Analysis of Google Play Store Data set and predict the popularity of an app on Google Play Store

Rimsha Maredia
Texas A&M University
College Stattion, Texas
rimsha.maredia@tamu.edu

Abstract

The google play store is one of the largest and most popular Android app stores. It has an enormous amount of data that can be used to make an optimal model. We have used a raw data set of Google Play Store from the Kaggle website. This data set contains 13 different features that can be used for predicting whether an app will be successful or not using different features. This data set is scraped from the Google Play Store. This journal talks about different classifier models that we used for prediction purposes and finding which one gives the highest accuracy. This journal also gives detailed information on feature extraction and the complete Data visualization done on this data set. Our project code can be found at https://github.com/Rimshamaredia/CSCE-421-Project.

1. Introduction

Mobile applications are one of the fastest-growing segments of downloadable software application markets. Out of all of the markets we choose Google Play store due to its increasing popularity and recent fast growth [10]. One of the main reasons for this popularity is the fact that about 81% of the apps are free of cost [3]. The market has increased to over 845900 Apps and 226,500 unique sellers in April 2013 [2]. This rapid market has, in turn, led to over 500 million users downloading around 40 billion Apps all over the world [2]. Developers and users play key roles in determining the impact that market interactions have on future technology. However, the lack of a clear understanding of the inner working and dynamic of popular app markets impacts both the developers and users. In this article, we seek to shed light on the dynamics of the Google Play Store and how we can use different features from this data set for prediction purposes.

In this article, we will provide a longitudinal study of Google Play app metadata which will give unique information that is not available through the standard approach of capturing a single app snapshot. Using feature extraction from a longitudinal app analysis will be used to find whether an app will be successful or not. Our Analysis is divided into four phases: data extraction, data cleaning, data visualization, and applying different models, and it is depicted in figure 8. First, we collect the data from the Kaggle website. In the next step, we try to do data cleaning on the data set to reduce the error percentage. After the data set is ready, we try to analyze the data set using different plots and remove the stuff not needed from the data set. The last step includes using different classification algorithms on the data set to see which one gives the highest percentage of accuracy. Finally, we narrate the analysis results to provide a clear vision of the relationship among the areas of interest.

We include a detailed discussion of the applicability and future research directions in the last section called Conclusion and future work.

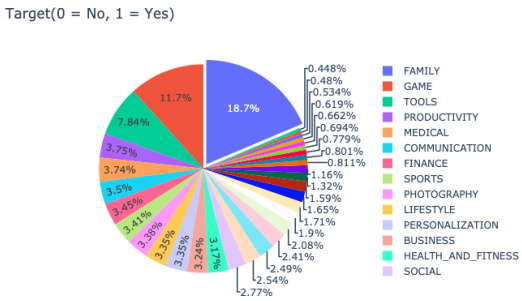

Figure 1. Number of apps available for download by category.

2. Analysis Methodology

Our analysis approach is divided into three phases: data extraction, data cleansing, and visualization, and data modeling. In the first step, we collected the raw data from Kag-

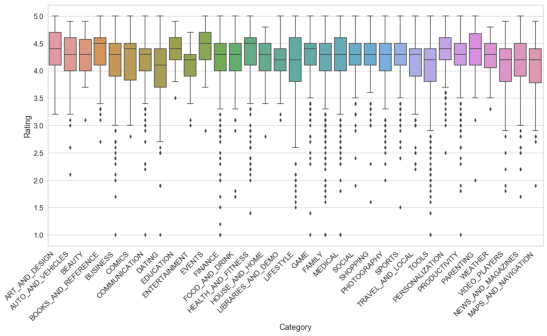

Figure 2. Box-plot to find average ratings.

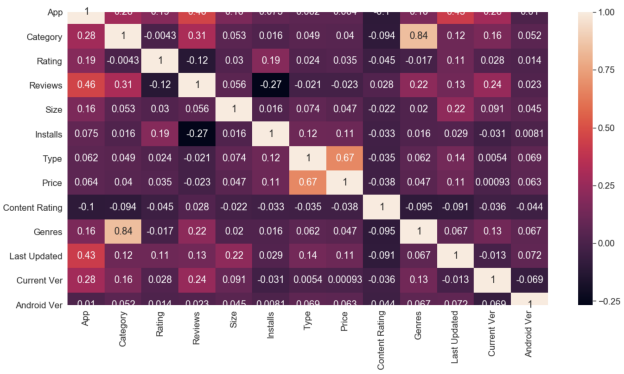

Figure 3. Correlation between installs and other parameters

gle. Then we did basic data cleaning and data visualization. After visualizing the data set, we removed some unnecessary features and made it ready for data modeling. Nest we conduct data modeling by using three models: Gaussian Naive Bayes Model, K-nearest neighbor model, and Decision Tree model.

2.1. Data Visualization

In this data set there are various features that can be used to analyse the data set. In this section we will be analysing different features to find which feature determines whether an app will be successful or not. The top five genres of the google play store include Tools, Entertainment, Education, Business and Medical. From figure 1 we can see that the most number of apps in Google Play store belongs to the categories of Family and game. This shows that apps that belongs to gaming and Family category are more common and apps in this category have high chances of being successful. There are only 8.26% of paid apps in Google play store. The majority of paid apps cost about $1. So the popularity doesn't depends on this factor.

While looking at the popularity feature, in Figure 2 we can see that the average rating is quite high, around 4.2 from 5.0. Most rating are in the range of (4.2,4.5). To further define which category are the highest rated, we will only look at the data for each category that have more than or equal to 4.0 in rating. This will help us in making more optimal model. To analyze the apps that would produce the most revenue, we will look at the correlation table between installs and other parameters. From figure 3 we can see that installs and reviews have the strongest inverse correlation. This is reasonable because popular apps tends to get more number of reviews.There is no correlation found between installs and other features like size,rating,number of installs and price. There is no correlation between rating and price also. Since installs parameter is independent and not correlated to any other parameters, we must only use installs to show the popularity of an app. For our analysis we will be only using number of installs for classifying whether the

Algorithm	Accuracy %
Decision Tree	95.32%
K-nearest neighbor	90.59%
Gaussian Naive Bayes	88.45%
Logistic Regression	89.86%

Table 1. Results.

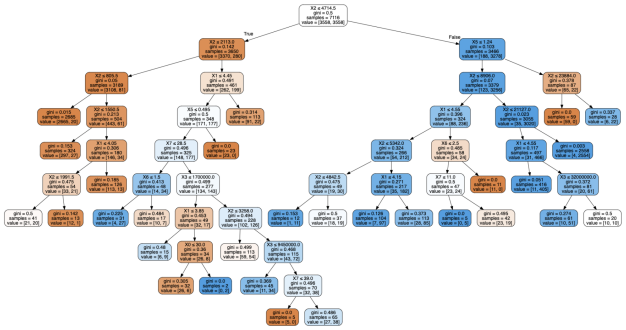

Figure 4. Decision tree chart

app is successful or not. If the number of installs for a particular app is greater than 1000, it will go under the class of successful and if it is less than 1000 we will classify it as unsuccessful.

2.2. Classification Models

2.2.1 Decision Tree Model

Decision tree is similar to the human decision-making process and so that it is easy to understand. Shannon invented the concept of entropy which measures the impurity if the input set[5].There are many benefits of decision tree.It can easily capture Non-linear pattern. The decision tree has no assumptions about distribution because of the non-parametric nature of the algorithm. To avoid biased tree, we balanced the data set before creating the model. Decision tree can handle uncertainty in both building and classification procedures.

The Figure 5 shows the decision tree for our model. Each

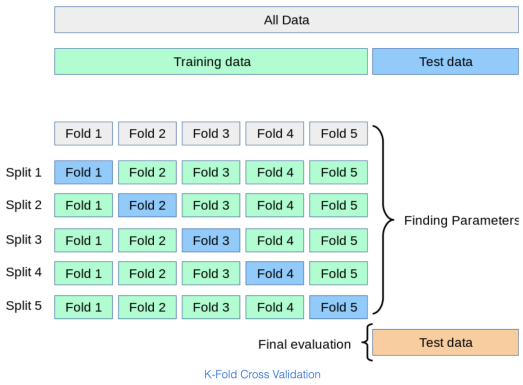

Figure 5. K-Fold Cross Validation

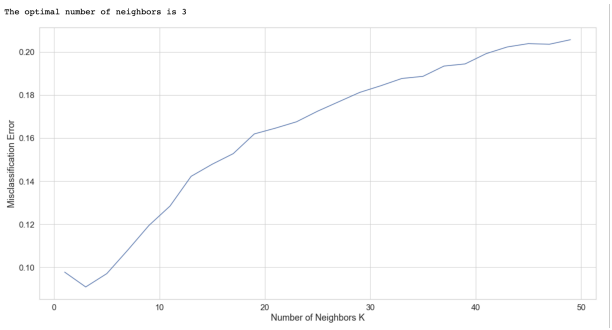

Figure 6. 10-fold cross validation tells us that k=7 results in the lowest validation error.

internal node has a decision rule that splits the data. You can understand the Gini index as a cost function used to evaluate splits in the data set. It is calculated by subtracting the sum of the squared probabilities of each class from one. It favors larger partitions and easy to implement whereas information gain favors smaller partitions with distinct values. The 'value' row in each node tells us how many of the observations that were sorted into the node fall into each of 2 categories. The biggest drawback to decision trees is that the split it makes at each node will be optimized for the data set it is fit to. This splitting process will rarely generalize well to other data. However, we can generate huge number of these decision trees, turned in slightly different ways, and combine their predictions to create some of our best models today. For our data set the decision tree classifier gave an accuracy of 95.32%.

2.2.2 k-nearest neighbour

The second classification algorithm we used for this project is K nearest neighbor classifier. It can be used both for classification and regression predictive problems[6] . It is one of the simplest algorithm. Even with such simplicity it gives competitive results. It is based on finding Euclidean distance between the new point and the existing points and then choosing k nearest neighbors. K in KNN is a hyperparameter that we should choose to get the best possible fit for the data set[4] . Intuitively we can think of K as controlling the shape of the decision boundary.

If we choose a small value of K, we are restraining the region of a given prediction and forcing our classifier to be more blind to the overall distribution. A small value of k provides the most flexible fit, which will have low bias but high variance. On the other hand, a larger value of K will have smoother decision boundaries which mean lower variance but increased bias.

An alternative and smarter approach involves estimating the test error rate by holding out a subset of the training set from the fitting process. This subset, called the validation set can be used to select the appropriate level of flexibility of our algorithm. For this data set, we have used k-fold cross-validation. As shown in figure 5, we can see that K-fold cross-validation involves randomly dividing the training set into k groups or folds of approximately equal size. The first fold is treated as a validation set and the method is fit on the remaining k-1 folds. This classification rate is then computed on the observations is treated as a validation set [4]. This process results in k estimates of the test error which are then averaged out. For our data set, we got k = 3 from 10-fold cross-validation as shown in figure 6. After using the value of k = 3, we got 90.53% accuracy.

2.2.3 Gaussian Naive Bayes Model

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of the feature given the value of the class variable [7]. Naive Bayes learners and classifier can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions means that each distribution can be independently estimated as one dimensional distribution. This in turn helps to alleviate problems stemming form the curse of dimensional.Gaussian Naive Bayes implements the Gaussian Naive Bayes algorithm for classification. The Gaussian distribution depends on 2 parameters of a series- the mean and the Standard Deviation.

Using the Gaussian Distribution Function, shown in Figure 7, if we input the values from series instead of x, using the mean value of the series and its standard deviation we can find the probability that the value x will occur. Using the Gaussian Naive Bayes classifier, we got 88.457% of accuracy. This is because it uses the probability function to predict the results.

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Figure 7. The Gaussian Distribution Function [source-https://i.stack.imgur.com/bBIbn.png.

2.2.4 Logistic Regression Model

Logistic Regression is one of the most simple and commonly used Machine Learning algorithms for two-class classification. In our data set, we are doing a binary classification. Logistic regression is mainly used for such binary classification. The outcome or target variable is dichotomous in nature [8]. It is a special case of linear regression where the target variable is categorical in nature. For this reason, logistic regression uses a Sigmoid function for binary classification. The Sigmoid function can take any real-valued number and map it into a value between 0 and 1. If the curve goes to positive infinity,y predicted will become 1 and if the curve goes to negative infinity, y predicted will become 0. If the output of the Sigmoid function is more than 0,5, we can classify the outcome as 1 or yes, and if it is less than 0,5, we can classify it as 0 or N0. The sigmoid function is given as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

For our data set using logistic regression from sci-kit learn library, we got 87.9% accuracy. It has shown higher accuracy than Gaussian Naive Bayes classifier.

3. Experimental Results

From this analysis, we found that there was no correlation between app features like size, rating, number of installs, and price and even between price and ratings. There was a strong negative correlation between the number of installs and the number of reviews. Most numbers of apps belonged under genres of tools, Entertainment, Education, Business, and Medical. On the base of the number of installs, we divided the apps into two categories: successful and unsuccessful. Decision Tree gave the highest accuracy percentage of 95.32% and the Gaussian Naive Bayes model gave the lowest accuracy of 88.45%.

The decision tree worked well because we had a simple model and we had a really important feature to take the decision which was the number of installs. On the other hand, we got the lowest accuracy using the Gaussian Naive Bayes model because it has strong feature independence assumptions. In Naive Bayes, the assumption that all features are independent is not usually the case in real life so it makes it less accurate [11]. There were total 33 different categories in our data set We also found that the average rating for all the apps was 4.0 which means that most people rated if they like the apps. The apps with most importance belonged to categories of "Entertainment" and "AUTO_AND_VEHICLE".

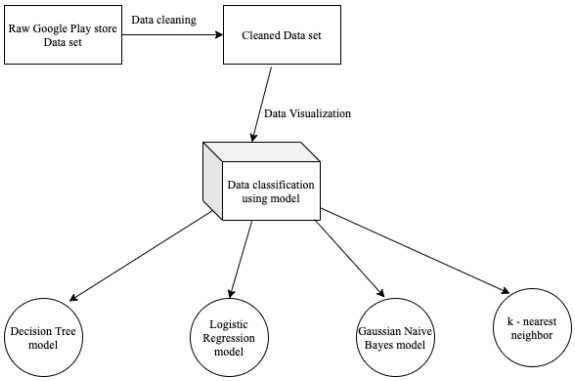

Figure 8. The overall architecture for App analysis

4. Conclusion and future work

This data set contains a large amount of data that can be used for various purposes. Currently, the decision tree model made using this data set can be used for future developers and Google plays store team to glance at the google play store market and what categories of the apps should be made to keep google play store popular in the future. It can be used to improve business values and google play store in general. It is not just limited to the problem we solved.

Using this data set, we applied various classification algorithms and found that Decision tress fits best for our problem statement. We also discovered how different algorithms work in different cases. We found that the Decision tree is easy to visualize and explain the model implementation and it also saves computational power. Using this data set the future work includes the prediction of other parameters such as the number of reviews and installs based on the regression model, identifying the categories and statistics of the most installed apps, exploring the correlation between the size of the app and its version of Android, etc on the number of installs.

5. Task Assignment and Acknowledgement

This project was completed by Rimsha Maredia under the supervision and instruction from Dr. Zhangyang (Atlas) Wang and Zhenyu Wu. The dataset was used from the Kaggle data store.

References

[1] Kaggle.com.(2018). Google Play Store Apps.[online] https://www.kaggle.com/lava18/google-play-store-apps [Accessed 3 Mar. 2020].

[2] "Mining and Analysis of Apps in Google Play," Proceedings of the 9th International Conference on Web Information Systems and Technologies, 2013.

[3] Google play store: number of apps 2018(2018). [online] https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/ [Accessed 3 Mar. 2020].

[4] Amit Chile, Dr. P. R. Gundalwar.(2019). Analysis of Google Play Store Application.[online] http://ijraset.com/fileserve.php?FID=24134 [Accessed 3 Mar. 2020]

[5] T. Denoeux, M. Skarstein-Bjanger, Induction of decision trees from partially classified data, in: Proceedings of the 2000 IEEE International Conference on Systems, Man and Cybernetics (SMC'00), IEEE, Nashville, TN, 2000, pp. 2923–2928.

[6] Harman, M., Jia, Y., and Zhang, Y. (2012). App store mining and analysis: Msr for app stores. In 2012 9th IEEE Working Conference on Mining Software Repositories (MSR),pages 108–111.

[7] R. P. Rajeswari, K. Juliet, and Aradhana, "Text Classification for Student Data Set using Naive Bayes Classifier and KNN Classifier," Int. J. Comput. Trends Technol., vol. 43, no. 1, pp. 8–12, 2017. https://doi.org/10.14445/22312803/ijctt-v43p103

[8] Jong, J. (2011). Predicting rating with sentiment analysis. [online] http://cs229.stanford. edu/proj2011/Jong-PredictingRatingwithSentimentAnalysis.pdf.

[9] [2015].Grover, S. 3 apps that failed (and what they teach us about app marketing). [online] https://blog.placeit.net/apps-fail-teach-us-app-marketing/.

[10] H. G. Schnack, M. Nieuwenhuis, N. E. van Haren, L. Abramovic, T. W. Scheewe, R. M. Brouwer, H. E. Hulshoff Pol, and R. S. Kahn, "Can structural MRI aid in clinical classification? A machine learning study in two independent samples of patients with schizophrenia, bipolar disorder and healthy subjects," NeuroImage, vol. 84, pp. 299–306, jan 2014.