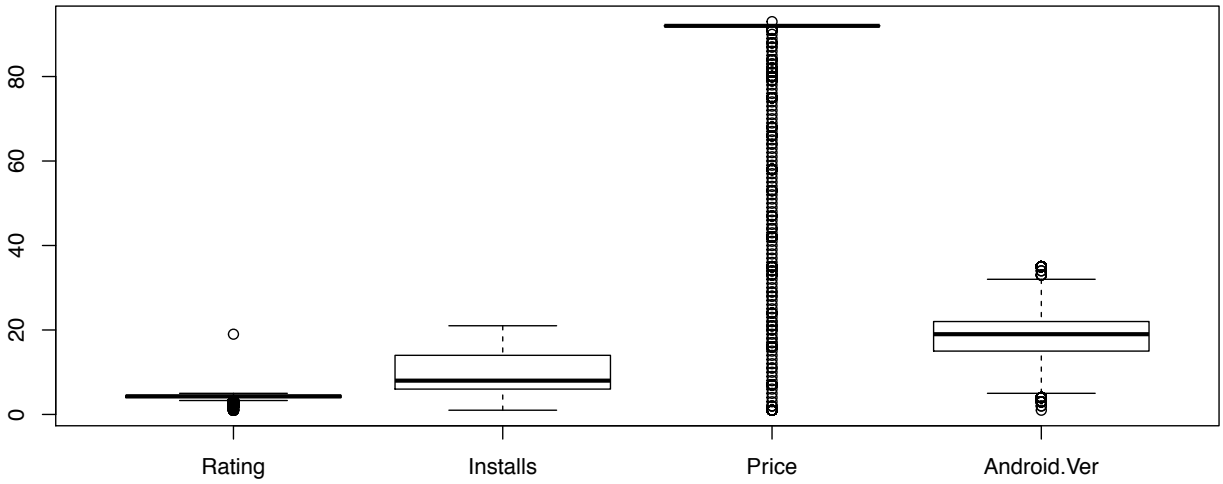
Dealing with duplicates and initial transformation of variables

Firstly, as there are duplicate rows for the same app in the dataset, we remove these via the distinct() function. This has reduced our observations from 10841 to 9660. We also removed variables that cannot be used in comparisons (Current ver.) since they under different formats for each app. Repeated variables were also removed: Genre (same inputs as Category) and Type (same as Price). We transformed the 'Last updated' variable into the 'date' class and created a new variable, 'DaysSinceLastup' computing the days from the 'last updated' date to the day we downloaded the dataset.

Checking the distribution of numeric-variables in order to find undetected errors or outliers.

Reviews and size's distribution are normally distributed and have no impossible/extreme values according to the boxplot. However, when we represent the distribution of other numeric variables (Rating, Installs, Price, Android.Ver, DaysSinceLastup):





As we see from the boxplot: DaysSinceLastup is quite skewed, but we would expect this as intuitively, most app developers update their apps frequently.It does not seem to have any extreme/outlier values so we remove this variable from the boxplot to show the rest of the 4 variables.

There is an outlier in 'Rating' (a rating value of 19), which is impossible as rating is out of 5. Price is also not distributed via the boxplot properly, which may mean that it is under the wrong (non-numeric) class. From function str() that displays the summary of each variable and its class, we find out that the predictor variables "Price", "Installs", "Android.Ver" are classified as 'factors'. We therefore convert them to the 'numeric' class to be useful for analysis.

We also found out from summary() that there appears to be a non-price data point under variable "Price". Removing it and the '$' signs, we are able to transform the datas to numeric. There are also NaN datapoints in "Android.Ver" which we also removed as well as a 'Free' under the "Installs" variable which is an error and has also been removed.

Due to the large number of possible categories and content rating options, we narrowed down the number of categories and options for a more insightful analysis.

For the 33 categories, we found that they could be easily categorised into the 4 broader category titles of "Hobbies", "Entertainment", "Productivity", "Lifestyle." For example, categories that fell under "Hobbies" include Art and Design, Health and Fitness, Sports, etc. Categories that fell under "Entertainment" include Comics and Games. Categories that fell under "Productivity" include Education and Books and References. Finally, categories that fell under "Lifestyle" include Beauty and Dating among others.

For Content Rating, we narrowed it down to 3 groups: "Everyone", "Mature 17+", "Teen" as the original 6 groups were repetitive, having similar ratings ie. "Everyone 10+" and "Everyone" , "Adults only 18+" and "Mature 17+"

We transformed each broad groups under 'Categories' and 'Content Rating' into dummy variables to be used in our regression and classification trees. This is because regression analysis treats all independent variables as numerical, thus if the categories were to be numbered for use in modelling, R would find meaning in them, when infact the numbers have no intrinsic meaning, referring to just the category.

We now have a dataset with 6738 observations and 14 predictor-variables after cleaning.

Assumption

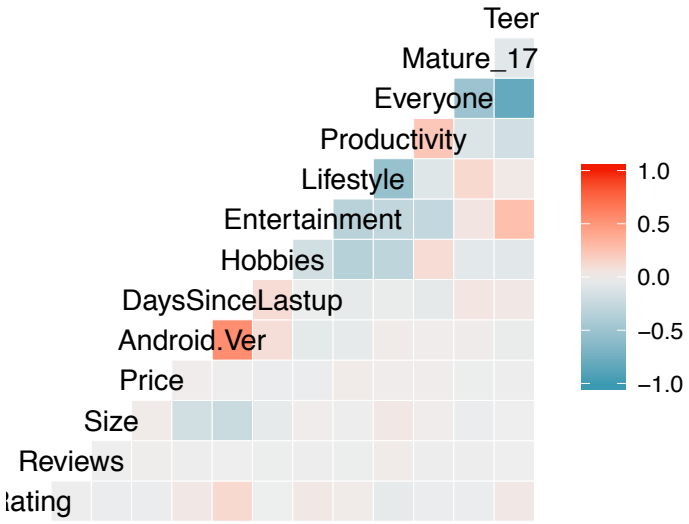
The objective of our project is to determine the predictors of mobile application success. However, we first need to define what exactly 'success' entails. We have decided to define an app's 'Success' by having more than 500,000 downloads, and from the data,

```
##   No  Yes
## 3909 2829
```

we can see that our dataset is fairly balanced.

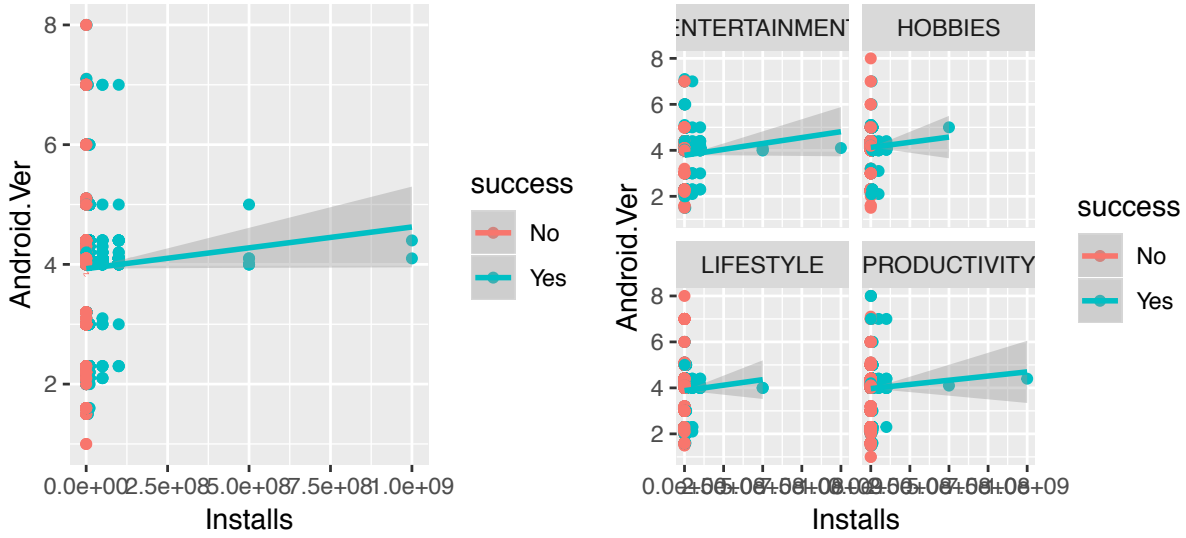
Exploratory Data Analysis

First, we explore the multicolinearity of variables as this would affect the reliability and stability of regression coefficient estimates when modelling the data

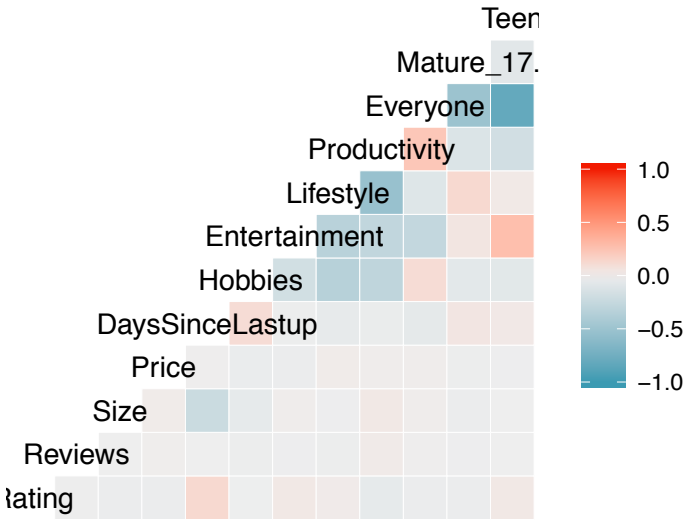


We see that most predictor variables have fairly weak correlations apart from Android.Ver and DaysSinceLastup. This is intuitive, as we would expect the more recent updated apps to have higher android version requirements. We therefore remove the Android.Ver variable as we can safely assume that this and DaysSinceLastup showcases the same results. We would expect Android.ver to influence success via its updates attracting more traffic (reflected in DaysSinceLastup). This is because intuitively, we would expect the lower the Android.ver, the more traffic, since more android systems are able to install it (which is not shown according to this correlation graph):

Android.Ver vs. Installs by Category



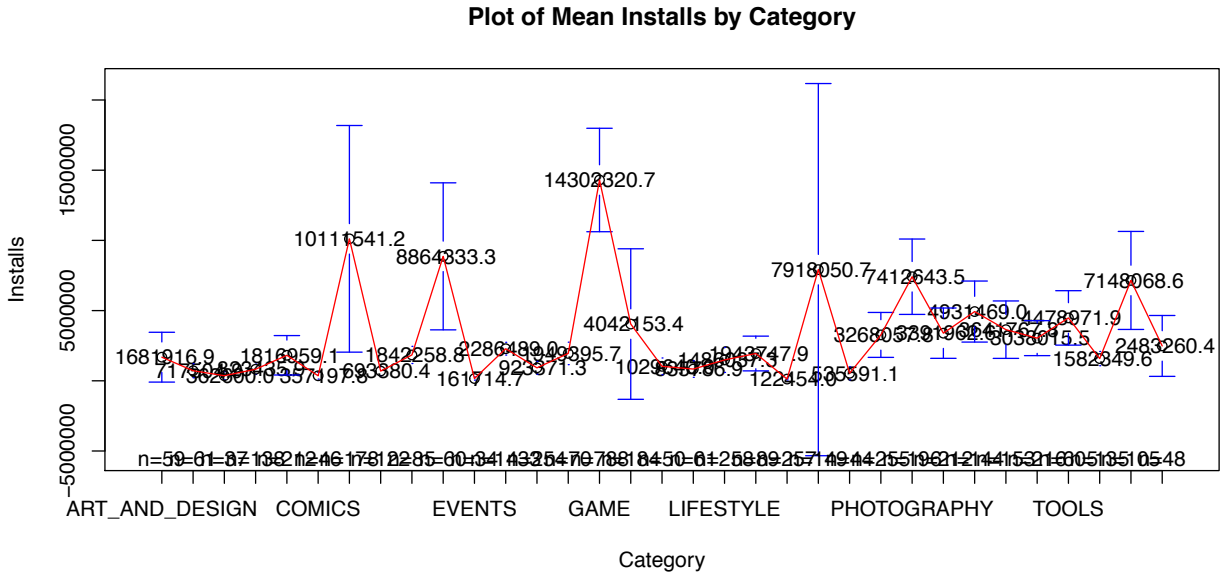
Our correlation plot with Android.Ver removed:



All predictor variables are fairly uncorrelated (Teen|Mature17|Everyone and Productivity|Lifestyle|Entertainment are dummy variables for each category / content.rating and hence are understandingly negatively correlated)

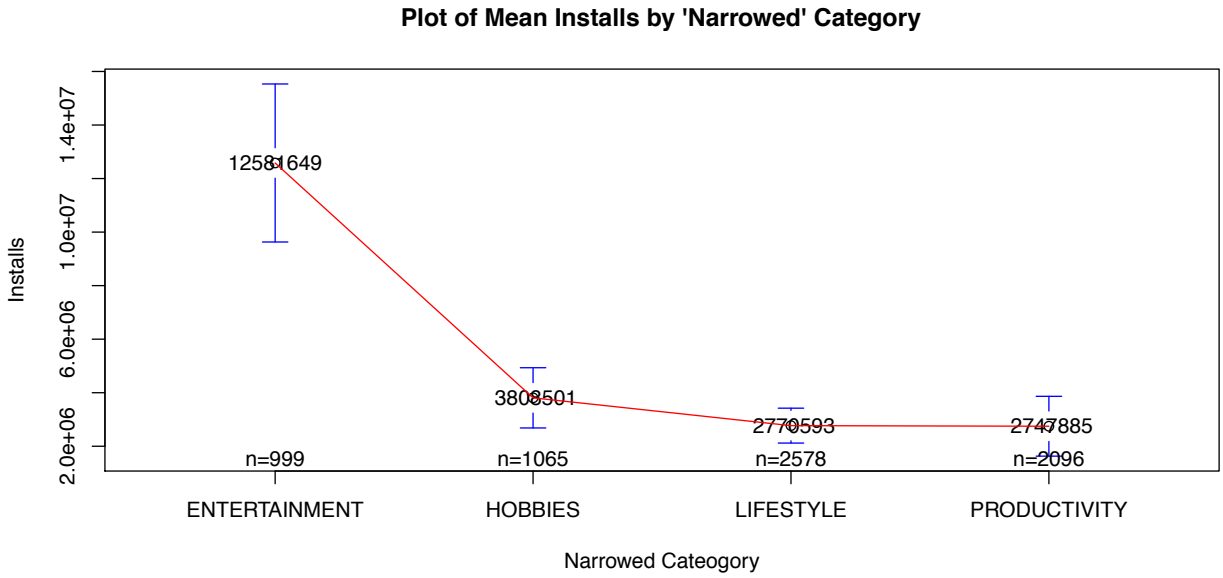
Categorical Exploration

We now perform the anova test to see whether there is a relationship between the narrowed categories and installs(the success determinant). The original plot:



Since the original plot could not provide valuable insights, having too many variables.

So for the new narrowed-category plot's ANOVA test



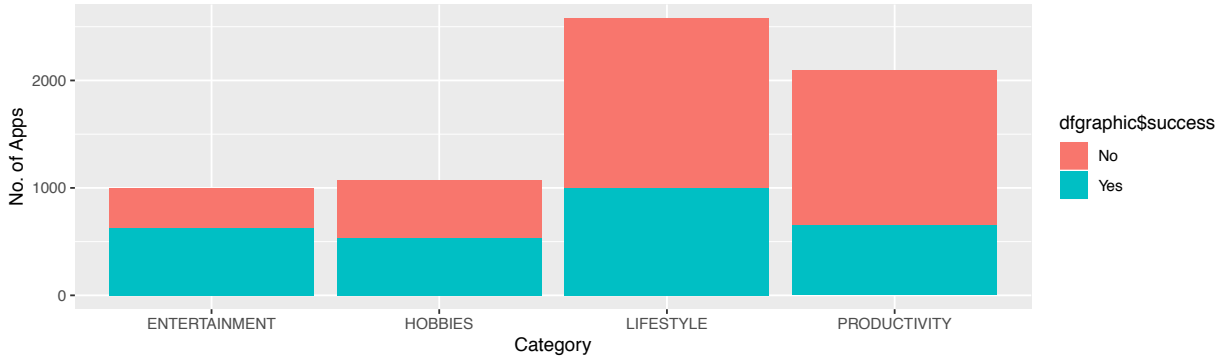
We define our H0 hypothesis as the mean Installs for each category being equal (different categories do not affect installs differently): 'Categories' as a variable is not significant in determining installs. H1 as the mean not being equal and category being significant.

Our Anova test gives us:

```
##                       Df    Sum Sq   Mean Sq F value Pr(>F)
## dfgraphic$Category     3 7.981e+16 2.660e+16   37.42 <2e-16 ***
## Residuals          6734 4.787e+18 7.108e+14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value is statistically significant with a high F value, the variation of mean installs among different categories is much larger than it's variation within each category. We therefore reject H0 and conclude that

there is a significant relationship between an app's category and the no. of installs (whether it is a success). It is meaningful to include this in the model. We expect to see entertainment as a significant variable in our predictive model as apps under this category seems to have higher installs. We can see from this barchart how a entertainment apps have more successful apps than the dataset average, and a higher proportion compared to other categories.



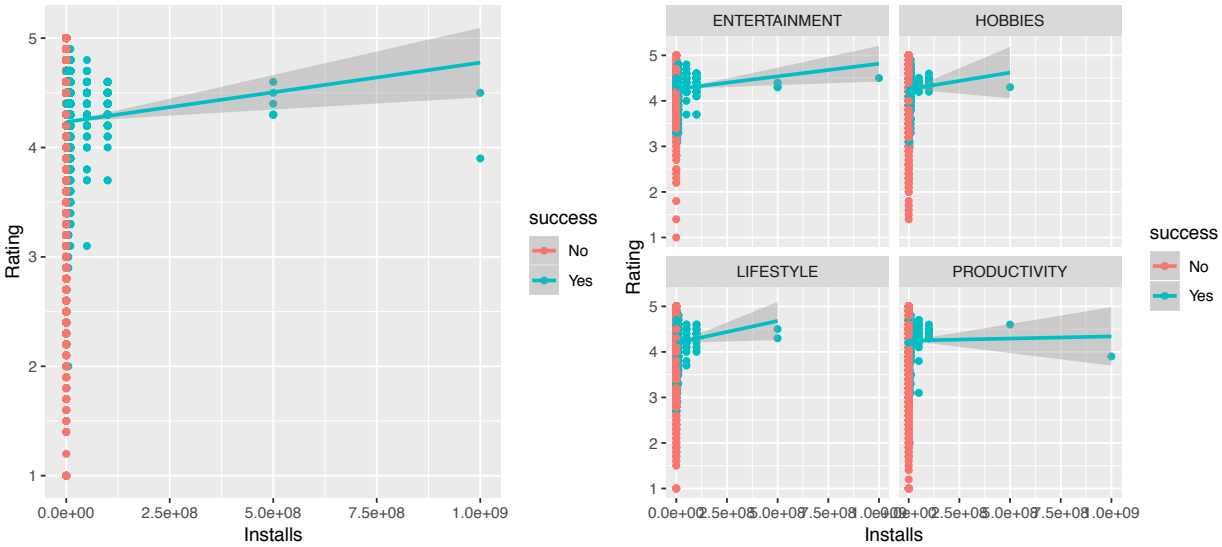
Anova test for content.rating:

```
##                              Df    Sum Sq   Mean Sq F value   Pr(>F)
## dfgraphic$Content.Rating      2 1.005e+16 5.024e+15   6.967 0.000949 ***
## Residuals                 6735 4.857e+18 7.211e+14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the p-value is much less significant for content.rating compared to category. However, as the relationship is still significant, it is meaningful to include this variable in the model.

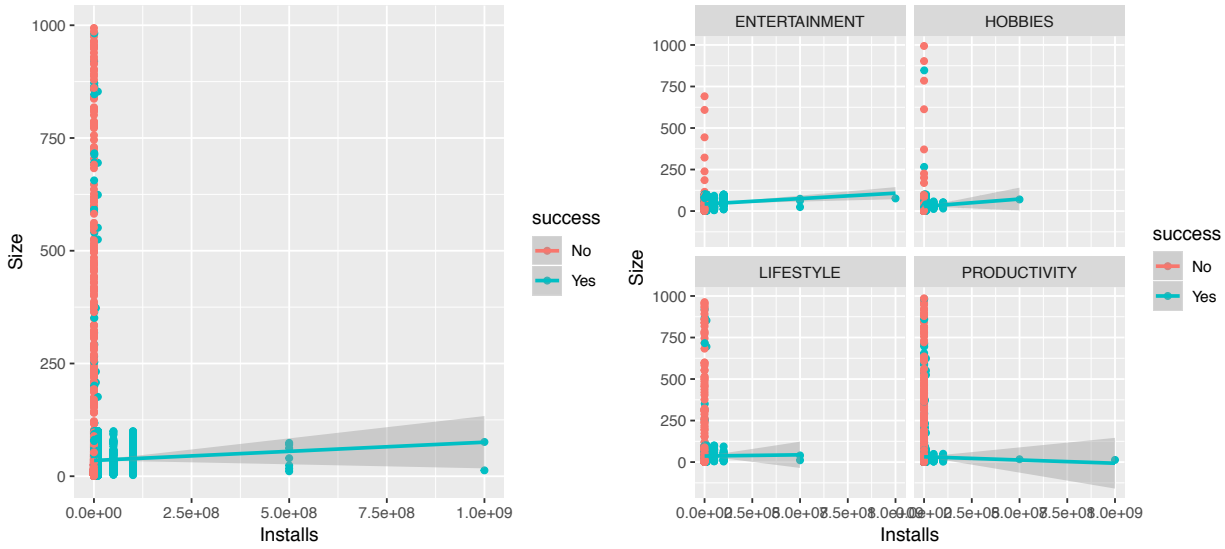
Since category is more significant, we look further into the successes by this variable

Rating vs. Installs by Category



We see that Rating has a positive correlation with an app's success for all categories

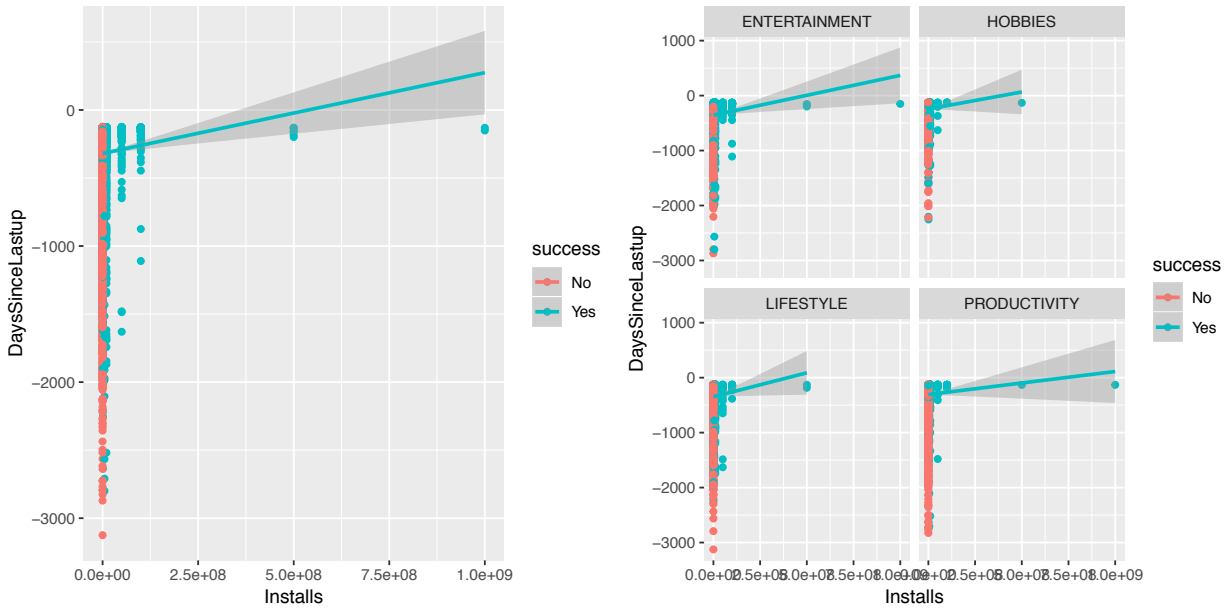
Size vs. Installs by Category



This plot shows that the larger the size of the application under "Entertainment", "Hobbies" and "Lifestyle" categories, the more successful they are. Interestingly, the relationship between the size of "Productivity" apps and its success do not behave the same. Apps with lighter sizes seems to do better for this category.

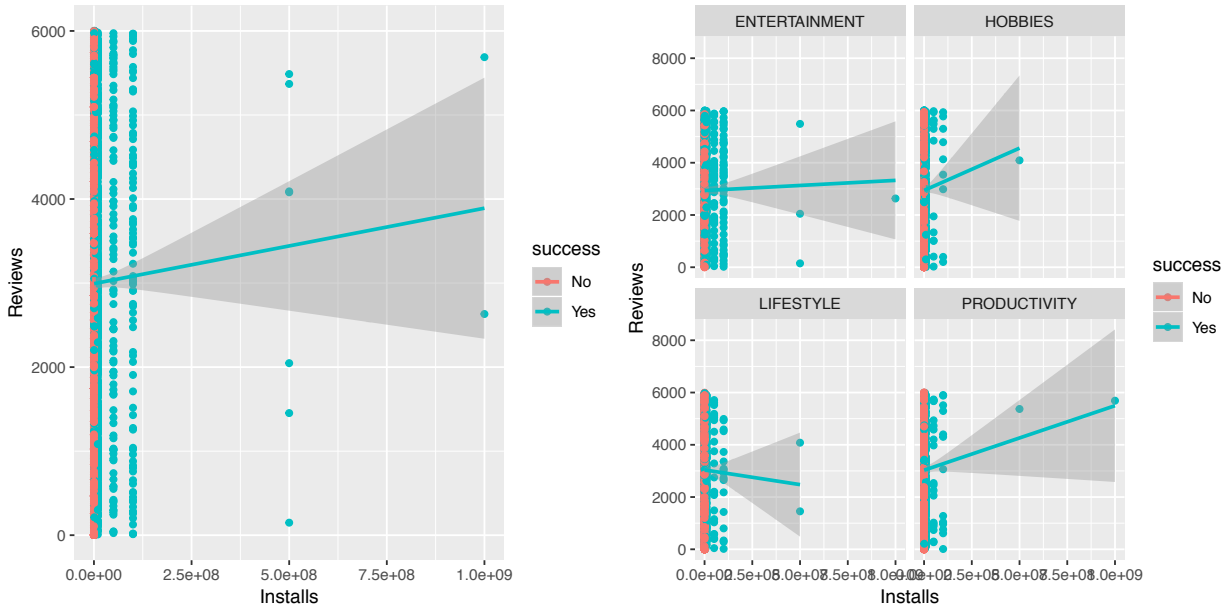
Days Since Last Updated vs. Installs by Category

(ignoring the bestfit line extending beyond 0 days since lastup(DaysSinceLastup>0), as this is not meaningful):



We see that there is a positive relationship between Days Since Last Updated and the success of apps across all categories and overall.

Reviews vs. Installs by Category



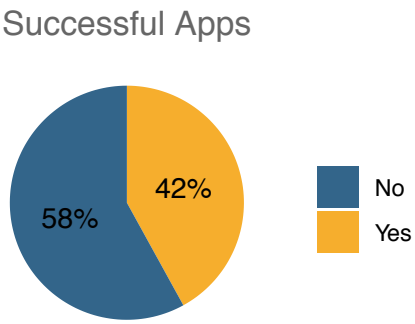
Reviews seems to be positively correlated with success for most of the categories. Surprisingly for 'Lifestyle', however, they are negatively correlated - the less reviews, the more installs an app under this category will have. This may mean that most reviews under the 'Lifestyle' category are negative which may signal the app's quality.

Final Dataset used in modelling

Our final cleaned-dataset used in modelling has 13 predictor-variables (Android.Ver has been removed through EDA) in 8 dimensions and 6738 obervations in total.

The 8 dimensions we used for our analysis are:

1. **Success** For our outcome variable, we transformed the 'installs' feature into a numeric value and then defined a new variable, success, as those apps having more than 500,000 app downloads. Apps are typically profitable with 500,000 downloads (Louis,2013). This variable was categorical for the classification tree and binary for the logistic regression model. 58% of the observations were classified as successful. NOTE: It is important to note that traffic is only one of the many factors that determine "success" (other important factors incl. app retention rates or in-app purchases) but due to limited data, we are assuming success is wholly due to traffic. This assumption is reasonable in the sense that ultimately, traffic is needed before other factors can be considered as significant in determining success.



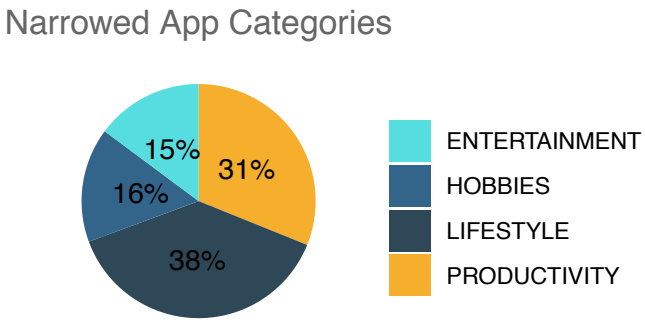
2. **Rating** The rating factor is the overall user rating of the app. Higher rated apps have been shown to

have more downloads (Lanza,2012) so we expected this variable to carry some weight in our result. For the rating variable, we removed all values above 5, since app scoring is on a 1-5 basis. According to our initial analysis, this is true.

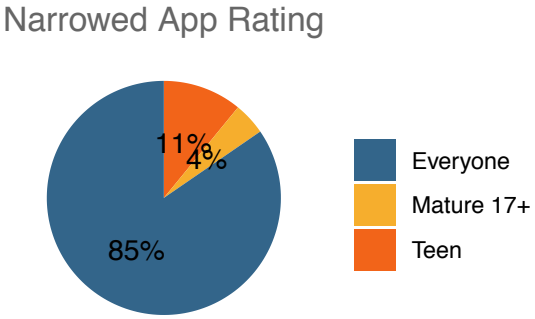
3. **Size** The "Size of App" factor captures various information on the app. Large apps might contain more features or better functionality. Thus, they might have better ratings. Larger apps could imply a higher probability to contain a bug and therein might have lower ratings (Zimmermann,2007). We converted the size factor into a numeric value and deleted all observations where size was specified as "vary with device". Size appears to influence number of downloads, more so for the entertainment category as we will see below.

4. **Category** For the category the app belongs to, we choose to recode the categories so that there were simply 4 categories: "Hobbies", "Entertainment", "Lifestyle", and "Productivity". Our grouping of the categories can be found in the Rhistory file. We coded these 4 categories as 4 mutually exclusive and collectively exhaustive binary variables. The category of an app might impact how other factors affect ratings and might influence user's expectation of an app (e.g., users might be more forgiving for a bug in a game app versus a financial app). We can see the distribution of the new narrowed categories:

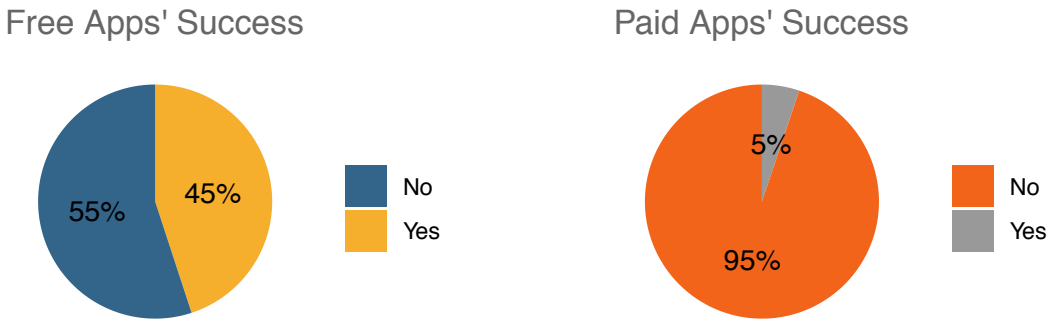
The two largest app categories are Lifestyle and Productivity, combined they account for 68% of apps in our dataset. Entertainment and Hobbies together constitutes the same proportion as productivity.



5. **Content Rating** For content rating we defined three age ratings: "Everyone", "Mature 17+" and "Teen". We recoded "Everyone 10+" and "Unrated" as "Everyone" and "Adults only 18+" as "Mature 17+" for simplicity. We coded these 3 ratings as 3 mutually exclusive and collectively exhaustive binary variables. Similarly, to "Category", rating might influence a user's expectation of an app.



6. **Price** For the price of an app, we coded the input at a numeric value. We would imagine this would be related to quality / willingness to pay, despite previous research which finds no correlation (Tian,2015).



The vast majority of apps are priced as free or for a very small fee (<$1). However, there are some outliers that charge >$300. These outlier apps could be classified as "Staus Apps" whereby customers purchase them to highlight their wealth e.g. VIP BLACK

With our definition of success, free apps tends to be more successful than paid by large. We expect price to be a significant variable that is negatively correlated to success in our model.

7. **Reviews** For the number of reviews, we simply coded this variable as a numeric value. There is a positive correlation between the number of reviews and the number of app downloads.
8. **Days Since Last Update** As the variable "Last Updated" could impact the number of installs via appearing in the 'newly updated tab' (thus attracting more traffic as there are more appstore users through time) in the google app store, we expect this variable to be significant. As the original "Last Updated" variable is a character class, we transformed it into a date and created a new variable, "DaysSinceLastup". This variable shows how many days ago the app was last updated (negative days) since 10/12/18, the day we downloaded the dataset. The analysis in previous sections show that there is a positive correlation between this and success.

Despite the relationships these variables have shown, some may prove to be more significant than others in determining success. Our model seeks to find out these most significant variables, thereby determining the characteristics of successful apps.

Modelling the data

Splitting the dataset into training and testing

Before creating any sort of classification model, we had to split the data into a training and a testing set. We did a 70-30 split where the training dataset had 70% of the original observations while the testing dataset had 30% of the dataset (4700 observations for training and 2038 observations for testing)

Classification modelling approaches are applied to the data inorder to find significant variables in determining the success of apps in the googleplay store

We will be using 2 main approaches:

1. Logistic Regression

2. Decision Trees

1. Logistic Regression

As our outcome, 'succcess' is binary ("Yes" or "No"), we use the logistic regression instead of the linear regression.

Modelling training data with stepwise

```
##
## Call:
## glm(formula = success.num ~ Price + Entertainment + DaysSinceLastup +
##     Rating + Productivity + Hobbies + Everyone + Reviews + Size,
##     family = binomial, data = trainData)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.895  -1.020  -0.594   1.149   3.276
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -1.566e+00  2.802e-01  -5.590 2.27e-08 ***
## Price           -7.811e-01  9.324e-02  -8.378  < 2e-16 ***
## Entertainment    1.070e+00  1.003e-01  10.673  < 2e-16 ***
## DaysSinceLastup  1.008e-03  9.376e-05  10.754  < 2e-16 ***
## Rating           4.043e-01  6.107e-02   6.620 3.60e-11 ***
## Productivity    -2.545e-01  7.859e-02  -3.239  0.00120 **
## Hobbies          4.026e-01  9.367e-02   4.298 1.72e-05 ***
## Everyone        -3.293e-01  9.250e-02  -3.560  0.00037 ***
## Reviews          5.129e-05  1.814e-05   2.827  0.00470 **
## Size             8.264e-04  3.455e-04   2.392  0.01675 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6397.3  on 4699  degrees of freedom
## Residual deviance: 5711.7  on 4690  degrees of freedom
## AIC: 5731.7
##
## Number of Fisher Scoring iterations: 9
```

This model has the minimum AIC according to stepwise (so is the best fitted model for stepwise). 9 out of the 16 variables are significant in determining an app's success: variables "Price", "Entertainment", "DaysSinceLastup" "Rating", "Productivity", "Hobbies", "Everyone", "Reviews" and "Size" are significant according to stepwise ('Size' much less significant than the rest). What is surprising here is that the 'Entertainment' category is the 2nd most significant variable for success, even more so than rating and number of reviews. This means, according to the data, ratings and reviews are less important than the app being under entertainment when consumers choose apps to download. The significant categories -"Entertainment" and "Hobbies" are mostly for leisure use. This may also signal how android users mostly use their device for entertainment / not-for-work as apps with the most downloads correlated strongly with these categories. DaysSinceLastup is also significant, which proves the fact that more recent apps gets more traffic, as we saw in the Exploratory Data Analysis of it's relationshop with the number of Installs.

Testing the model against testing data

We now test the model formulated from the training data against the testing data to observe its performance

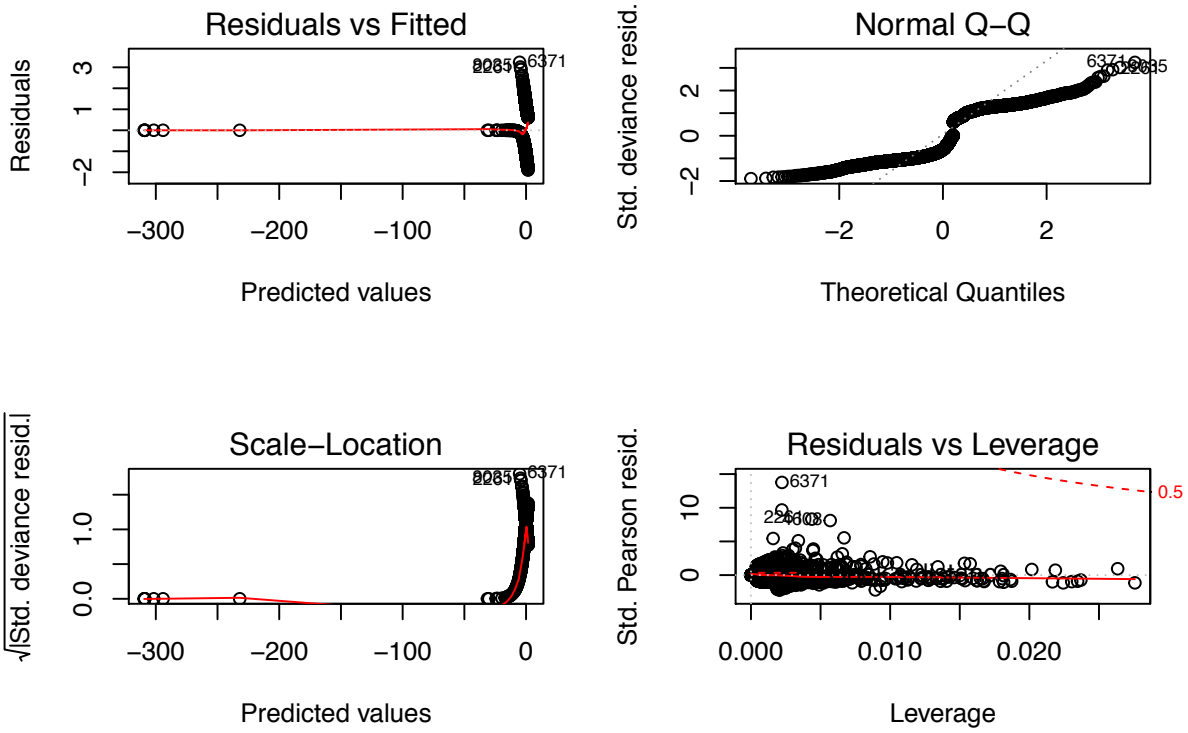
```
##          testData$success.num
## glm.pred   0   1
##       No  964 483
##       Yes 223 368
```

Where "1" is success in numerical form. This model correctly predicts 1332 successes out of 2038: The test data is predicted correctly $\frac{964+368}{2038} = 65.4\%$ of the time, yielding a

```
## [1] 0.3464181
```

=34.6% misclassification rate

Assessing the fit of the model



The Normal Q-Q graph plot shows that the residuals are not normally distributed (but are heavy-tailed), as the points do not form a straight line - but since the logistic regression does not go by the assumption of residuals being normally distributed, the model is still valid: this plot does not tell us anything. The other three plots do not provide much information for the model either.

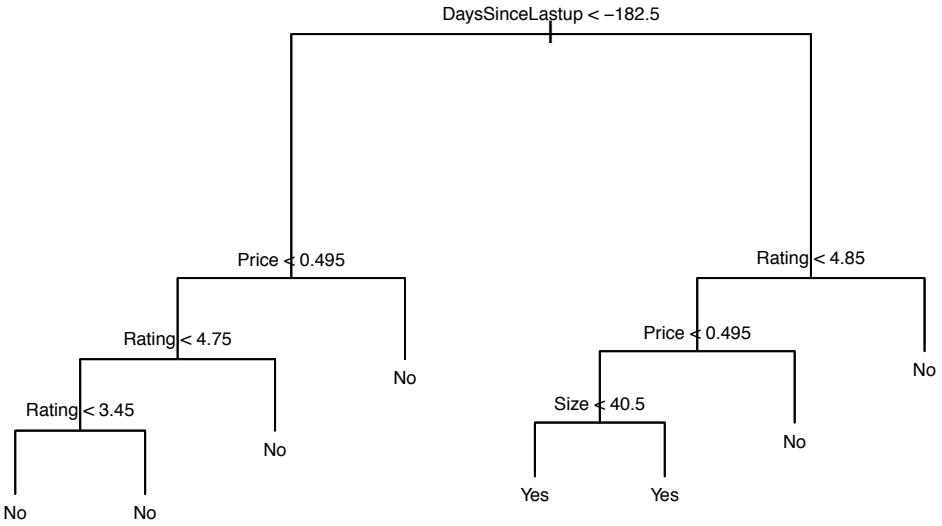
2. Decision tree

We then move towards creating a classification decision tree to be able to predict the success of mobile applications based on our selected independent variables.

```
##
## Classification tree:
## tree(formula = success ~ . - Installs - success.num, data = trainData)
## Variables actually used in tree construction:
```

```
## [1] "DaysSinceLastup" "Price"           "Rating"          "Size"
## Number of terminal nodes:  8
## Residual mean deviance:  1.148 = 5388 / 4692
## Misclassification error rate: 0.3181 = 1495 / 4700
```

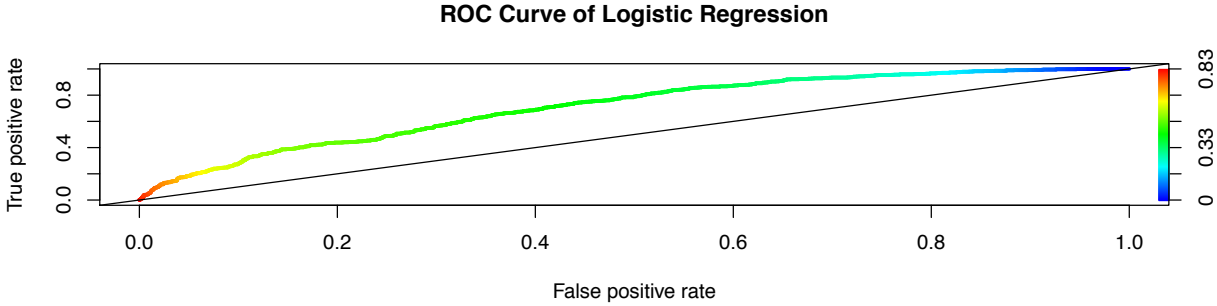
As can be seen in the above summary of the classification tree, the independent variables that were used include "DaysSinceLastup", "Price", "Rating" and "Size". Rating seems to be an obvious indepedent variable used since apps that have higher ratings should logically be more successful (be installed more). The same goes for Price where generally, one would think that the cheaper the app, the more times it will be installed. However, 'Size' and 'DaysSinceLastup' were not obvious choices yet were used in the tree construction.



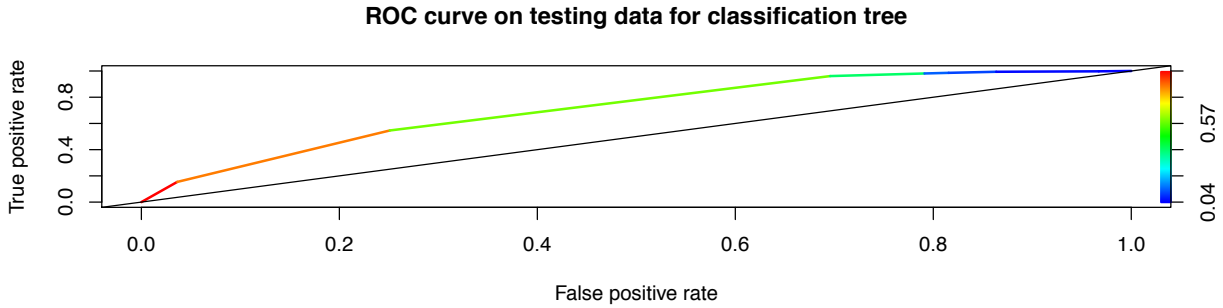
We will then proceed to construct the ROC curves for both classification approaches for a performance metric to use as a comparison between the two; testing the models on the testing data

Plotting ROC curves to compare the performance of Decision trees and Logistic Regression

Logistic Regression



Decision Tree



AUC Calculation

Logistic regression

AUC:

```
## [[1]]
## [1] 0.7090484
```

With a misclassification rate of 34.6% from earlier (of training model tested against test data)

Decision Tree

AUC:

```
## [[1]]
## [1] 0.7133696
```

```
##
## tree.pred  No Yes
##       No  889 386
##       Yes 298 465
```

and misclassification rate of

```
## [1] 0.3356232
```

Misclassification rate comparison: 33.6%(decision tree) $<$ 34.6%(logistic regression)

AUC comparison: 0.7133 (decision tree) $>$ 0.709 (logistic regression)

The decision tree has the lower misclassification rate and higher AUC. It is the better model according to both AUC and the misclassification rate. So we seek to improve the models further, starting with improving the classification tree.

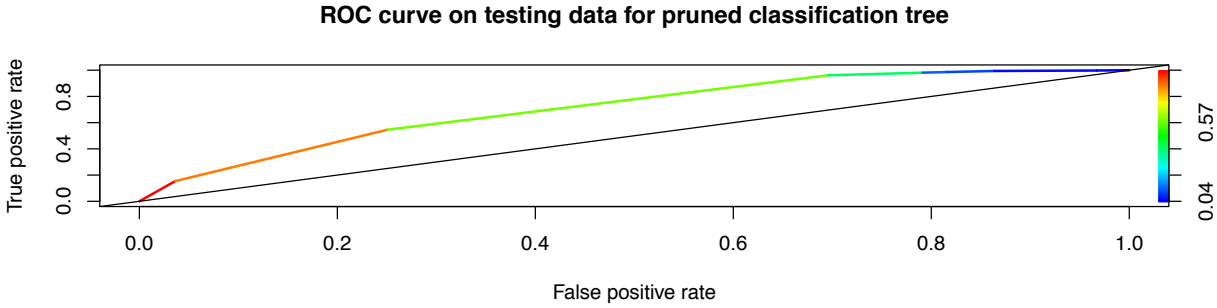
Pruning the Decision Tree

We believe that there could be a chance that our decision tree has overfitted the training data and we could thus improve the predictive abilities of our decision tree by pruning the tree.

However, we found that our pruned decision tree is identitcal to the unpruned decision tree with the same number of terminal nodes and the same variables used in the tree construction.

```
##
## Classification tree:
## tree(formula = success ~ . - Installs - success.num, data = trainData)
## Variables actually used in tree construction:
## [1] "DaysSinceLastup" "Price"           "Rating"          "Size"
## Number of terminal nodes:  8
## Residual mean deviance:  1.148 = 5388 / 4692
## Misclassification error rate: 0.3181 = 1495 / 4700
```

We then evaluate its performance through the ROC curve and calculate its misclassification error.




```
## [[1]]
## [1] 0.7133696
```

```
##
## tree.pred  No Yes
##       No  889 386
##       Yes 298 465
```

```
## [1] 0.3356232
```

We found that our pruned decision tree has the same AUC and misclassification error rate as our unpruned decision tree thus we can conclude that our original classification tree is already the most optimal with regards to avoiding overfitting.

We now attempt to model the significant variables from the pruned tree with a logistic regression to see if the model would improve

Modelling the 4 variables "Price", "Rating", "Size" and "DaysSinceLastup" from the tree, we find out that all are statistically significant and hence no variables are eliminated.

The logistic model with the 4 variables has a misclassification rate of

```
## [1] 0.3464181
```

And auc of

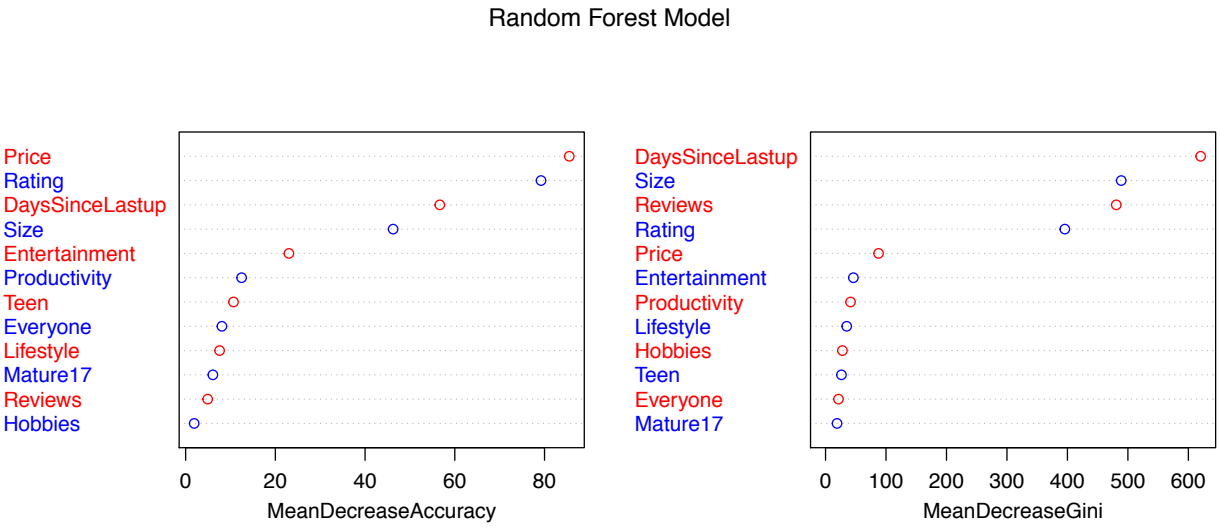
```
## [[1]]
## [1] 0.6469845
```

This signals overfitting as despite the misclassification rate remaining the same as the original logistic model (and <decision tree), the AUC is smaller than both the original pruned tree and the regression. It's AIC is also larger than the stepwise-model AIC which means this is not an improvement to the original model.

Finally, we seek to improve the classifiction further with the random forest model

After creating a classification decision tree, we feel that there is a chance that perhaps other models could be better at predicting the success of mobile apps. One possibility that we will be trying out is the Random Forest model.

We then use the varImpPlot function to establish which variables are more important in our random forest model. The first graph shows that if a variable is assigned values by random permutation by how much will the MSE increase, this is what is important to us. The Random Forest model reaffirms that Price, Rating, DaysSinceLastup and Size are the four most important variables. However, unlike the decision tree, the most important variable is not DaysSinceLastup but instead Price, followed by Rating, DaysSinceLastup and followed by Size.



We then calculate the ROC curve, performance score and misclassification rate of our Random Forest model to evaluate if it is better than our previous models.

With it's AUC being:


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

```
## [1] 0.7883134
```

And it's misclassification rate of

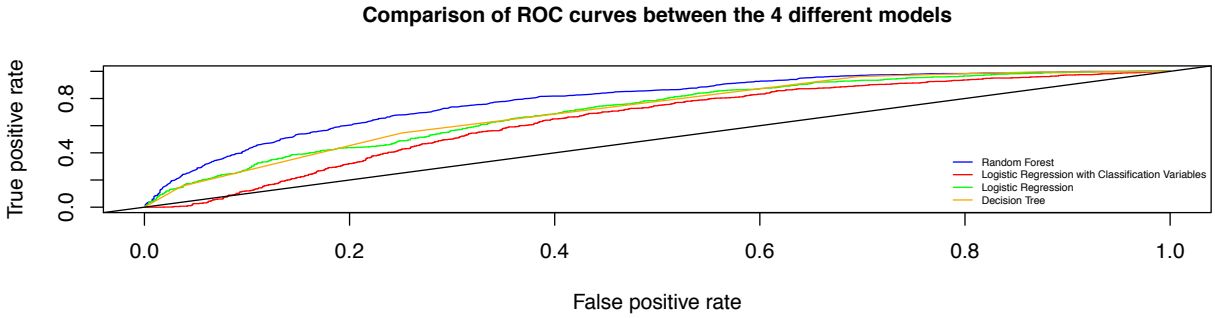
```
##
## rf.predict  No Yes
##        No  919 300
##        Yes 268 551
```

```
## [1] 0.2787046
```

The random forest model performs better than all previous models according to the AUC and misclassification rate. This can be seen from the below table which compares the misclassification rate and AUC performance score among the 4 models that we used.

A comparison of all the models

Model Type	Misclassification Error	AUC score
Logistic Regression	34.6	70.9
Decision Tree	33.6	71.3
Logistic Regression with Variables from Decision Tree	34.6	64.7
Random Forest	27.9	78.8



All models are above the abline, thus meaning they are all at least better than a guess.

Conclusion

From our study and use of the 4 different models, we can conclude that the random forest model is most effective in predicting the success of mobile applications. Our random forest model demonstrates that Price, Rating, DaysSinceLastup and Size are the most important variables used in the construction of the model.

This is important for developers who want to develop successful mobile applications. They should hence develop free applications with the aim of receiving high ratings in the Google Play Store. This could mean that developers trying to make money from their apps could stand to make their apps free but concentrate on in-app monetization opportunities instead. Making consumers pay for the app itself seems to be a major deterrent for users to download the app.

More interestingly, developers should also continuously and consistently update the app as we found that apps that were more recently updated tend to be more successful. We also found that overall, apps that were larger in size were more successful (although less significant than the other three variables). We think that size could be a good proxy for complexity and sophistication, hence apps that are more complex and developed tend to be more popular.

While we believe that we have made a valuable contribution to better understanding mobile applications and consumer behaviour, we also acknowledge that there are some potential limitations to our study. Firstly, we have used the number of downloads as a proxy for success. However, it could be the case that many users may download an app without actually using it or it could be that an app may have a lot of downloads but is simply a fad that fades quickly.

Furthermore, in the Google Play Store, the number of downloads for individual apps are given in a range (e.g 100,000 - 500,000). This brings up several potential problems. Firstly, there is quite a bit of difference between apps that fall in the lower end of the range and the upper end of the range. This difference could even be bigger than the difference between an app in another range. For example, an app with 90,000 downloads would not be considered to be a part of the 100,000 to 500,000 range yet it is more similar to an app with 120,000 downloads then the 120,000 download app with the 480,000 download app. However, despite these concerns, we still believe that downloads are the best possible proxy for success. It is a metric that is easily found and is consistent among all apps. Other possible metrics would be significantly harder to find and might not be consistent among all apps.

Bibliography

"App Annie." 2019. Accessed February 11. http://go.appannie.com/report-app-economy-forecast-part-two.

"App Promo White Paper (June 2013)." n.d. www.app-promo.com.

Bavota, Gabriele, Mario Linares-Vásquez, Carlos Eduardo Bernal-Cárdenas, Massimiliano Di Penta, Rocco Oliveto, and Denys Poshyvanyk. 2015. "The impact of API change- and fault-proneness on the user ratings of android apps." *IEEE Transactions on Software Engineering* 41 (4): 384–407. doi:10.1109/TSE.2014.2367027.

Elizalde, Ignacio, and Jorge Ancheyta. 2014. "Modeling catalyst deactivation during hydrocracking of atmospheric residue by using the continuous kinetic lumping model." *Fuel Processing Technology* 123 (2): 114–21. doi:10.1080/10864415.2016.1087823.

Ghose, Anindya, and Sang Pil Han. 2014. "Estimating Demand for Mobile Applications in the New Economy." *Management Science* 60 (6): 1470–88. doi:10.1287/mnsc.2014.1945.

"Global developers per app store 2017 | Statistic." 2019. Accessed February 11. https://www.statista.com/statistics/276437/developers-per-appstore/.

"Google Play | Android Developers." 2019. Accessed February 11. https://developer.android.com/distribute/.

Guerrouj, Latifa, Shams Azad, and Peter C. Rigby. 2015. "The influence of App churn on App success and StackOverflow discussions." *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*. IEEE, 321–30. doi:10.1109/SANER.2015.7081842.

Lanza, Michele., Massimiliano. Di Penta, Tao Xie, Institute of Electrical and Electronics Engineers., and Switzerland) International Conference on Software Engineering (34th : 2012 : Zurich. 2012. "App store paper." *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories*, 254.

Lee, Gunwoong, and T. S. Raghu. 2014. "Determinants of Mobile Apps' Success: Evidence from the App Store Market." *Journal of Management Information Systems* 31 (2): 133–70. doi:10.2753/MIS0742-1222310206.

Lee, Sang M., Na Rang Kim, and Soon Goo Hong. 2017. "Key success factors for mobile app platform activation." *Service Business* 11 (1). Springer Berlin Heidelberg: 207–27. doi:10.1007/s11628-016-0329-y.

Tian, Yuan, Meiyappan Nagappan, David Lo, and Ahmed E. Hassan. 2015. "What are the characteristics of high-rated apps? A case study on free Android Applications." *2015 IEEE 31st International Conference on Software Maintenance and Evolution, ICSME 2015 - Proceedings*, 301–10. doi:10.1109/ICSM.2015.7332476.

Tristan Louis. 2013. "How Much Do Average Apps Make?" https://www.forbes.com/sites/tristanlouis/2013/08/10/how-much-do-average-apps-make/{\#}a774ae346c4b.

Zimmermann, Thomas, Rahul Premraj, and Andreas Zeller. 2007. "Predicting defects for eclipse." *Proceedings - ICSE 2007 Workshops: Third International Workshop on Predictor Models in*

Software Engineering, PROMISE'07. doi:10.1109/PROMISE.2007.10.