Google Play Store: User Story 100 Report

User would like to see how many entertainment apps have a content rating of mature.

KenzieM

1. Introduction

For this report, I used data on apps from the Google Play Store to answer the user story: “User would like to see how many entertainment apps have a content rating of mature.” The data contains 10,841 rows and 13 columns. The columns of this data set include information such as name, category, genre, rating, size, number of installations, price, content rating and more. Every row of the set represents an individual application available on the Google Play Store. In the following report, I will discuss how I transformed the data in order to answer the question, and provide some visual aids to better understand that answer.

1. Data

As stated in the introduction, I worked with one CSV from the Google Play Store which contained 10,841 rows and 13 columns. I used Python’s Pandas library to parse the csv into a dataframe, from which I could inspect the rows and columns for irregularities. First, I checked the data for duplicate rows, of which there were about 1,500, and removed these and reset the indices. Next I inspected the columns for nulls and incorrect data types and found quite a few. Specifically four columns containing numerical information that were still listed as string types, one column with many null values, and several columns with only 1. After drilling down into the columns with only one null value, I found a row that had several entry errors and decided to delete this row outright since the dataset is relatively large. Of the columns with incorrect data types, the Reviews column was the most straightforward to fix. Two columns, however, contained symbols like commas, plus signs, and dollar signs, which make a direct type conversion a little trickier. I used the following code to remove the dollar signs from the price column:

|  |
| --- |
| df['Price'] = df['Price'].apply(lambda x: x.replace('$', '')) |

The column containing information about the size of the application was the most difficult to fix, as it had two different units, k for kilobytes and M for Megabytes. To do this, I wrote the following function, which removes the symbol and converts the data type to a numeric, before applying the correct multiplication:

|  |
| --- |
| def new\_size(x):  if 'k' in x:  return (float(x.replace('k', ''))) \* 1000  if 'M' in x:  return (float(x.replace('M', ''))) \* 1000000   df['Size'] = df['Size'].apply(new\_size) |

Once I completed the data type conversions, I saved the final cleaned data frame into a new CSV. I recognize that most of this cleaning was not necessary for this user story in particular, but I wanted to have a clean csv to work with on all future stories.

1. Method

With clean data in hand (on file?), I focused on the entertainment category as requested in the user story. I created a new data frame including just the rows which contain apps in the entertainment category. This new dataframe had 111 rows, meaning there are 111 total apps in the entertainment category. In order to determine how many apps were in each category, I used the following code to group the data frame by content rating and then count the entries in each group:

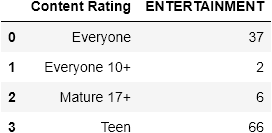
|  |
| --- |
| ent\_content = ent\_df.groupby('Content Rating').count() |

This gave me a data frame with an index of content rating and counts of the amount of rows for each one in all of the columns (each column now has identical values), so I just selected one of these columns for ease of use.

I also performed this aggregation using the crosstab() function, which allowed me to have a look at the frequencies of different content ratings in all categories. I used the following code to create the crosstab frequency table, and then isolate the Entertainment category:

|  |
| --- |
| content\_by\_category = pd.crosstab(df['Category'], df['Content Rating']) ent\_content = pd.DataFrame(content\_by\_category.loc['ENTERTAINMENT']) |

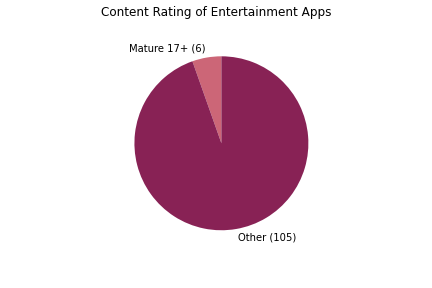
The benefit of using the crosstab function was that I was able to generate some other interesting questions about the data such as, why are there so many “Mature 17+” apps in the family category? The drawback of this method is that my resulting data frame contained rows with zero values, which I did not want included in my graphs. It is possible, however, for other purposes, that I may want those rows included, so I am glad to know this difference. Isolating the entertainment row first, on the other hand, removed the existence of content ratings not included, and thus they were not present in my groupby object. Here is the table I created and used for this story:



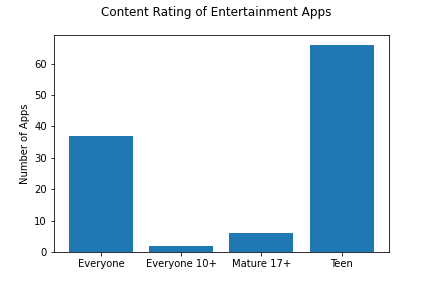
Finally, using the data contained in these tables, I used matplotlib.pyplot to plot some simple visuals representing the information, which will be shown in the results.

1. Results

After extracting just the data for content rating in the entertainment category, I was able to see that there are 6 apps in the entertainment category with content rating mature out of a total of 111 apps. I created the following pie chart comparing apps with rating mature vs all others (in the entertainment category):



The user story did not ask specifically for the breakdown of all content ratings in the entertainment category, but since I had the information readily available, I went ahead and made this bar graph:



1. Conclusion

In conclusion, there are 6 applications in the entertainment category that have a content rating of mature. These applications make up 5.41% of the entertainment category, which is a pretty small proportion. I hope you found this information useful. Feel free to ask additional questions about the data or provide any feedback on this report.