ORIE 4741 Midterm Report

mmd272, mp668, rt359

November 2020

1 Introduction

In today's digital age, it is becoming increasingly difficult to create successful mobile applications. Currently there are over three million apps on the Google Play Store alone, creating a saturated and competitive market. We want to investigate if there are certain factors (i.e. category, memory size, price) that developers should focus on to achieve high ratings on the Google Play Store. We focus on ratings rather than number of downloads since often the number of downloads is strongly correlated with the number of marketing dollars spent. For our findings to be meaningful, our findings must be applicable to both a large, high-budget developer and a small, upstart developer.

The data set we are investigating consists of data from nearly 10,000 apps from the Google Play Store and includes features such as category, price, and user reviews. We plan to use the data to predict the rating of a given app.

1.1 Data Cleaning

The googleplaystore.csv file has 10841 rows and 13 columns. Since we are using *Rating* as the target, we identified all rows with missing *Rating* values and set the rating equal to the mean of all ratings in the data set.

For the rest of the 12 columns, we performed the following cleaning measures:

App	
Category	We one-hot-encoded *Category* values.
Reviews	
Size	We removed the "M" and "k" from every row and replace with the corresponding number.
Installs	We removed the "+" from the end of the String *Installs* values and remove the commas.
Type	We discarded rows in which *Type* is missing and replaced rows with value 0 to Free. We also one-hot-encoded *Type* values.
Price	We removed the "$" from the *Price* values.
ContentRating	We replaced missing *ContentRating* values with "Unrated." We also one-hot-encoded *ContentRating* values.
Genres	We many-hot-encoded *Genres* values.
LastUpdated	
CurrentVer	
AndroidVer	

1.2 Categorizing Data Types

- Real Valued Data: *Reviews, Size, Installs, Price*
- Categorical Data: *Category, Type, ContentRating, Genres*
- Text Data: We plan to analyze the file googleplaystore_user_reviews.csv, which contains the 100 most relevant user reviews for each app. Each review has been pre-processed and attributed with the following 3 features, which we plan to explore more in depth: *Sentiment, Sentiment Polarity, Sentiment Subjectivity.*

2 Exploratory Data Analysis

2.1 Data Features

As previously mentioned, each data point in our sample has 13 features. The first group of features are real valued features and include: *Reviews, Size, Installs.* The minimum and maximum number of reviews are 0 and 78 million. Most of the applications have less than one million reviews, with the mean being about 444 thousand. It is likely only the well-known applications have more than one million reviews. The application sizes range from 0 to 100 MB, and its distribution is right-skewed, with the mean being 18 MB. The number of installs also follows a distribution that is right-skewed. The minimum and maximum number of installs are 0 and 1 billion, while the mean is 15 million installs.

The next group of features are the categorical features including: *Category, Type, Content Rating.* There exists 33 unique categories of applications in our training set. As seen in Figure 1, Family, Game and Tools applications have the highest market share, with Family having a large lead. The type of the app specifies whether it is free or not. The majority of the apps are free with 92.5% being free and only 7.5% being paid for. The content rating of the application is the intended audience. The ratings and percentage of apps that falls in each are: Everyone: 80%, Teen: 11%, Everyone 10+: 4%, Mature 17+: 4.5% and Adults only 18+ and Unrated both only have 2 observations. In addition to categorical data, there is also set data contained in the column *Genres.* There exists 53 unique genres of applications in our training set. The maximum number of genres in the set for an app is 2, with the majority, 92%, only having 1.

The final feature we analyzed is *Rating*, which is treated as the label for each observation. The mean rating for the apps is 4.19, and the minimum and maximum are 1.0 and 5.0 respectively. Figure 2 displays the density of ratings for the apps.

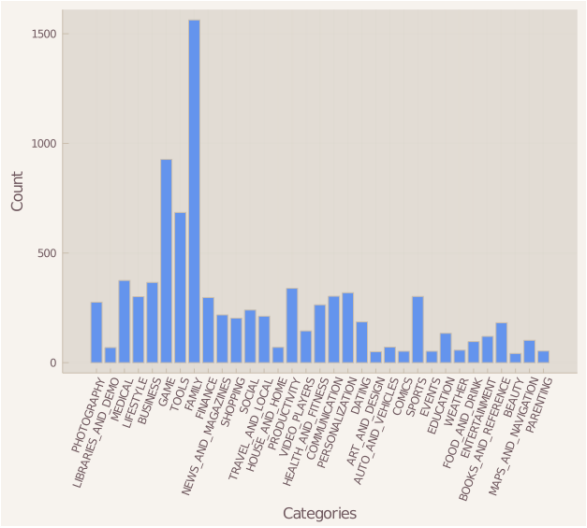

Figure 1: App Category Count

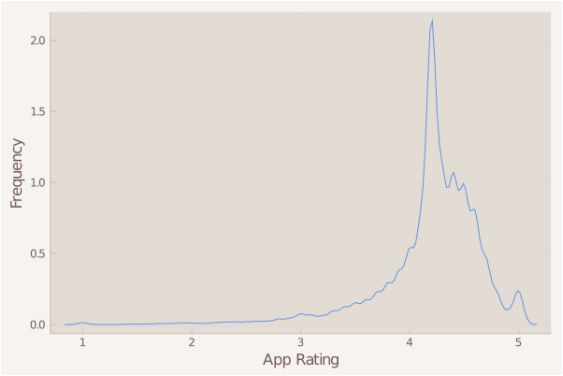

Figure 2: App Rating Frequency

The remaining columns include: *App, LastUpdated, CurrentVer, AndroidVer*. These features should not provide anymore insight into the success of an application than the others already mentioned. *App* is simply the name of the app and cannot be incorporated into the model unless natural language processing is done. We will consider this for our final report. *CurrentVer* and *AndroidVer* are the version numbers of the applications. These features were not included because they are dependent on other factors such as the app developers' strategies. It is possible that one app that is on version 10 has the same number of changes made to app on version 2 if the developers chose to release a new version for every individual change rather than grouping them. *LastUpdated* is the date of the last update made to the app and therefore was left out due to similar reasoning. However, if the error in future models is high, these features will be further explored and considered.

2.2 Variable Correlation

As seen in Figure 3, the largest positive correlation exists between the number of *Reviews* and the number of *Installs*. This means that these variables tend to move in the same direction. If an application has a large number of reviews, it tends to have more downloads. On the other hand, if an application has a large number of downloads, customers tend to leave a review. This signifies that app owners should encourage reviews in order to increase the number of downloads, and potentially the popularity of the app. Another important observation is that the *Reviews* variable has the largest positive correlation with *Rating*. As the number of reviews of an app increases, the rating of the app tends to increase. This will also help app owners increase their app's success in the Google Play Store. The *Size* of the app and the number of *Installs* for an app also display a positive correlation with *Rating*.

Figure 3: Real-Valued Feature Correlation

3 Least Squares Regression

3.1 Real-Valued Features

Least Squares Regression is a linear estimation of the best fit for a set of data points. It works by minimizing the sum of the offsets or residual points from the plotted curve. We used Least Squares Regression to predict the behavior of our dependent variable *Ratings* based on our other data columns, the independent variables.

After the data cleaning and classifying steps mentioned above we created train and test data sets splitting our data 80-20 and ran our first regression on *Ratings*. The results from plotting and getting the train and test Mean Squares errors are seen in Table 1. The data points are all clustered together with predicted ratings always being slightly above 4 and true ratings ranging from 1 to 5 with a concentration between 3.5 and 5. Our train and test errors are very similar and small.

	Train MSE	Test MSE
Real-Valued	0.22662	0.23330
+ One-Hot	0.22012	0.23092
+ Many-Hot	0.21808	0.22874

Table 1: Training and Testing Error

3.2 Real-Valued Features

We then added one hot encoding for all category type columns *(Category, Type and ContentRating)*. The percent reduction from this change is $\approx 3\%$ for both training and testing error. Finally, we added many hot encoding for our column *Genres* to create a vector for every entry. This further reduced our errors by 1%.

3.3 Evaluate Fit

Our data for *Ratings* ranges from 1 to 5 points which means that our maximum error would be 16. Our MSE for training

and test data sets are 0.22 which can be justified by the fact that only 360 data points have errors above 1.

Our model does not follow the trend of the data and we can therefore reach the preliminary conclusion that it would not make good predictions for new inputs.

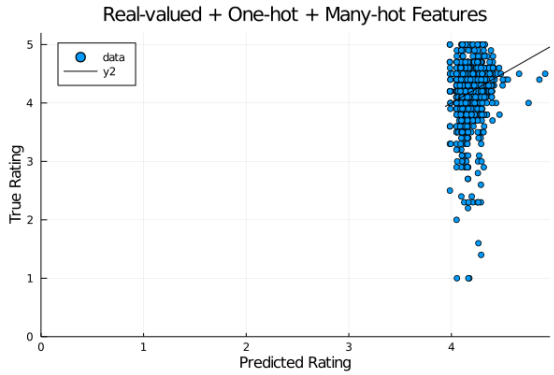

Figure 4: Final Least Squares Regression

4 Next Steps

We have made some preliminary models to predict the rating of an application using least squares regression. We tested the effectiveness of our model by splitting on our data into training and testing datasets and calculating the training and testing error. In the future, we plan to continue this strategy and incorporate k-fold cross validation. We will continue to use empirical risk minimization to build future models. However, we will consider the following loss functions to replace square loss: hinge, logistic, absolute. We will also experiment with the following regularizers: L_2 norm, L_1 norm and L_0 norm. In addition, we will explore our second dataset, which can be found in googleplaystore_user_reviews.csv. This dataset includes sentiment analysis of text based user reviews. We can match these data points to the same app in our original dataset and see the affect sentiment has on the app ratings.