DATA 621 Final: Predicting A Mobile Application's Success

David Quarshie

5/23/2019

Abstract

In an effort to see what developers can focus on to get more downloads of their mobile application, this project will use various modeling techniques to predict an application's success. These models will we built using data about apps on Apple's and Google's stores. This will provide a methodology to scientifically predict how well an app will do in either store when certain metrics are improved.

Keywords: Apps, Apple, Google, Poisson, Quasi

Introduction

In 2014 Dong Nguyen revealed that his game Flappy Bird was earning an average of $50,000 a day from in-app ads. At that time, Flappy Bird, an addictive and tough game, had been number 1 on the Apple App Store and Google Play Store for almost a month and was gaining more downloads daily. In fact, by January 2014 the app had 50 million downloads, 68,000 reviews, and held the number 1 spot for most downloaded free game in 53 countries.

With millions of apps available for download today, many app developers could only dream of the success that Flappy Bird saw. So, did Nguyen know how to make an app so successful, and is there any way for us use app statistics to predict the success of an app? Kaggle has a couple of datasets with mobile application statistics that can possibly help us answer these questions. The datasets contain information around Apple apps on the App Store and Google apps on the Play Store; information centered around ratings, app size, installs, and price. Our idea is to take these datasets to see if we can use several regression methods to predict the successfulness of an app.

Methodology

Apple

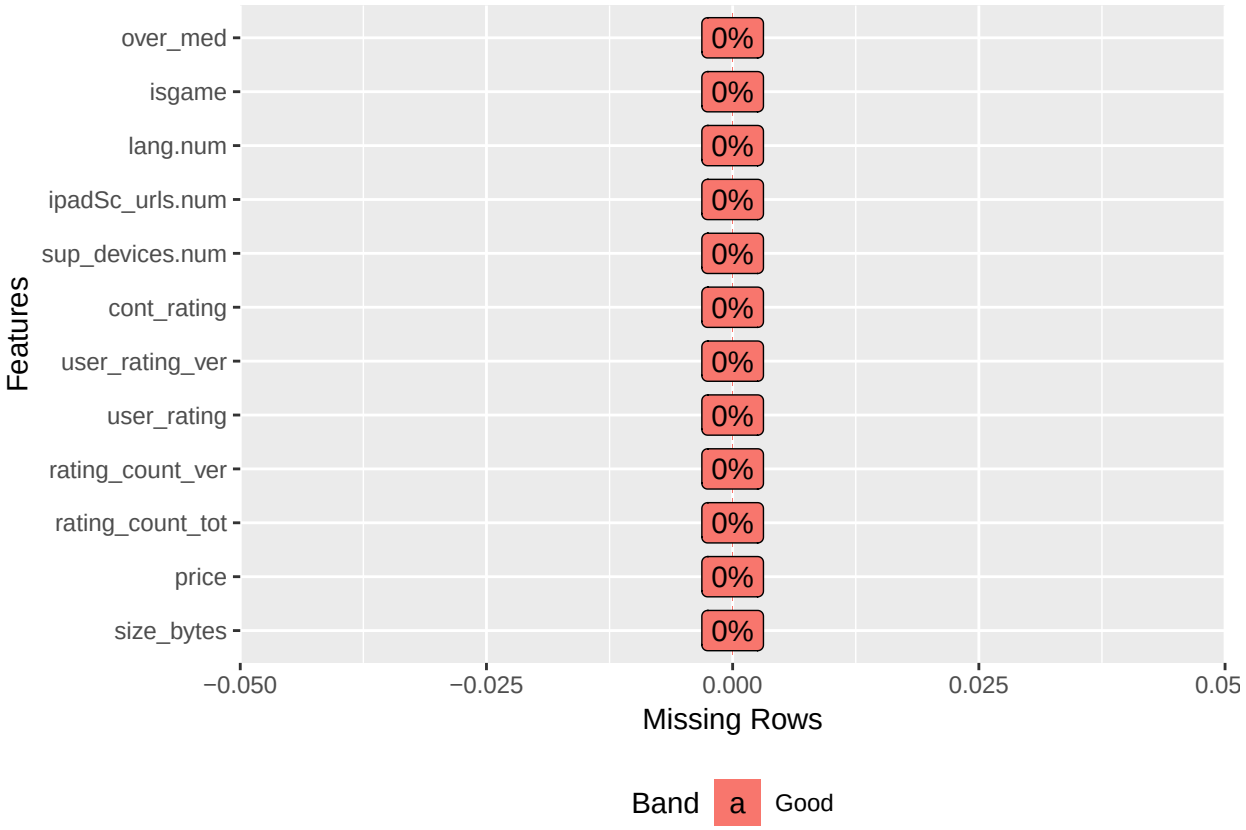
Apple Store Data

For this project, data from the Apple store on 7,197 apps was sourced from Kaggle at this site: https://www.kaggle.com/ramamet4/app-store-apple-data-set-10k-apps. The data content on these apps revolved around the apps' size, price, ratings, genre, and number of supporting devices. However, there is no data around how many times the app was downloaded. As a workaround, this study used the total number of ratings as a measure of how many times the app was downloaded and therefore how successful it is.

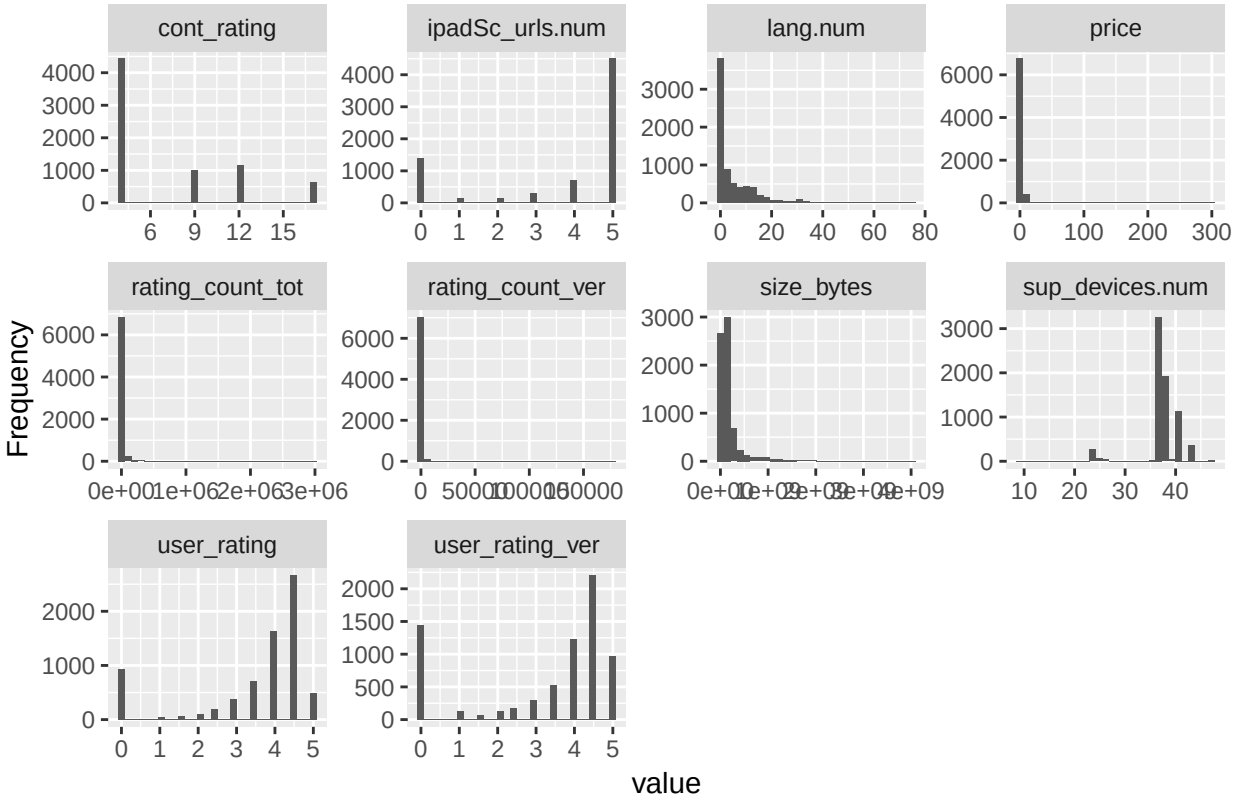
Data Management

Thankfully none of the data in the Apple Store dataset was missing. Two new fields were added to the dataset, one to distinguish if the app was a game or not and another to see if the total ratings count for the

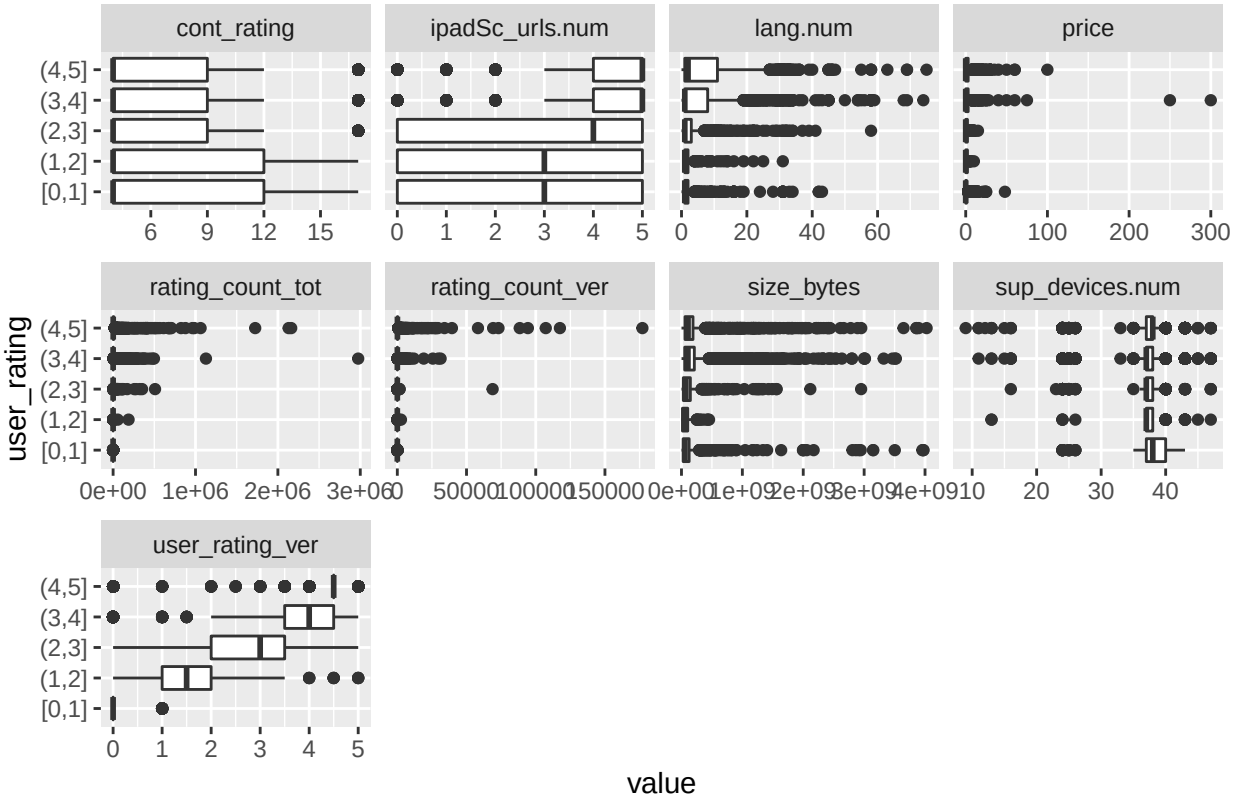
app was over or under the median of the dataset. For the content rating field the '+' symbol was removed and the field was transformed into an integer. Here's a summary of the final dataset.



Histogram of Fields



Boxplot of Fields




```
## NULL
```

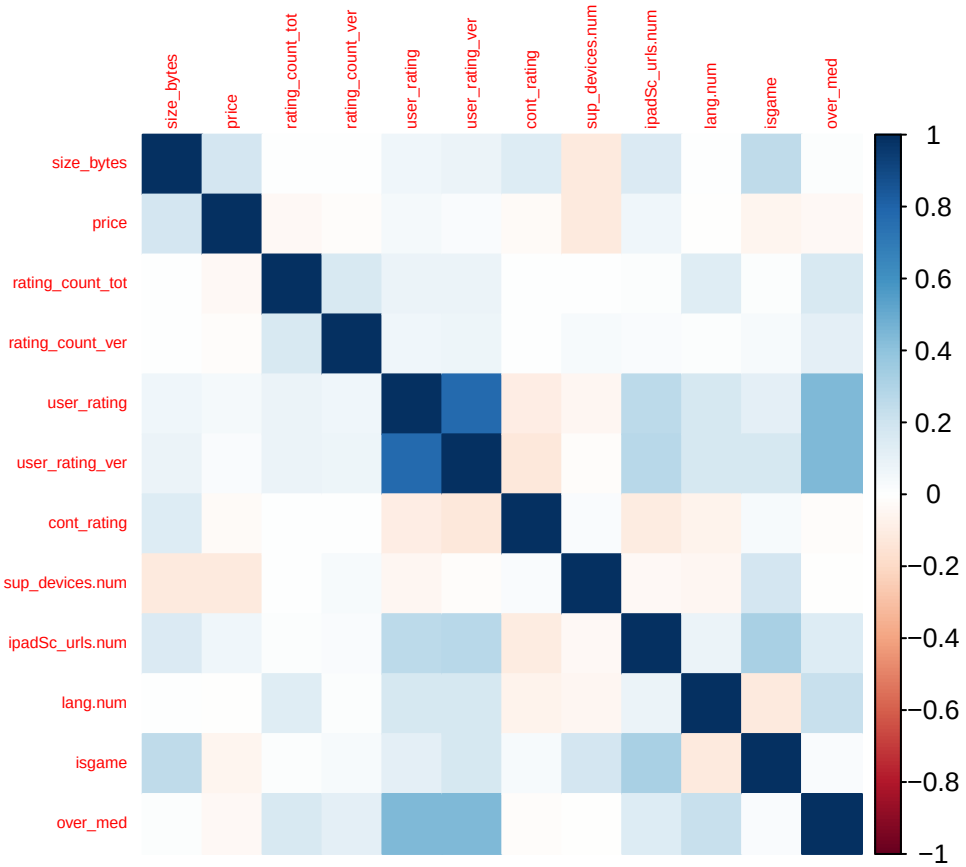
Train – Test Split

Given that the Apple Store dataset is not conveniently split to do model training, it was split by randomly choosing fifty percent of the data to be in the train set and the remaining data to be in the testing set.

Results

Correlation

Before beginning the data analysis, the correlation plot between the metrics was analyzed to see if there was anything worth noting.



Ratings Total Summary

In order to gauge how well the forthcoming models did in predicting the total number of ratings, here's the summary of ratings from the train dataset.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0      26     308   12124    2738 1061624
```

Poisson Model

The first model ran is a Poisson Model.

```
##
## Call:
## glm(formula = rating_count_tot ~ ., family = "poisson", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -744.94  -122.50    -9.25     2.85  2088.04
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       3.059e+00  4.270e-03   716.3   <2e-16 ***
## size_bytes        2.904e-10  5.259e-13   552.2   <2e-16 ***
## price            -3.425e-01  1.566e-04 -2187.6   <2e-16 ***
```

```
## rating_count_ver  1.645e-05  6.698e-09  2456.5   <2e-16 ***
## user_rating       5.140e-02  3.837e-04   133.9   <2e-16 ***
## user_rating_ver   1.881e-01  2.353e-04   799.6   <2e-16 ***
## cont_rating       1.494e-02  4.176e-05   357.8   <2e-16 ***
## sup_devices.num   1.529e-02  6.307e-05   242.4   <2e-16 ***
## ipadSc_urls.num  -4.476e-02  1.030e-04  -434.3   <2e-16 ***
## lang.num          3.303e-02  1.268e-05  2605.6   <2e-16 ***
## isgame            6.433e-02  4.194e-04   153.4   <2e-16 ***
## over_med          5.365e+00  3.376e-03  1589.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 163175599  on 2878  degrees of freedom
## Residual deviance:  96237216  on 2867  degrees of freedom
## AIC: 96257617
##
## Number of Fisher Scoring iterations: 7
```

```
##   TARGET_FLAG
##  Min.   :     0.00
##  1st Qu.:    49.22
##  Median :   203.49
##  Mean   : 11913.71
##  3rd Qu.: 22609.33
##  Max.   :286323.69
```

Running the Poisson model gave the ouput that every field was statistically significant, so they were all kept for the predictions. The results from this model resembles the train's results.

Quasi

```
##
## Call:
## glm(formula = rating_count_tot ~ ., family = "quasi", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -175165   -15450    -2500     1995  1015817
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -5.665e+03  1.169e+04  -0.485  0.62804
## size_bytes        3.158e-06  3.153e-06   1.002  0.31664
## price            -7.739e+02  2.725e+02  -2.840  0.00455 **
## rating_count_ver  2.358e+00  2.106e-01  11.194  < 2e-16 ***
## user_rating      -4.473e+02  1.040e+03  -0.430  0.66699
## user_rating_ver   6.326e+02  8.800e+02   0.719  0.47225
## cont_rating       2.497e+02  2.431e+02   1.027  0.30441
## sup_devices.num   5.448e+01  2.930e+02   0.186  0.85254
## ipadSc_urls.num  -4.963e+02  5.683e+02  -0.873  0.38254
## lang.num          9.832e+02  1.362e+02   7.220 6.64e-13 ***
## isgame            1.287e+03  2.338e+03   0.550  0.58212
```

```
## over_med          1.737e+04  2.384e+03   7.283 4.19e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasi family taken to be 3015279193)
##
##     Null deviance: 9.6284e+12  on 2878  degrees of freedom
## Residual deviance: 8.6448e+12  on 2867  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 2
```

Running the Quasi model showed that some fields were not significant so they were taken out.

Quasi Reduced

```
##
## Call:
## glm(formula = rating_count_tot ~ rating_count_ver + lang.num +
##     over_med, family = "quasi", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -177437   -15217    -1393     2439  1017035
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -3390.4937  1521.2070  -2.229   0.0259 *
## rating_count_ver     2.3783     0.2104  11.304  < 2e-16 ***
## lang.num           951.7282   133.4447   7.132 1.25e-12 ***
## over_med         17968.6402  2121.2549   8.471  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasi family taken to be 3020300378)
##
##     Null deviance: 9.6284e+12  on 2878  degrees of freedom
## Residual deviance: 8.6834e+12  on 2875  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 2
```

```
##   TARGET_FLAG
##  Min.   : -3390
##  1st Qu.: -2327
##  Median : 15535
##  Mean   : 11695
##  3rd Qu.: 19475
##  Max.   :239838
```

After removing the non significant fields from the model and running the reduced Quasi model the results do not resemble the summary from the training dataset.

Conclusion

The Poisson model gave the best results in determining a mobile application's success but it also kept all of the variables from the dataset.

Appendix

Code for this project can be found her: https://github.com/dquarshie89/DATA-621-FINAL_PROJECT/blob/master/AppPrediction.Rmd