Predicting Google Play Store Apps' Ratings

Jihoon Kim

Abstract

In my DSE 200X Final Project, I will be using a Google PlayStore App dataset from kaggle.com, which features the apps and their ratings, installs, price, reviews, sentiment analysis, etc. With this dataset, I will figure out if I can use Regression Analysis using Machine Learning and sklearn to predict the rating of apps with their sentiment polarity and sentiment subjectivity. Before I made a machine learning model, I figured out the best features for the model by making scatter plots using matplotlib and finding correlation coefficients. By finding the mean and median of all of the sentiment polarities/subjectivities in an app's reviews, I was able to train and test a regression machine learning model to predict the rating of an app. I also performed the same test but with a Decision Tree Regressor instead. The most accurate model was a Decision Tree Regressor model, which had the closest min, max, and mean of the test values and had a reasonable RMSE value.

Motivation

With more smartphones, especially Andriod phones, being more affordable and advanced, people from all over the world now have access to smartphones. And with more Andriod users, there are more reviews in the Google Play Store. The overwhelming amount of reviews may startle app developers as they won't be able to get through all of them and cannot see how their product is in the eyes of consumers. So, I decided to explore the best ways to determine an app's rating by using Regression Analysis with the app's Sentiment Polarity and Subjectivity. With this model, the app developers will be able to quickly check if their app is gaining praise or criticism and will be able to better improve their app more efficiently.

Dataset(s)

- I used the Google Play Store Apps dataset found on kaggle.com. It is a 2MB dataset with 2 csv files.
  - The first csv file called googleplaystore.csv has columns for app name, category, rating, reviews, size, installs, type, price, content rating, genres, last updated, current ver, and andriod ver for around 10.8k apps.
  - The second csv file called googleplaystore_user_reviews.csv has columns for app name, its translated review, sentiment analysis, sentiment polarity, and sentiment subject with around 64.3k reviews, but only for 865 apps.

Data Preparation and Cleaning

To prepare my Data for analysis, a lot of cleaning and preparations were made.

- googleplaystore.csv turned into dataframe called P_Store
  - Dropped 1481 null rows of googleplaystore.csv
  - For the installs column, had to delete every '+' (ex. 100000+) as it wouldn't convert to a float without the '+' being dropped
  - Also had to drop " , " for every value in installs column
  - For the Price column, had to drop '$'
  - For Size column, had to drop 'M' meaning Millions and had to replace 'k' with '000' (k meaning thousands)
  - Deleted unnecessary columns: Current Ver, Last Updated, Andriod Ver, Category, Genre, Type
  - For Content Rating column, replaced Everyone to 0, Everyone 10+ to 1, Teen to 2, Mature 17+ to 3, Adults only 18+ to 4, and Unrated to 5 so it can be used as a potential feature in matplotlib and regression.

Data Preparation and Cleaning cont.

- Googleplaystore_user_reviews.csv
  - Dropped 26,868 null rows
  - Dropped unnecessary columns: Translated Review, Sentiment
- Combining both csv files into one dataframe
  - Used pd.merge() to inner merge both dataframes called merged_inner
  - With the merged dataframe, grouped it by App through sentiment polarities' mean/mode and sentiment subjectivities' mean/mode called grouped_apps… (created 4 more dataframes)
    - This made the app the index, which I didn't want, so I reset the index so there App would be a column and integers would be the index
  - Then merged grouped_apps and P_Store since there weren't reviews for all apps in P_Store. This would negate that and only shows apps with reviews.
  - For best predictions, deleted apps with < 100 reviews. (Only 1 was dropped)

Now, I was ready to experiment with my data.

Research Questions

- Which factors correlate positively with the rating of an app?
- Can Machine Learning be used to accurately predict the rating of an app using the correct features above?

Methods

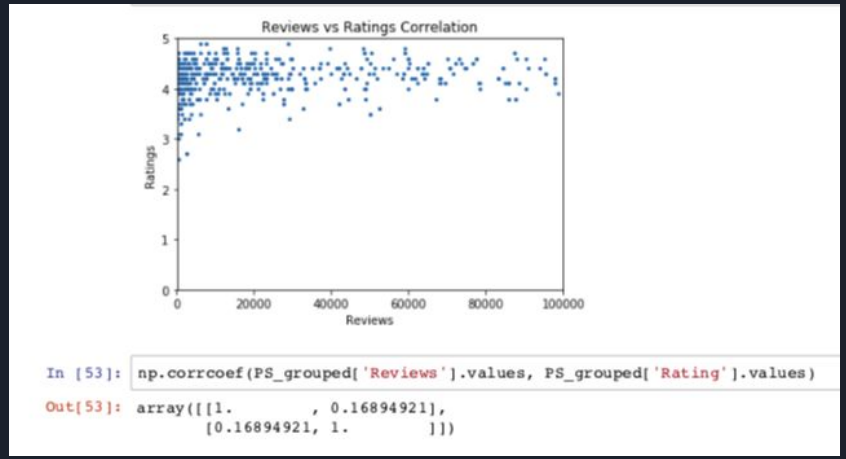
To find the features that correlate with the rating of an app, I made a scatter plot using matplotlib for each feature vs rating. The features with the best correlations would be added onto the features list for machine learning.

To find the best prediction model using regression, I used 2 different methods: Regression Analysis and Decision Tree Regressor. Then I found the RMSE, min, max, and mean for each to figure out which model was more accurate.

Findings (Question 1)

The factors that I considered to test for features were: Reviews, Size, Installs, Sentiment Polarity (Mean), and Sentiment Polarity (Median)

- Reviews

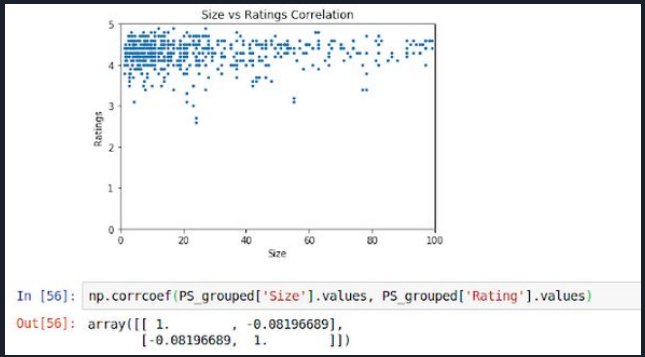


This graph and correlation shows that there is a positive correlation between Reviews and Ratings

- Even though there is a positive correlation, Reviews did not become a factor in the features list as the graph seemed too dispersed and the correlation coefficient was not strong.
- The axis were cut short as a Review value above 100000 were considered to be outliers.

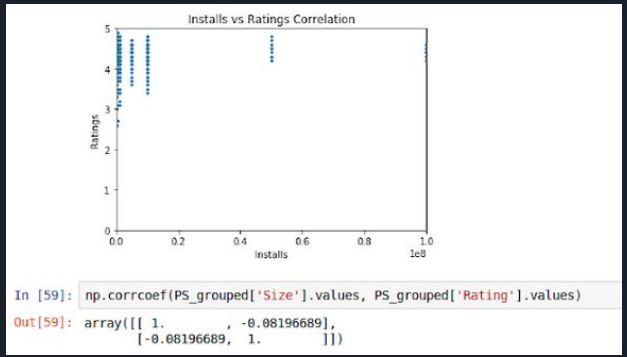
Findings (Question 1) cont.

- Size



The size graph and correlation coefficient proved to show that there is no correlation between Size and Ratings, thus Size is not a viable feature.

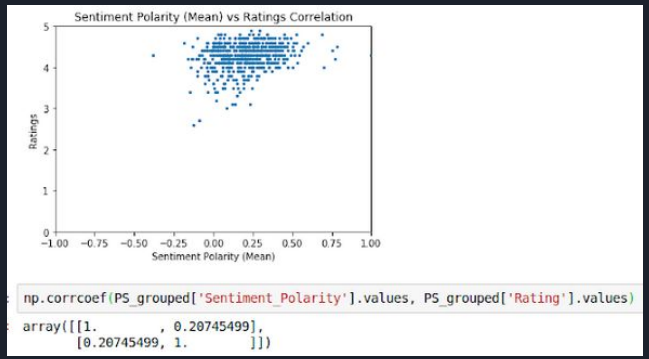
- Installs



The Installs graph and correlation coefficient proved to show that there is no correlation between Installs and Ratings, thus Installs is not a viable feature. Also, the data from Installs were too vague and was not a good source to be a viable feature.

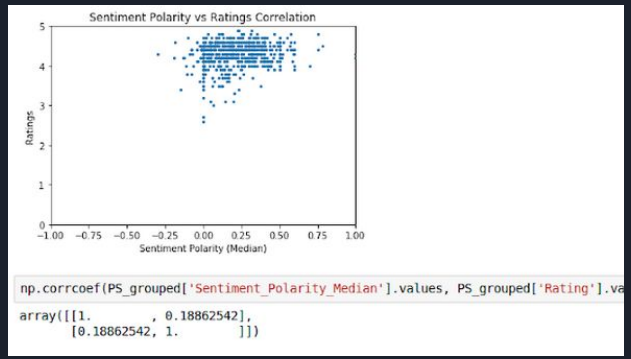
Findings (Question 1) cont.

- Sentiment Polarity (Mean)



The Sentiment Polarity (Mean) graph didn't really show any correlation, but the correlation coefficient states that there is a positive correlation. A little bit of positive correlation can be seen in the graph so it was put in the features for Regression.

- Sentiment Polarity (Median)



The same applies to this graph and coefficient. For the sake of uniformity, the median value was also put in the features list regardless.

Findings (Question 1) cont.

- Overall:
  - Even though correlation may be a good indicator to find the best features for a regression model, it did not prove to be true in this case. These findings veered me away from irrelevant data for the machine learning model, but they did not confirm me to use relevant data. However, Sentiment Polarity (both mean and mode) and Sentiment Subjectivity (both mean and mode) made it to the features list as I believed that those factors would definitely contribute to accurately predicting the rating of an app. And it did.

Findings (Question 2)

As stated earlier, I used 2 different regression models for this experiment

- 1st Model: Linear Regression Analysis using sklearn.linear_model

```
In [75]: print("Min: ", y_prediction.min())
         print("Max: ", y_prediction.max())
         print("Mean: ",y_prediction.mean())

         Min:  3.9936452760946657
         Max:  4.641480147458698
         Mean:  4.270467161750242

In [76]: y_test.describe()

Out[76]:
```

	Rating
count	268.000000
mean	4.300373
std	0.296104
min	2.700000
25%	4.100000
50%	4.300000
75%	4.500000
max	4.900000

```
In [77]: RMSE = sqrt(mean_squared_error(y_true = y_test, y_pred = y_prediction))

In [78]: print(RMSE)
         0.2927830970916976
```

By training and testing using the Linear Regression model, I found the min, max,mean, and RMSE of the predictions and actual results. It turns out that the mean and max were very close to the actual values, and the RMSE was at a reasonable point (lower is better). However, seeing that the min was more than 1 point off the actual value, this model proved to be inaccurate.

Findings (Question 2) cont.

2nd Model: Decision Tree Regressor using sklearn.tree

```
In [81]: print("Min: ", y_prediction.min())
         print("Max: ", y_prediction.max())
         print("Mean: ",y_prediction.mean())

         Min:  2.7
         Max:  4.8
         Mean:  4.276462519596849

In [82]: y_test.describe()

Out[82]:
```

	Rating
count	268.000000
mean	4.300373
std	0.296104
min	2.700000
25%	4.100000
50%	4.300000
75%	4.500000
max	4.900000

```
In [83]: RMSE = sqrt(mean_squared_error(y_true = y_test, y_pred = y_prediction))

In [84]: print(RMSE)

         0.30565474819566274
```

By training and testing using the Decision Tree Regressor model, I found the min, max,mean, and RMSE of the predictions and actual results. This model showed accurate values for min, max, and mean (actually very accurate);however, it had a higher RMSE value than the 1st model. But, this model still seemed to be the ore accurate since it was able to hit the min,max, and mean with great precision.

Limitations

- My experiment and results may not have been accurate as I may have not had the best data
  - I do not know how the reviews' sentiment analysis was performed
    - Did not have any power over these values and do not know for sure if they are correct although I tested some of the values by eye
  - More factors may influence the rating of an app that is not in my dataset and is not a factor that I am aware of
- The results for question 1 may not have been the best
  - I, currently, do not know of any other correlation models (exponential, log, etc) that may have sowed better correlation
  - I also do not know of any other values other than graphing and finding correlation coefficients to see if a factor should be in the features for a machine learning model predicting ratings
- A better/more sophisticated model might have been better for this experiment

Conclusions

- Which factors correlate positively with the ratings of apps?
  - According to my graphs and correlations, Reviews, Sentiment Polarity (Mean), and Sentiment Polarity (Median) correlate positively with the ratings of apps. However, these coefficients may not have been the best to go by.
- Can Machine Learning be used to accurately predict the rating of an app using the correct features above?
  - Yes, by using the Decision Tree Regressor and using the following features: Sentiment Polarity (Mean), Sentiment Polarity (Median), Sentiment Subjectivity (Mean), and Sentiment Subjectivity (Median), I was able to accurately predict the rating of an app with little errors.

Acknowledgements

- My dataset was from kaggle.com
  - Link: https://www.kaggle.com/lava18/google-play-store-apps
- Informal Analysis on Sentiment Polarity/Sentiment Subjectivity
  - Link: https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/
- I revised my work with my High School colleagues
  - Got feedback to not put Reviews in the features list even though it gave a positive correlation because of the graph.

References

Works Cited:

- https://www.kaggle.com/lava18/google-play-store-apps
- https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/