Google Play store apps Analysis

Neha Supe
MS in Computer Science at Indiana University Bloomington, IN, US
nehasupe@iu.edu

Abstract

Classification and Prediction are two common tasks in Machine Learning. The Google Playstore dataset has various interesting features which can be explored and used for these tasks. The two Machine learning tasks performed are- Predicting the rating of an app, sentiment analysis for app reviews.

1 Introduction

Google has millions of Android apps on its playstore. Some apps become very popular. The users can let the developers and other users know how they feel about the app easily through ratings and reviews. For apps to remain popular, it is essential for app developers to look out for these feedbacks and not solely rely on the app installs number as the app store environment is highly competitive and anytime an app could be replaced by another if the developers do not bring out upgrades in their app to satisfy the goals of the user.

In this paper, we will look at the google play store dataset. This dataset was downloaded from Kaggle. It consists of two files and has two datasets. Both files have different features. I decided to use both of them as I thought both could provide interesting visualization and inferences. The two tasks that I decided to perform on these datasets were-

Task 1: The first task is to predict the ratings for an application given its features. We have performed regression to train the model and used the trained model coefficients to predict the rating of a app that was not seen in the training set.

Task 2: The second task is to do sentiment analysis on reviews. The reviews for the app are classified as positive, negative or neutral.

The entire project was written in python. The libraries used for querying and visualization are- Pandas, matplotlib, WordCloud, and Seaborn. The NLTK and Sklearn library was used for preprocessing the text data for the second task. All the models and evaluations were done using Sklearn library.

Using the dataset, the following contributions have been made through this project:

- Explored the dataset. Performed various queries to understand the data in the dataset

- Performed the task of predicting the ratings for an app given some of its features.

- Performed the task identifying the sentiment from the reviews.

2 Data Description

The datset consists of 2 files- Googleplaystore.csv and Googleplaystore_user_reviews.csv. The first file contains 13 features and 10841 rows. The App feature uniquely identifies each row. The category feature has categorical values like 'ART_AND_DESIGN', 'FAMILY',etc. The 'FAMILY' category is seen to have most number of apps followed by 'GAME'. The Rating feature is the rating of the app. There 13% values are missing in this feature. The value for this feature range from 0-5.

The Reviews feature has the number of reviews for the app. The Size feature tells the size of the app. The size of the app is in MB or kB and for some apps the size varies with device so the have the value "Varies with device". Installs feature has number of installs of the apps. The numbers are not exact numbers but a range and have '+' at the end of the value.The Type feature tells whether the app is free or paid. The Price feature has the price of the app. Almost 80% apps in this dataset are free apps. The paid apps have Price value with prefix'$'.

The Content Rating tells who the audience can be or the required age limit to use the app. There

	No of missing values	%
Rating	1474	13.6
Current Ver	8	0.1
Android Ver	3	0.0
Type	1	0.0
Content Rating	1	0.0

Figure 1: Missing values for the Googleplaystore.csv

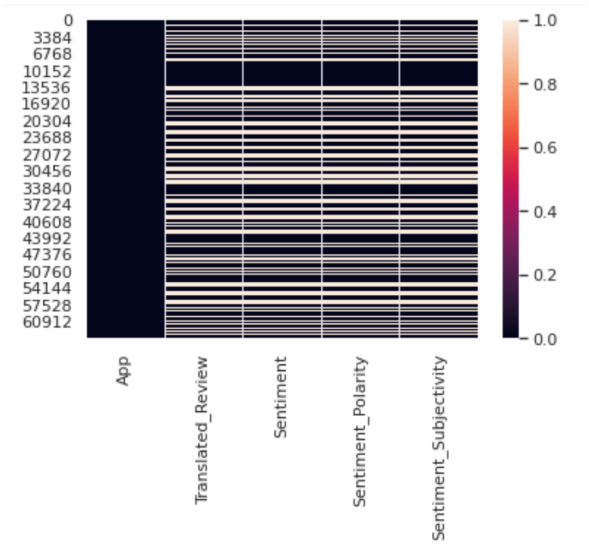

Figure 2: Missing values for the second csv

are 3 more features- Last Updates, Current ver, Andriod ver which were not explored or used for the Machine Learning task. It was observed from visualization that Reviews and Installs can be one of the most influencing features for the value of target feature. the seemed to have a clear correlating relationship.

The second file- googleplaystore_user_reviews.csv consists of features- App, Translated_Review, Sentiment, Sentiment_Polarity, Sentiment_Subjectivity. This dataset has 64295 rows. But 41.8% have null values. After dropping the missing values, only 37427 rows were left. The App feature uniquely identifies the row. Translated_review has the actual review provided by the user. Sentiment is categorical value- positive, negative, neutral and is the target feature for the task 2. Sentiment_polarity and Sentiment_Subjectivity feature gives an indication of how much a review is true to the sentiment. For the second task of classifying the sentiment of the review, the Sentiment feature was used as target feature and the Translated_Review feature was used for training the model. For this dataset, visualizations included for the vectorized reviews.

3 Data Pre-Processing and Feature Engineering

3.1 For Task 1

First, the data needs to be prepared. There are 1474 missing values for ratings feature. Since the task is to perform regression, we need all the features in a numerical format. From the 13 features, only rating, review are in actual correct numerical form, that is they are either in integer or float format. Other features like size have 'M ' and 'k' at the end indicating to be MB or kB size format, price feature has $ before the actual price number. So features like size, installs, price first were preprocessed to convert to only a numerical form. Anomalies such as a stray category in columns eg. In the Category feature had an anomaly value 1.9. The rating column had value above 5 which needed to be removed. In the size feature, some rows had value 'varies with device'. Such values were replaced by the mean of the column. The categories in feature Type and Content Rating were merged as they had same meaning. The other preprocessing step taken was converting the categorical features into numerical features using one-hot encoding. One-hot encoding maps all the categories in a feature into columns which represent binary values. One-hot encoding was performed for features like category, type, content rating. The feature genres also had categorical values but the values were a subset of the feature Category and had 119 values and hence were not considered for onehot encoding. The next decision was to drop certain features which were difficult to convert to numerical values like Current ver and Android ver.

3.2 For Task 2

For this task, our feature of concern is Translated_Review and Sentiment for training the model. The Translated_Review feature features reviews in English language with punctuations and other characters. To train the model, we need numerical values. So these strings of sentences need to be vectorized. The complete NLP pipeline is followed to convert the text review into numerical

Figure 3: Words in Reviews

vectors.

The following steps were taken to make the data ready for the model:

3.2.1 Remove Punctuation:

Characters like ’́, ’.’, ’!’, ’?’ etc were frequently observed in text.These need to be removed as the do not contribute any meaning to the target variable as the method that we are using for classifying relies on the count of the words. So ’Good!’ and ’Good’ need to be treated equally and are the same for our methodology. Hence, we strip of all characters that are not letters or numbers.

3.2.2 Tokenize:

The review string is then further tokenized. That is the string is broken down into list of words. By tokenizing, each word can be counted. Tokenizing makes the formation of the vector easier.

3.2.3 Remove stopwords:

Words like a, an, the, etc do not contribute much to the target variable. These words can be found in reviews of all three classes and hence do not provide any new information to differentiate between the classes. Also, it speeds up the training process as the model has to train on a smaller vector.

3.2.4 Lemmatization:

The tokenized words are converted to the root words. Again, reduces the size of the vector so the model can train faster.

3.2.5 Generating vectors:

Before performing this step, the dataset is split into train and test set. The words in the train set are considered to be the training vocabulary. The total training vocabulary is of 18824 words. The number of times a word occurs in a review is counted by CountVectorizer and a vector is generated for each review. So for training and testing the number of features is 18824 but the value of most of the is usually 0.

Algorithm	Root mean square error
Linear Regression	0.258
RandomForestRegressor	0.225
KNeighborsRegressor	0.302

Table 1: Results for prediction of ratings

Algorithm	Accuracy
MultinomialNB	0.772
Logistic Regression	0.925
MLPClassifier	0.90

Table 2: Results for classification of Sentiment

4 Modeling

4.1 Prediction task:

The dataset was split 70-30. 70% (6555 rows) data was used for training and 30% (2810 rows) were used for testing. Three models were used for training and tested, and their performance was recorded. The three models are- Linear Regression, KNeighborsRegressor, RandomForestRegressor. All the models were imported from sklearn library.

4.2 Classification task:

The dataset was split in the same way as task 1- 70-30 split. 26198 rows for training and 11229 for testing. The review feature was vectorized and used for training the models.

5 Results

The result for task 1 were observed as follows in Table 1. The predicted values were evaluated with root mean square error. It can be seen that the RandomForestRegressor is performing best.

For task 2, the results observed were as follows in Table 2

6 Evaluation/ Discussion of results

For the task of predicting the ratings, if the predicted values are observed and compared to the actual values it can be observed that the predicted values like around the median values. For the actual rating of 2 or 3, suppose the model is predicting a value of 4, the model could be overfitting

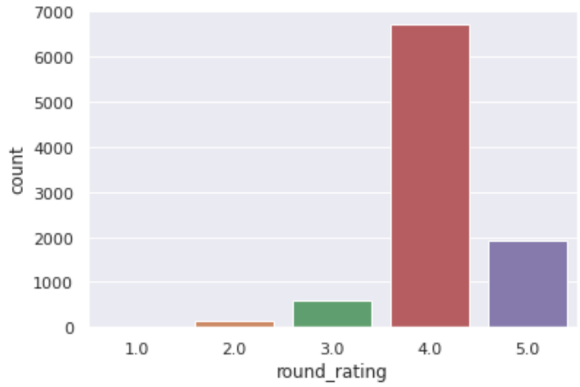

Figure 4: Target feature 'Rating'

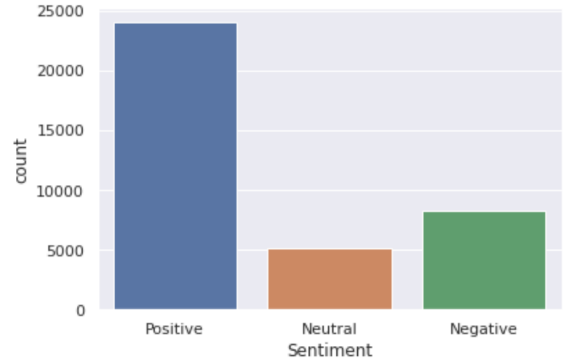

Figure 5: Target feature 'Sentiment'

as it is always predicting value close to a median value.

For the sentiment analysis task, the target feature is quiet unbalanced. The reviews with positive sentiment are 23998 in number and whereas Negative are 8271 and Neutral are 5158. This could also lead to model overfitting.

One other problem seen with this methodology is, since it uses tokenized words for training and not takes into account the sequence of words, it can lead to some wrong classifications. For eg. "This app should rank as the best app for slow apps". This review is a negative review but can be classified as positive as 'best' word related to positive class.

7 Conclusion

The Google play store dataset was explored in this paper. Various features in the dataset and how they correlate to each other was understood. Visualizations helped to understand the features in a better way. Inferences could be drawn based on visualizations. These inferences were tested using Machine Learning Algorithms. Two tasks- prediction of rating of an app and sentiment analysis of a review were performed. The results were evaluated and limitations of these approaches were discussed. The RandomForestRegressor model performed best for the prediction task and Logistic Regression performed best for the classification task.

8 References

[1]https://pandas.pydata.org/
[2]https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
[3]https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
[4]https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[5] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html
[6]https://scikit-learn.org/stable/auto_examples/neighbors/plot_regression.html#sphx-glr-auto-examples-neighbors-plot-regression-py
[7]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[8]https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html
[9]https://www.nltk.org/
[10]https://seaborn.pydata.org/
[11]https://pypi.org/project/wordcloud/
[12]https://matplotlib.org/