Netflix Reviews on Google Play Store

Narges Fardnia

July 2024

Contents

Introduction

Understanding customer opinions about a product or service is essential for growth and survival in today's competitive market. But can we predict customer preferences based on their past behavior? Can we recommend products they might like based on their history? How can we retain loyal customers by understanding their interests? How can we identify which types of products are more popular among consumers?

We can address all of these questions through data analysis. In this project, we will examine a dataset of Google Play Store user reviews on Netflix. Our primary objective is to develop a model that predicts user ratings using recommendation systems based on the available features.

The dataset is updated daily. For this task, I utilized version 52, collected on July 1, 2024. The dataset can be accessed through Kaggle, and the code for this project is available here.

Exploratory Data Analysis

This dataset contains 113609 records and 8 columns. The columns are:

- reviewId: Contains the ID of the review, including numbers, characters, and punctuation (type: object).
- userName: The chosen name of reviewer by themselves (type: object).
- content: The comments left by the reviewer (type: object).
- score: An integer from 1 to 5 indicating the user's satisfaction with the application (type: int64).
- thumbsUpCount: The number of positive reactions from other users to the given review (type: int64).
- reviewCreatedVersion: The version of review (type: object).
- at: The date and time of the review submission (type: object).
- appVersion: The version of the Netflix app on which the review was posted (type: object).

Missing Values

In the first step, we should address the missing values. To do so, we need to identify the missing values and then either fill them with appropriate values or drop them.

- We have 2 missing values in the userName column, which can be substituted with "user n" (where n is the index).
- We have 2 missing values in the Content column, which can be replaced with "no comments" to preserve other features related to these records.
- The columns reviewCreatedVersion and appVersion contain 16,638 missing values, which is about 15% of the total data in these columns. We can simply drop reviewCreatedVersion since it does not provide meaningful information. For appVersion, we can fill the missing values with an empty character to reduce data loss.

Visualization

In the next step, we aim to extract insights from the data through visualization. Figure 1 shows the distribution of scores. Significant portion of the scores are at both ends of the range: approximately 49% of

reviewers are dissatisfied with the Netflix app, giving it a score of 1 or 2. Conversely, around 41% of reviewers are highly satisfied, giving it a score of 5 or 4.

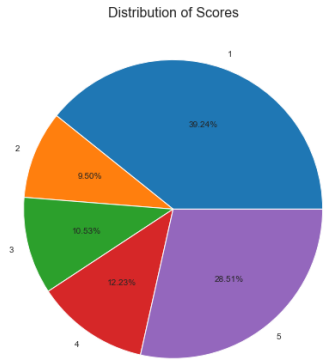

Figure 1. distribution of scores.

By analyzing the review dates, we observe that the number of dissatisfied reviews increased, with some fluctuations, from 2018 to 2024, peaking at approximately 9,500 reviews in 2024. On the other hand, although the number of satisfied reviews increased from 2018 to 2020, reaching a maximum of around 9,600 in 2020, it declined in the subsequent years, dropping to around 4,500 by 2024 (Figure 2).

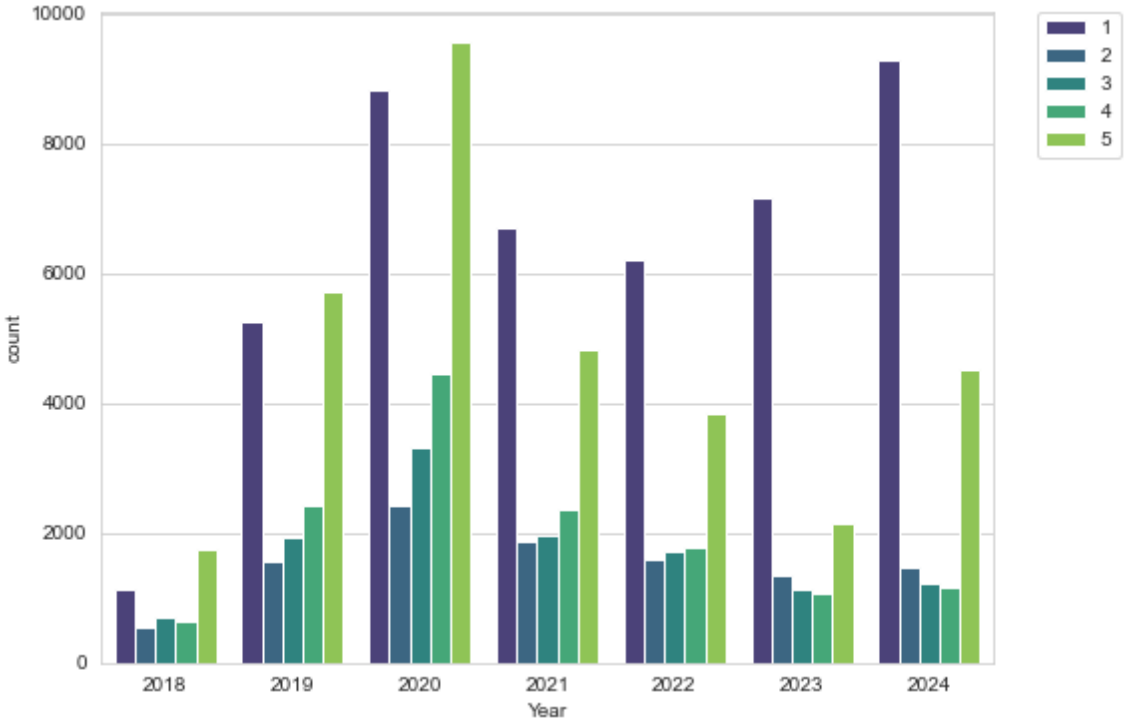

Figure 2. distribution of scores in different years.

The trend in the number of reviews containing content (not just scores) is also significant. Figure 3 illustrates this trend. The earliest available data is from September 12, 2018, resulting in a relatively low number of reviews, approximately 500, for that year. The number of reviews grew steadily, with slight fluctuations, until mid-2020, reaching around 3,000 reviews. After this peak, the number of reviews began

to decline, hitting a minimum of about 500 reviews by the end of 2023. Following this low point, the trend reversed, and by mid-2024, the number of reviews surged to a maximum of 5,000.

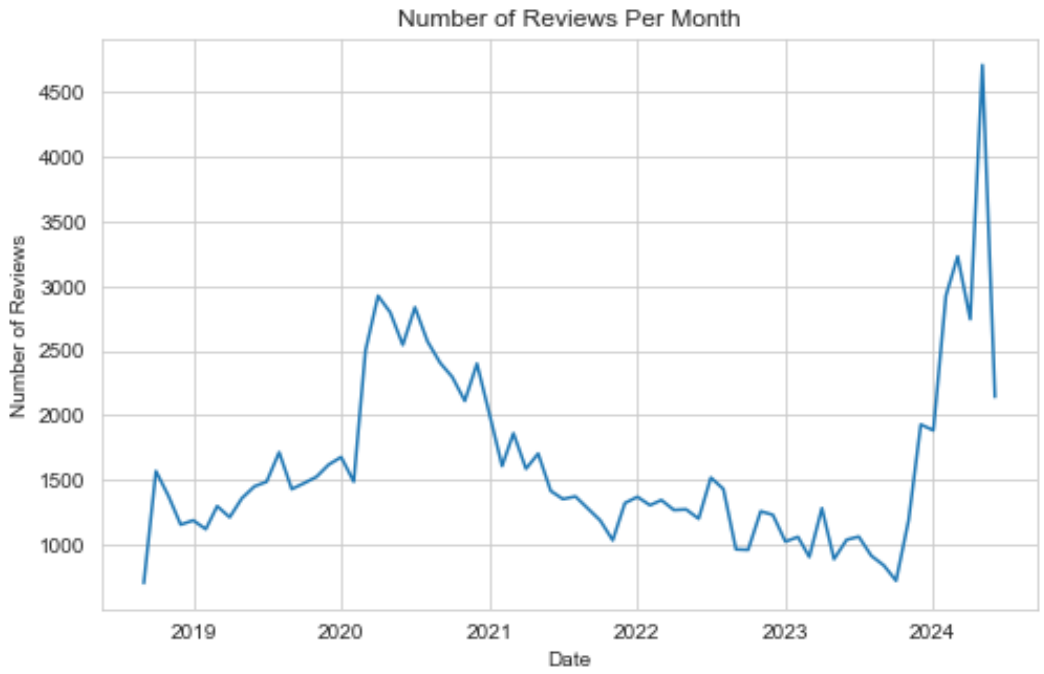

Figure 3. number of reviews per month.

Finally, analyzing the likes (thumbsUpCount) given to reviews reveals that the highest number of likes, with 8,032 thumbs up, was received by a comment posted on August 6, 2019: "'Wow! To be able to have Netflix on my phone is an absolute life-saver! Not only can I download my favourite TV shows and movies and be able to watch them anywhere at anytime (with or without Wi-Fi), there's also so much content to choose from (some a little too much content to choose from 😂). Really happy with it and looking forward to Stranger Things S4?" This user gave a score of 5 to the Netflix application. However, it's noteworthy that 111,561 reviews received fewer than 100 thumbs up. Interestingly, reviews with a score of 2, averaging around 16 likes, garnered the most agreement (Figure 4).

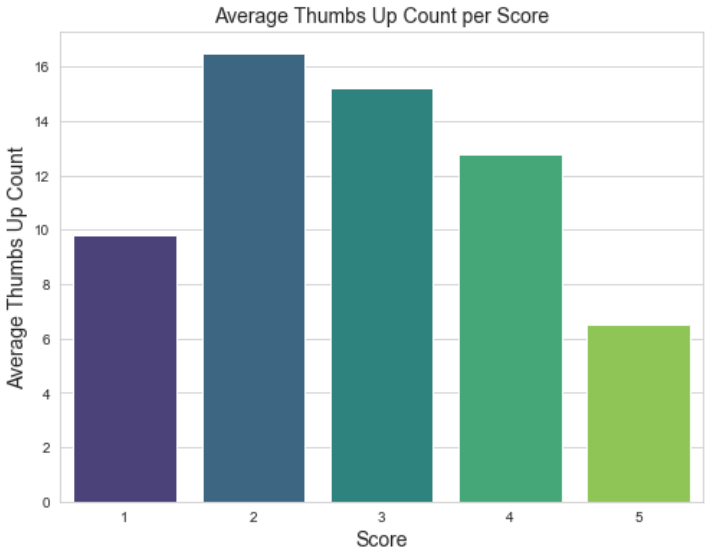

Figure 4. Average likes per score.

Text Analyzing

Analyzing text is often a challenging task due to its complexities. In this section, we focus on the "content" column, which contains user reviews, to extract meaningful insights. Firstly, by measuring the length of each review, we can introduce a new feature called "length" to our dataframe. The longest review contains 1,742 characters, while the shortest consists of a single character (an emoji). On average, reviews extend approximately 157 characters in length. Figure 5 demonstrates an almost consistent distribution of review lengths across different score categories.

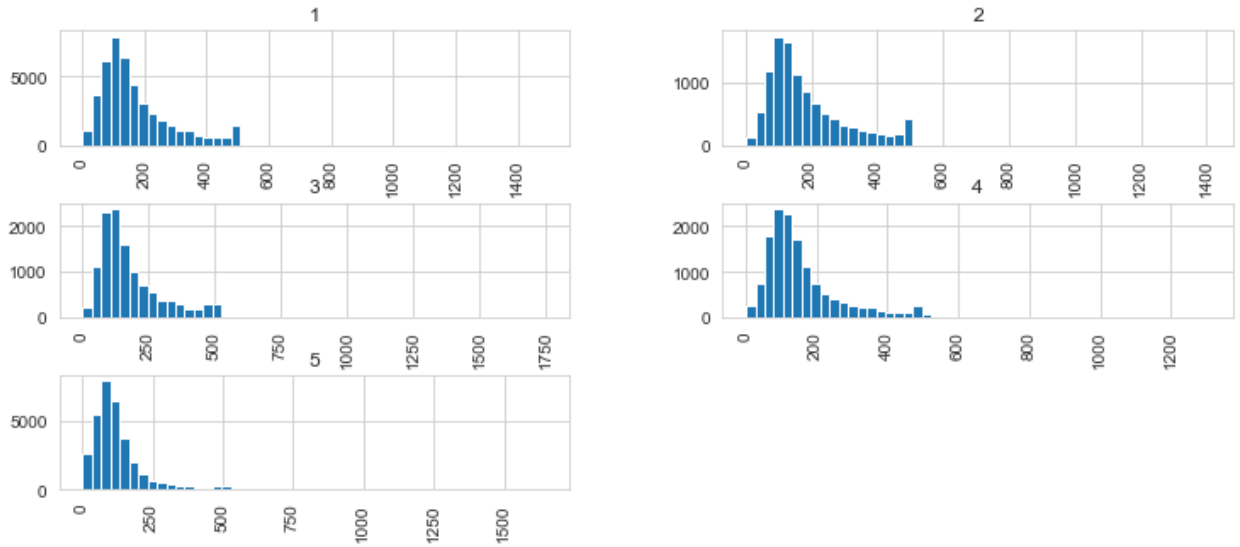

Figure 5. Distribution of review lengths across score classes.

After adding the new feature to the dataframe, the next step involves preprocessing the text. This includes the following steps:

1. Removing numbers

2. Converting all characters to lowercase
3. Removing mentions (using "@" symbol)
4. Removing links
5. Removing emojis
6. Removing punctuation
7. Lemmatizing and stemming words
8. Removing stopwords

Once the text is cleaned using these steps, we can utilize this feature to explore similarities within the data.

Figure 6. Word cloud for high and low scores.

Similarity

To identify similarities between features, various measures can be employed. In this section, we utilize correlations and cosine similarities. Figure 7 illustrates the cosine similarity among the numerical features. The most correlated features are thumbsUpCount and length, with a coefficient of 0.23, while score and length exhibit the highest negative correlation at -0.18.

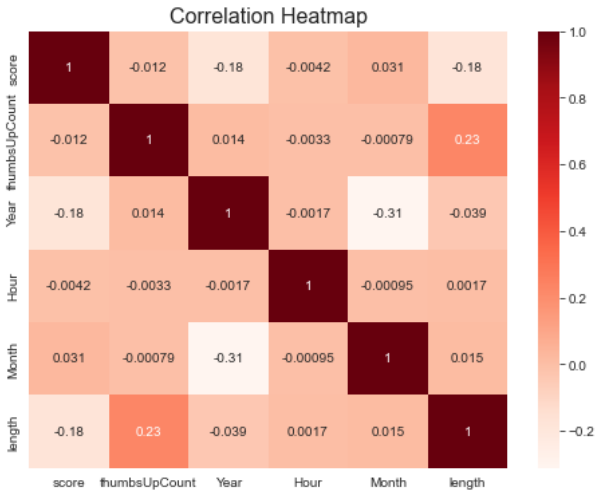

Figure 7. Correlation between numerical features.

We can also compute cosine similarities between categorical features by converting them into numerical vectors. Due to the large dataset size, we select a representative 10% sample and compute cosine similarity between "content" and "score" within this subset to avoid memory errors. This approach is similarly applied in the next section, where we aim to predict scores by leveraging similarities between features.

Recommendation System

A recommendation system (or recommender system) is a class of machine learning that uses data to help predict, narrow down, and find what people are looking for among an exponentially growing number of options (Anon., n.d.). Although it may seem unusual, in this section, we are not offering any options to the user. Instead, we aim to predict the score users will give the Netflix app if they see a new version of it on the Google Play Store, based on their previous reviews. Interestingly, we will not use any machine learning algorithms, as that is not the objective of this course. Instead, we will make our predictions using cosine similarity.

To achieve this goal, we first identify users who have posted more than one review. We then divide the features into two groups: categorical and numerical. We use one-hot encoding to convert categorical features into vectors. For numerical values, we scale them to ensure they are in the same range and to eliminate any significant gaps. After selecting the features, we calculate the cosine similarity between them. In the next step, we randomly select 20% of the data, remove their scores, and try to predict these values. To predict the scores, we define a function that identifies similar users and for each user with missing values, identify the top 10 similar users and averaging their given score. Finally, we can fill in the missing values with the weighted average.

Evaluation

For evaluating the model, we used Root Mean Square Deviation (RMSD) and obtained a value of 1.67 for the entire dataset.

Conclusion

In this project, we analyze a dataset containing reviews of Netflix on the Google Play Store, employing various techniques to extract information. Our approach involves establishing similarities between features and constructing a framework to predict future user scores for the app.

To enhance our model, we can integrate additional features such as TF-IDF or other similarity measures. Furthermore, implementing machine learning algorithms will enable us to develop a more robust recommendation system.

References

Anon., n.d. *NVIDIA.* [Online]
Available at: https://www.nvidia.com/en-us/glossary/recommendation-system/#:~:text=A%20recommendation%20system%20(or%20recommender,exponentially%20growing%20number%20of%20options.