# Python_for_Data_Science_Project

May 8, 2019

## 1 Profitable App Profiles for the App Store and Google Play

The aim of this project is to find mobile app profiles that are profitable for the App Store and Google Play markets. Our job is to enable developers to make data-driven decisions with respect to the kind of apps they build.

These apps are free to download and install; therefore, the main source of revenue consists of in-app ads. This means that the revenue for any given app is mostly influenced by the number of users that use the apps. Our goal for this project is to analyze data to help developers understand what kinds of apps are likely to attract the most users.

## 2 Opening and Exploring the Data

As of September 2018, there were approximately 2 million iOS apps available on the App Store, and 2.1 million Android apps available on Google Play.

Collecting data for over four million apps requires a significant amount of time and money, so we'll try to analyze a sample of data instead. To avoid spending resources with collecting new data ourselves, we should first try to see whether we can find any relevant existing data at no cost. Luckily, these are two data sets that seem suitable for our purpose:

- App Store data set (contains approximately 7,000 iOS apps from the App Store)
- Google Play data set (contains approximately 10,000 Android apps from the Google Play Store)

Let's start by opening the two data sets

```
In [1]: from csv import reader
        ### App Store data set ###
        opened_file = open('AppleStore.csv')
        read_file = reader(opened_file)
        appstore = list(read_file)
        app_header = appstore[0]
        app_data = appstore[1:]

        ### Google Play data set ###
        opened_file = open('googleplaystore.csv')
        read_file = reader(opened_file)
```

```
googleplay = list(read_file)
google_header = googleplay[0]
google_data = googleplay[1:]
```

First we'll write a function named `explore_data()` that we can use repeatedly to explore rows in a more readable way. We'll also add an option for our function to show the number of rows and columns for any data set.

```
In [2]: def explore_data(dataset, start, end, rows_and_columns=False):
            dataset_slice = dataset[start:end]
            for row in dataset_slice:
                print(row)
                print('\n') # adds a new (empty) line after each row

            if rows_and_columns:
                print('Number of rows:', len(dataset))
                print('Number of columns:', len(dataset[0]))
```

Now let's use `explore_data()` on our two data sets.

```
In [3]: print(app_header)
        print('\n')
        explore_data(app_data, 0, 4, rows_and_columns = True)
```

```
['id', 'track_name', 'size_bytes', 'currency', 'price', 'rating_count_tot', 'rating_count_ver',


['284882215', 'Facebook', '389879808', 'USD', '0.0', '2974676', '212', '3.5', '3.5', '95.0', '4+


['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23',


['529479190', 'Clash of Clans', '116476928', 'USD', '0.0', '2130805', '579', '4.5', '4.5', '9.24


['420009108', 'Temple Run', '65921024', 'USD', '0.0', '1724546', '3842', '4.5', '4.0', '1.6.2',


Number of rows: 7197
Number of columns: 16
```

We have 7197 iOS apps in the App Store data set, and the columns that seem interesting are: 'track_name', 'currency', 'price', 'rating_count_tot', 'rating_count_ver', and 'prime_genre'. Not all column names are self-explanatory in this case, but details about each column can be found in the data set documentation.

```
In [4]: print(google_header)
        print('\n')
        explore_data(google_data, 0, 4, rows_and_columns = True)
```

```
['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type', 'Price', 'Content Rating',

['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,00

['Coloring book moana', 'ART_AND_DESIGN', '3.9', '967', '14M', '500,000+', 'Free', '0', 'Everyon

['U Launcher Lite - FREE Live Cool Themes, Hide Apps', 'ART_AND_DESIGN', '4.7', '87510', '8.7M',

['Sketch - Draw & Paint', 'ART_AND_DESIGN', '4.5', '215644', '25M', '50,000,000+', 'Free', '0',


Number of rows: 10841
Number of columns: 13
```

The Google Play data set has 10841 apps and 13 columns. At a quick glance, the columns that might be useful for the purpose of our analysis are 'App', 'Category', 'Reviews', 'Installs', 'Type', 'Price', and 'Genres'.

## 3 Deleting Wrong Data

The Google Play data set has a dedicated discussion section, and we can see that one of the discussions outlines an error in row 10472. Let's print this row and compare it against the header and another row that is correct.

```
In [5]: print(google_data[10472])  # row we are inspecting
        print('\n')
        print(google_header)  # header
        print('\n')
        print(google_data[0])      # correct row
```

```
['Life Made WI-Fi Touchscreen Photo Frame', '1.9', '19', '3.0M', '1,000+', 'Free', '0', 'Everyon

['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type', 'Price', 'Content Rating',

['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,00
```

Row 10472 corresponds to the app "Life Made WI-Fi Touchscreen Photo Frame" and we can see that the rating is 19. This is clearly off because the maximum rating for a Google Play app is 5. In reality, the rating is 1.9 and what is missing is the main category for this app. As a consequence, we'll delete this row.

```
In [6]: print(len(google_data))
        del google_data[10472]
        print(len(google_data))
```

```
10841
10840
```

# 4 Removing Duplicate Entries

## 4.1 Part One

If we explore the Google Play data set, we'll find that some apps have more than one entry. For example, the application Instagram has four entries:

```
In [7]: for app in google_data:
            name = app[0]
            if name == 'Instagram':
                print(app)
```

```
['Instagram', 'SOCIAL', '4.5', '66577313', 'Varies with device', '1,000,000,000+', 'Free', '0',
['Instagram', 'SOCIAL', '4.5', '66577446', 'Varies with device', '1,000,000,000+', 'Free', '0',
['Instagram', 'SOCIAL', '4.5', '66577313', 'Varies with device', '1,000,000,000+', 'Free', '0',
['Instagram', 'SOCIAL', '4.5', '66509917', 'Varies with device', '1,000,000,000+', 'Free', '0',
```

However, there are many more cases where an app appears more than once. This also occurs in the App Store data set.

```
In [8]: duplicate_apps = []
        unique_apps = []

        for app in google_data:
            name = app[0]
            if name in unique_apps:
                duplicate_apps.append(name)
            else:
                unique_apps.append(name)

        print('Number of duplicate apps: ', len(duplicate_apps))
        print('\n')
        print('Examples of duplicate apps: ', duplicate_apps[:10])
```

```
Number of duplicate apps:  1181


Examples of duplicate apps:  ['Quick PDF Scanner + OCR FREE', 'Box', 'Google My Business', 'ZOOM
```

Some apps appear multiple times in the data set. We don't want to count the apps more than once when we anaylze the data, so we need to remove the duplicate entries.

If we look at the multiple rows for the example above with the Instagram app, the reviews information shows different numbers indicating that the data was collected at different times. Therefore we'll keep the rows that have the highees number of reviews because the higher number of reviews, the more reliable the ratings.

To do that, we will: - Create a dictionary where each key is a unique app name, and the value is the highest number of reviews of that app - Use the dictionary to create a new data set, which will have only one entry per app (and we only select the apps with the highest number of reviews)

## 5 Part Two

Let's start by building that dictionary

```
In [9]: reviews_max = {}

        for app in google_data:
            name = app[0]
            n_reviews = float(app[3])

            if (name in reviews_max) and (reviews_max[name] < n_reviews):
                reviews_max[name] = n_reviews

            elif name not in reviews_max:
                reviews_max[name] = n_reviews
```

In a previous code cell, we found that there are 1,181 cases where an app occurs more than once, so the length of our dictionary (of unique apps) should be equal to the difference between the length of our data set and 1,181.

```
In [10]: print('Expected Length: ', len(google_data)-1181)
         print('Actual Length: ', len(reviews_max))
```

```
Expected Length:  9659
Actual Length:  9659
```

Now, let's use the reviews_max dictionary to remove the duplicates. For the duplicate cases, we'll only keep the entries with the highest number of reviews. In the code cell below: - We start by initializing two empty lists, `google_clean` and `already_added`. - We loop through the Google Play data set, and for every iteration: - We isolate the name of the app and the number of reviews. - We add the current row (app) to the `google_clean` list, and the app name (name) to the `already_cleaned` list if: - The number of reviews of the current app matches the number of reviews of that app as described in the reviews_max dictionary; and - The name of the app is not already in the already_added list. We need to add this supplementary condition to account for those cases where the highest number of reviews of a duplicate app is the same for more than one entry (for example, the Box app has three entries, and the number of reviews is the same). If we just check for reviews_max[name] == n_reviews, we'll still end up with duplicate entries for some apps.

```
In [11]: google_clean = []
         already_added = []

         for app in google_data:
             name = app[0]
             n_reviews = float(app[3])

             if (reviews_max[name] == n_reviews) and (name not in already_added):
                 google_clean.append(app)
                 already_added.append(name)
```

Now let's quickly explore the new data set, and confirm that the number of rows is 9,659.

```
In [12]: explore_data(google_clean, 0, 3, True)
```

```
['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,00


['U Launcher Lite - FREE Live Cool Themes, Hide Apps', 'ART_AND_DESIGN', '4.7', '87510', '8.7M',


['Sketch - Draw & Paint', 'ART_AND_DESIGN', '4.5', '215644', '25M', '50,000,000+', 'Free', '0',


Number of rows: 9659
Number of columns: 13
```

We have 9659 rows as expected.

# 6 Removing Non-English Apps

## 6.1 Part One

If you explore the data sets enough, you'll notice the names of some of the apps suggest they are not directed toward an English-speaking audience. Below, we see a couple of examples from both data sets:

```
In [13]: print(app_data[813][1])
         print(app_data[6731][1])

         print(google_clean[4412][0])
         print(google_clean[7940][0])
```

```
[U+7231][U+5947][U+827A]PPS -[U+300A][U+6B22][U+4E50][U+9882]2[U+300B][U+7535][U+89C6][U+5267]
[U+3010][U+8131][U+51FA][U+30B2][U+30FC][U+30E0][U+3011][U+7D76][U+5BFE][U+306B][U+6700][U+5F8
[U+4E2D][U+56FD][U+8A9E] AQ[U+30EA][U+30B9][U+30CB][U+30F3][U+30B0]
[U+0644][U+0639][U+0628][U+0629] [U+062A][U+0642][U+062F][U+0631] [U+062A][U+0631][U+0628][U+0
```

We're not interested in keeping these kind of apps, so we'll remove them. One way to go about doing this is by removing apps whose name contains a symbol that is not commonly used in English text.

English characters are encoded using the ASCII standard, with characters having a corresponding number between 0 and 127 associated with it. We can therefore take advantage of this to build a function that checks an app name and tells us whether it contains non-ASCII characters.

We built this function below, and we use the built-in ord() function to find out the corresponding encoding number of each character.

```
In [14]: def is_english(string):
             non_ascii = 0

             for character in string:
                 if ord(character) > 127:
                     non_ascii += 1

             if non_ascii > 3:
                 return False
             else:
                 return True

         print(is_english('Instagram'))
         print(is_english('[U+7231][U+5947][U+827A]PPS -[U+300A][U+6B22][U+4E50][U+9882]2[U+300
         print(is_english('Docs To Go™ Free Office Suite'))
         print(is_english('Instachat [U+1F61C]'))
```

```
True
False
True
True
```

## 6.2 Part Two

Below, we use the `is_english()` function to filter out the non-English apps for both data sets if the app name has more than three non-ASCII characters:

```
In [15]: apple_english = []
         google_english = []

         for app in app_data:
             name = app[1]
             if is_english(name):
                 apple_english.append(app)

         for app in google_clean:
             name = app[0]
             if is_english(name):
                 google_english.append(app)
```

```
explore_data(apple_english, 0, 3, True)
print('\n')
explore_data(google_english, 0, 3, True)
```

```
['284882215', 'Facebook', '389879808', 'USD', '0.0', '2974676', '212', '3.5', '3.5', '95.0', '4+
```

```
['389801252', 'Instagram', '113954816', 'USD', '0.0', '2161558', '1289', '4.5', '4.0', '10.23',
```

```
['529479190', 'Clash of Clans', '116476928', 'USD', '0.0', '2130805', '579', '4.5', '4.5', '9.24
```

```
Number of rows: 6183
Number of columns: 16
```

```
['Photo Editor & Candy Camera & Grid & ScrapBook', 'ART_AND_DESIGN', '4.1', '159', '19M', '10,00
```

```
['U Launcher Lite - FREE Live Cool Themes, Hide Apps', 'ART_AND_DESIGN', '4.7', '87510', '8.7M',
```

```
['Sketch - Draw & Paint', 'ART_AND_DESIGN', '4.5', '215644', '25M', '50,000,000+', 'Free', '0',
```

```
Number of rows: 9614
Number of columns: 13
```

We find that we're left with 9614 Google Play apps and 6183 Apple Store apps.

## 7 Isolating Free Apps

As mentioned in the introduction, we are only concerned with the free apps. Since our data sets contain both free and non-free apps, we'll need to isolate the free apps for our analysis.

```
In [16]: apple_final = []
         google_final = []

         for app in apple_english:
             cost = app[4]
             if cost == '0.0':
                 apple_final.append(app)

         for app in google_english:
             cost = app[7]
```

```
            if cost == '0':
                google_final.append(app)

        print('Final Data')
        print('Apple App Store: ', len(apple_final))
        print('Google Play Store: ', len(google_final))
```

```
Final Data
Apple App Store:  3222
Google Play Store:  8864
```

We find a total of 3222 apps in the Apple App Store and 8864 apps in the Google Play Store that meet our criteria.

# 8 Most Common Apps by Genre

## 8.1 Part One

As we mentioned in the introduction, our aim is to determine what kinds of apps are more likely to attract users as these will be the most profitable since the in app ad revenue is highly influenced by the number of people using the apps.

To minimize risk, our strategy for an app idea is comprised of three steps: - Build a minimal android version of the app and add it to Google Play. - If the app has a good response from users we then develop it further. - If the app is profitable after 6 months we also build an iOS version of the app and add it to the App Store.

Because our end goal is to add the app on both the App Store and Google Play, we need to find app profiles that are successful on both markets.

Let's begin the analysis by determining what the most common genres for each market are. For this, we'll build a frequency table for the prime_genre column of the App Store data set, and the Genres and Category columns of the Google Play data set.

## 8.2 Part Two

We'll build two functions to analyze the frequency tables: - One function to generate frequency tables that show percentages - Another function that we can use to display the percentages in a descending order

```
In [17]: def freq_table(dataset, index):
             table = {}
             total = 0

             for row in dataset:
                 total += 1
                 value = row[index]
                 if value in table:
                     table[value] +=1
                 else:
```

```
                table[value] = 1

        table_percentages = {}
        for key in table:
            percentage = (table[key] / total) * 100
            table_percentages[key] = percentage

        return table_percentages

    def display_table(dataset, index):
        table = freq_table(dataset, index)
        table_display = []
        for key in table:
            key_value_as_tuple = (table[key], key)
            table_display.append(key_value_as_tuple)

        table_sorted = sorted(table_display, reverse = True)
        for entry in table_sorted:
            print(entry[1], ':', entry[0])
```

### 8.3 Part Three

We start by examining the frequency table for the `prime_genre` column of the App Store data set.

```
In [18]: display_table(apple_final, -5)
```

```
Games : 58.16263190564867
Entertainment : 7.883302296710118
Photo & Video : 4.9658597144630665
Education : 3.662321539416512
Social Networking : 3.2898820608317814
Shopping : 2.60707635009311
Utilities : 2.5139664804469275
Sports : 2.1415270018621975
Music : 2.0484171322160147
Health & Fitness : 2.0173805090006205
Productivity : 1.7380509000620732
Lifestyle : 1.5828677839851024
News : 1.3345747982619491
Travel : 1.2414649286157666
Finance : 1.1173184357541899
Weather : 0.8690254500310366
Food & Drink : 0.8069522036002483
Reference : 0.5586592178770949
Business : 0.5276225946617008
Book : 0.4345127250155183
Navigation : 0.186219739292365
Medical : 0.186219739292365
```

```
Catalogs : 0.12414649286157665
```

We can see that among the free English apps, more than a half (58.16%) are Games. Entertainment apps are about 8%, followed by Photo & Video apps, which comprise about 5%. Only 3.66% of the apps are designed for Education, followed by Social Networking apps which account for 3.29% of the apps in our App Store data set.

The general impression is that App Store (at least the part containing free English apps) is dominated by apps that are designed for fun (games, entertainment, photo and video, social networking, sports, music, etc.), while apps with practical purposes (education, shopping, utilities, productivity, lifestyle, etc.) are more rare. However, the fact that fun apps are the most numerous doesn't also imply that they also have the greatest number of users — the demand might not be the same as the offer.

Let's continue by examining the Genres and Category columns of the Google Play data set (two columns which seem to be related).

```
In [19]: display_table(google_final, 1) # Category
```

```
FAMILY : 18.907942238267147
GAME : 9.724729241877256
TOOLS : 8.461191335740072
BUSINESS : 4.591606498194946
LIFESTYLE : 3.9034296028880866
PRODUCTIVITY : 3.892148014440433
FINANCE : 3.7003610108303246
MEDICAL : 3.531137184115524
SPORTS : 3.395758122743682
PERSONALIZATION : 3.3167870036101084
COMMUNICATION : 3.2378158844765346
HEALTH_AND_FITNESS : 3.0798736462093865
PHOTOGRAPHY : 2.944494584837545
NEWS_AND_MAGAZINES : 2.7978339350180503
SOCIAL : 2.6624548736462095
TRAVEL_AND_LOCAL : 2.33528880866426
SHOPPING : 2.2450361010830324
BOOKS_AND_REFERENCE : 2.1435018050541514
DATING : 1.861462093862816
VIDEO_PLAYERS : 1.7937725631768955
MAPS_AND_NAVIGATION : 1.3989169675090252
FOOD_AND_DRINK : 1.2409747292418771
EDUCATION : 1.1620036101083033
ENTERTAINMENT : 0.9589350180505415
LIBRARIES_AND_DEMO : 0.9363718411552346
AUTO_AND_VEHICLES : 0.9250902527075812
HOUSE_AND_HOME : 0.8235559566787004
WEATHER : 0.8009927797833934
EVENTS : 0.7107400722021661
PARENTING : 0.6543321299638989
```

```
ART_AND_DESIGN : 0.6430505415162455
COMICS : 0.6204873646209386
BEAUTY : 0.5979241877256317
```

This is significantly different on Google Play: there are not that many apps designed for fun, and it seems that a good number of apps are designed for practical purposes (family, tools, business, lifestyle, productivity, etc.). However, if we investigate this further, we can see that the family category (which accounts for almost 19% of the apps) means mostly games for kids.

Nonetheless, practical apps seem to be more popular on Google Play compared to the App Store. This is also confirmed by the frequency table we see for the Genres column:

```
In [20]: display_table(google_final, -4)
```

```
Tools : 8.449909747292418
Entertainment : 6.069494584837545
Education : 5.347472924187725
Business : 4.591606498194946
Productivity : 3.892148014440433
Lifestyle : 3.892148014440433
Finance : 3.7003610108303246
Medical : 3.531137184115524
Sports : 3.463447653429603
Personalization : 3.3167870036101084
Communication : 3.2378158844765346
Action : 3.1024368231046933
Health & Fitness : 3.0798736462093865
Photography : 2.944494584837545
News & Magazines : 2.7978339350180503
Social : 2.6624548736462095
Travel & Local : 2.3240072202166067
Shopping : 2.2450361010830324
Books & Reference : 2.1435018050541514
Simulation : 2.0419675090252705
Dating : 1.861462093862816
Arcade : 1.8501805054151623
Video Players & Editors : 1.7712093862815883
Casual : 1.7599277978339352
Maps & Navigation : 1.3989169675090252
Food & Drink : 1.2409747292418771
Puzzle : 1.128158844765343
Racing : 0.9927797833935018
Role Playing : 0.9363718411552346
Libraries & Demo : 0.9363718411552346
Auto & Vehicles : 0.9250902527075812
Strategy : 0.9138086642599278
House & Home : 0.8235559566787004
Weather : 0.8009927797833934
```

Events : 0.7107400722021661
Adventure : 0.6768953068592057
Comics : 0.6092057761732852
Beauty : 0.5979241877256317
Art & Design : 0.5979241877256317
Parenting : 0.4963898916967509
Card : 0.45126353790613716
Casino : 0.42870036101083037
Trivia : 0.41741877256317694
Educational;Education : 0.39485559566787
Board : 0.3835740072202166
Educational : 0.3722924187725632
Education;Education : 0.33844765342960287
Word : 0.2594765342960289
Casual;Pretend Play : 0.236913357400722
Music : 0.2030685920577617
Racing;Action & Adventure : 0.16922382671480143
Puzzle;Brain Games : 0.16922382671480143
Entertainment;Music & Video : 0.16922382671480143
Casual;Brain Games : 0.13537906137184114
Casual;Action & Adventure : 0.13537906137184114
Arcade;Action & Adventure : 0.12409747292418773
Action;Action & Adventure : 0.10153429602888085
Educational;Pretend Play : 0.09025270758122744
Simulation;Action & Adventure : 0.078971119133574
Parenting;Education : 0.078971119133574
Entertainment;Brain Games : 0.078971119133574
Board;Brain Games : 0.078971119133574
Parenting;Music & Video : 0.06768953068592057
Educational;Brain Games : 0.06768953068592057
Casual;Creativity : 0.06768953068592057
Art & Design;Creativity : 0.06768953068592057
Education;Pretend Play : 0.056407942238267145
Role Playing;Pretend Play : 0.04512635379061372
Education;Creativity : 0.04512635379061372
Role Playing;Action & Adventure : 0.033844765342960284
Puzzle;Action & Adventure : 0.033844765342960284
Entertainment;Creativity : 0.033844765342960284
Entertainment;Action & Adventure : 0.033844765342960284
Educational;Creativity : 0.033844765342960284
Educational;Action & Adventure : 0.033844765342960284
Education;Music & Video : 0.033844765342960284
Education;Brain Games : 0.033844765342960284
Education;Action & Adventure : 0.033844765342960284
Adventure;Action & Adventure : 0.033844765342960284
Video Players & Editors;Music & Video : 0.02256317689530686
Sports;Action & Adventure : 0.02256317689530686
Simulation;Pretend Play : 0.02256317689530686

```
Puzzle;Creativity : 0.02256317689530686
Music;Music & Video : 0.02256317689530686
Entertainment;Pretend Play : 0.02256317689530686
Casual;Education : 0.02256317689530686
Board;Action & Adventure : 0.02256317689530686
Video Players & Editors;Creativity : 0.01128158844765343
Trivia;Education : 0.01128158844765343
Travel & Local;Action & Adventure : 0.01128158844765343
Tools;Education : 0.01128158844765343
Strategy;Education : 0.01128158844765343
Strategy;Creativity : 0.01128158844765343
Strategy;Action & Adventure : 0.01128158844765343
Simulation;Education : 0.01128158844765343
Role Playing;Brain Games : 0.01128158844765343
Racing;Pretend Play : 0.01128158844765343
Puzzle;Education : 0.01128158844765343
Parenting;Brain Games : 0.01128158844765343
Music & Audio;Music & Video : 0.01128158844765343
Lifestyle;Pretend Play : 0.01128158844765343
Lifestyle;Education : 0.01128158844765343
Health & Fitness;Education : 0.01128158844765343
Health & Fitness;Action & Adventure : 0.01128158844765343
Entertainment;Education : 0.01128158844765343
Communication;Creativity : 0.01128158844765343
Comics;Creativity : 0.01128158844765343
Casual;Music & Video : 0.01128158844765343
Card;Action & Adventure : 0.01128158844765343
Books & Reference;Education : 0.01128158844765343
Art & Design;Pretend Play : 0.01128158844765343
Art & Design;Action & Adventure : 0.01128158844765343
Arcade;Pretend Play : 0.01128158844765343
Adventure;Education : 0.01128158844765343
```

The difference between the Genres and the Category columns is not clear cut, but we do notice that the Genres column is much more granular (it has more categories). We're only looking for the bigger picture at the moment, so we'll only work with the Category column moving forward.

Up to this point, we found that the App Store is dominated by apps designed for fun, while Google Play shows a more balanced distribution of both practical and for-fun apps. Now we'd like to get an idea about the kind of apps that have the most users.

## 9 Most Popular Apps by Genre in the App Store

One way to find out what genres are the most popular (have the most users) is to calculate the average number of installs for each app genre. For the Google Play data set, we can find this information in the `Installs` column, but for the App Store data set this information is missing. As a workaround, we'll take the total number of user ratings as a proxy, which we can find in the rating_count_tot app.

Below, we calculate the average number of user ratings per app genre on the App Store:

```
In [21]: apple_genres = freq_table(apple_final, -5)

         for genre in apple_genres:
             total = 0
             len_genre = 0
             for app in apple_final:
                 genre_app = app[-5]
                 if genre_app == genre:
                     n_ratings = float(app[5])
                     total += n_ratings
                     len_genre += 1
             avg_n_ratings = total / len_genre
             print(genre, ':', avg_n_ratings)
```

```
Medical : 612.0
Health & Fitness : 23298.015384615384
Utilities : 18684.456790123455
Finance : 31467.944444444445
Weather : 52279.892857142855
Entertainment : 14029.830708661417
Book : 39758.5
Education : 7003.983050847458
Navigation : 86090.33333333333
Sports : 23008.898550724636
Lifestyle : 16485.764705882353
News : 21248.023255813954
Reference : 74942.11111111111
Catalogs : 4004.0
Music : 57326.530303030304
Business : 7491.117647058823
Shopping : 26919.690476190477
Productivity : 21028.410714285714
Photo & Video : 28441.54375
Food & Drink : 33333.92307692308
Travel : 28243.8
Games : 22788.6696905016
Social Networking : 71548.34905660378
```

On average, Navigation apps have the highest number of user reviews, but this figure is heavily influenced by Waze and Google Maps, which have close to half a million user reviews together:

```
In [22]: for app in apple_final:
             if app[-5] == 'Navigation':
                 print(app[1], ':', app[5]) # print name and number of ratings
```

```
Waze - GPS Navigation, Maps & Real-time Traffic : 345046
Google Maps - Navigation & Transit : 154911
```

```
Geocaching® : 12811
CoPilot GPS - Car Navigation & Offline Maps : 3582
ImmobilienScout24: Real Estate Search in Germany : 187
Railway Route Search : 5
```

Reference apps have 74,942 user ratings on average, but it's actually the Bible and Dictionary.com which skew up the average rating:

```
In [23]: for app in apple_final:
             if app[-5] == 'Reference':
                 print(app[1], ':', app[5]) # print name and number of ratings
```

```
Bible : 985920
Dictionary.com Dictionary & Thesaurus : 200047
Dictionary.com Dictionary & Thesaurus for iPad : 54175
Google Translate : 26786
Muslim Pro: Ramadan 2017 Prayer Times, Azan, Quran : 18418
New Furniture Mods - Pocket Wiki & Game Tools for Minecraft PC Edition : 17588
Merriam-Webster Dictionary : 16849
Night Sky : 12122
City Maps for Minecraft PE - The Best Maps for Minecraft Pocket Edition (MCPE) : 8535
LUCKY BLOCK MOD ™ for Minecraft PC Edition - The Best Pocket Wiki & Mods Installer Tools : 4693
GUNS MODS for Minecraft PC Edition - Mods Tools : 1497
Guides for Pokémon GO - Pokemon GO News and Cheats : 826
WWDC : 762
Horror Maps for Minecraft PE - Download The Scariest Maps for Minecraft Pocket Edition (MCPE) Fr
VPN Express : 14
Real Bike Traffic Rider Virtual Reality Glasses : 8
[U+6559][U+3048][U+3066]!goo : 0
Jishokun-Japanese English Dictionary & Translator : 0
```

One thing we could do is take another popular book and turn it into an app where we could add different features besides the raw version of the book. This might include daily quotes from the book, an audio version of the book, quizzes about the book, etc. On top of that, we could also embed a dictionary within the app, so users don't need to exit our app to look up words in an external app. This idea seems to fit well with the fact that the App Store is dominated by for-fun apps. This suggests the market might be a bit saturated with for-fun apps, which means a practical app might have more of a chance to stand out among the huge number of apps on the App Store.

Other genres that seem popular include weather, book, food and drink, or finance. The book genre seems to overlap a bit with the app idea we described above, but the other genres don't seem too interesting to us: - Weather apps — people generally don't spend too much time in-app, and the chances of making profit from in-app adds are low. Also, getting reliable live weather data may require us to connect our apps to non-free APIs. - Food and drink — examples here include Starbucks, Dunkin' Donuts, McDonald's, etc. So making a popular food and drink app requires actual cooking and a delivery service. - Finance apps — these apps involve banking, paying bills, money transfer, etc. Building a finance app requires domain knowledge and we don't want to hire a finance expert just to build an app.

Now let's analyze the Google Play market a bit.

## 10 Most Popular Apps by Genre on Google Play

For the Google Play market, we actually have data about the number of installs, so we should be able to get a clearer picture about genre popularity. However, the install numbers don't seem precise enough — we can see that most values are open-ended (100+, 1,000+, 5,000+, etc.):

```
In [24]: display_table(google_final, 5) # the installs column
```

```
1,000,000+ : 15.726534296028879
100,000+ : 11.552346570397113
10,000,000+ : 10.548285198555957
10,000+ : 10.198555956678701
1,000+ : 8.393501805054152
100+ : 6.915613718411552
5,000,000+ : 6.825361010830325
500,000+ : 5.561823104693141
50,000+ : 4.7721119133574
5,000+ : 4.512635379061372
10+ : 3.5424187725631766
500+ : 3.2490974729241873
50,000,000+ : 2.3014440433213
100,000,000+ : 2.1322202166064983
50+ : 1.917870036101083
5+ : 0.78971119133574
1+ : 0.5076714801444043
500,000,000+ : 0.2707581227436823
1,000,000,000+ : 0.22563176895306858
0+ : 0.04512635379061372
0 : 0.01128158844765343
```

To perform computations we'll need to convert each install number to float — this means that we need to remove the commas and the plus characters, otherwise the conversion will fail and raise an error. We'll do this directly in the loop below, where we also compute the average number of installs for each genre (category).

```
In [26]: google_categories = freq_table(google_final, 1)

         for category in google_categories:
             total = 0
             len_category = 0
             for app in google_final:
                 category_app = app[1]
                 if category_app == category:
                     n_installs = app[5]
                     n_installs = n_installs.replace('+', '')
                     n_installs = n_installs.replace('-', '')
                     n_installs = n_installs.replace(',', '')
                     total += float(n_installs)
```

```
                    len_category += 1

            avg_n_installs = total / len_category
            print(category, ':', avg_n_installs)
```

```
BOOKS_AND_REFERENCE : 8767811.894736841
BEAUTY : 513151.88679245283
GAME : 15588015.603248259
SOCIAL : 23253652.127118643
COMMUNICATION : 38456119.167247385
ART_AND_DESIGN : 1986335.0877192982
COMICS : 817657.2727272727
EVENTS : 253542.22222222222
LIFESTYLE : 1437816.2687861272
EDUCATION : 1833495.145631068
PERSONALIZATION : 5201482.6122448975
NEWS_AND_MAGAZINES : 9549178.467741935
DATING : 854028.8303030303
WEATHER : 5074486.197183099
TOOLS : 10801391.298666667
HEALTH_AND_FITNESS : 4188821.9853479853
ENTERTAINMENT : 11640705.88235294
PHOTOGRAPHY : 17840110.40229885
PRODUCTIVITY : 16787331.344927534
SPORTS : 3638640.1428571427
HOUSE_AND_HOME : 1331540.5616438356
SHOPPING : 7036877.311557789
MAPS_AND_NAVIGATION : 4056941.7741935486
MEDICAL : 120550.61980830671
BUSINESS : 1712290.1474201474
VIDEO_PLAYERS : 24727872.452830188
FOOD_AND_DRINK : 1924897.7363636363
AUTO_AND_VEHICLES : 647317.8170731707
FAMILY : 3695641.8198090694
LIBRARIES_AND_DEMO : 638503.734939759
PARENTING : 542603.6206896552
TRAVEL_AND_LOCAL : 13984077.710144928
FINANCE : 1387692.475609756
```

On average, Communication apps have the most installs: 38,456,119. This number is heavily skewed up by a few apps that have over one billion installs (WhatsApp, Facebook Messenger, Skype, Google Chrome, Gmail, and Hangouts), and a few others with over 100 and 500 million installs:

```
In [28]: for app in google_final:
             if app[1] == 'COMMUNICATION' and (app[5] == '1,000,000,000+'
                                               or app[5] == '500,000,000+'
```

```
                                                or app[5] == '100,000,000+'):
                print(app[0], ':', app[5])
```

```
WhatsApp Messenger : 1,000,000,000+
imo beta free calls and text : 100,000,000+
Android Messages : 100,000,000+
Google Duo - High Quality Video Calls : 500,000,000+
Messenger - Text and Video Chat for Free : 1,000,000,000+
imo free video calls and chat : 500,000,000+
Skype - free IM & video calls : 1,000,000,000+
Who : 100,000,000+
GO SMS Pro - Messenger, Free Themes, Emoji : 100,000,000+
LINE: Free Calls & Messages : 500,000,000+
Google Chrome: Fast & Secure : 1,000,000,000+
Firefox Browser fast & private : 100,000,000+
UC Browser - Fast Download Private & Secure : 500,000,000+
Gmail : 1,000,000,000+
Hangouts : 1,000,000,000+
Messenger Lite: Free Calls & Messages : 100,000,000+
Kik : 100,000,000+
KakaoTalk: Free Calls & Text : 100,000,000+
Opera Mini - fast web browser : 100,000,000+
Opera Browser: Fast and Secure : 100,000,000+
Telegram : 100,000,000+
Truecaller: Caller ID, SMS spam blocking & Dialer : 100,000,000+
UC Browser Mini -Tiny Fast Private & Secure : 100,000,000+
Viber Messenger : 500,000,000+
WeChat : 100,000,000+
Yahoo Mail - Stay Organized : 100,000,000+
BBM - Free Calls & Messages : 100,000,000+
```

If we removed all the communication apps that have over 100 million installs, the average would be reduced roughly ten times:

```
In [29]: under_100_m = []

         for app in google_final:
             n_installs = app[5]
             n_installs = n_installs.replace(',', '')
             n_installs = n_installs.replace('+', '')
             n_installs = n_installs.replace('-', '')
             if (app[1] == 'COMMUNICATION') and (float(n_installs) < 100000000):
                 under_100_m.append(float(n_installs))

         sum(under_100_m) / len(under_100_m)

Out[29]: 3603485.3884615386
```

We see the same pattern for the video players category, which is dominated by apps like Youtube, Google Play Movies & TV, or MX Player. The pattern is repeated for social apps (where we have giants like Facebook, Instagram, Google+, etc.), photography apps (Google Photos and other popular photo editors), or productivity apps (Microsoft Word, Dropbox, Google Calendar, Evernote, etc.).

Again, the main concern is that these app genres might seem more popular than they really are.

The game genre seems pretty popular, but previously we found out this part of the market seems a bit saturated, so we'd like to come up with a different app recommendation if possible.

The books and reference genre looks fairly popular as well, with an average number of installs of 8,767,811. It's interesting to explore this in more depth, since we found this genre has some potential to work well on the App Store, and our aim is to recommend an app genre that shows potential for being profitable on both the App Store and Google Play.

Let's take a look at some of the apps from this genre and their number of installs:

```
In [30]: for app in google_final:
             if app[1] == 'BOOKS_AND_REFERENCE':
                 print(app[0], ':', app[5])

E-Book Read - Read Book for free : 50,000+
Download free book with green book : 100,000+
Wikipedia : 10,000,000+
Cool Reader : 10,000,000+
Free Panda Radio Music : 100,000+
Book store : 1,000,000+
FBReader: Favorite Book Reader : 10,000,000+
English Grammar Complete Handbook : 500,000+
Free Books - Spirit Fanfiction and Stories : 1,000,000+
Google Play Books : 1,000,000,000+
AlReader -any text book reader : 5,000,000+
Offline English Dictionary : 100,000+
Offline: English to Tagalog Dictionary : 500,000+
FamilySearch Tree : 1,000,000+
Cloud of Books : 1,000,000+
Recipes of Prophetic Medicine for free : 500,000+
ReadEra - free ebook reader : 1,000,000+
Anonymous caller detection : 10,000+
Ebook Reader : 5,000,000+
Litnet - E-books : 100,000+
Read books online : 5,000,000+
English to Urdu Dictionary : 500,000+
eBoox: book reader fb2 epub zip : 1,000,000+
English Persian Dictionary : 500,000+
Flybook : 500,000+
All Maths Formulas : 1,000,000+
Ancestry : 5,000,000+
HTC Help : 10,000,000+
English translation from Bengali : 100,000+
```

Pdf Book Download - Read Pdf Book : 100,000+
Free Book Reader : 100,000+
eBoox new: Reader for fb2 epub zip books : 50,000+
Only 30 days in English, the guideline is guaranteed : 500,000+
Moon+ Reader : 10,000,000+
SH-02J Owner's Manual (Android 8.0) : 50,000+
English-Myanmar Dictionary : 1,000,000+
Golden Dictionary (EN-AR) : 1,000,000+
All Language Translator Free : 1,000,000+
Azpen eReader : 500,000+
URBANO V 02 instruction manual : 100,000+
Bible : 100,000,000+
C Programs and Reference : 50,000+
C Offline Tutorial : 1,000+
C Programs Handbook : 50,000+
Amazon Kindle : 100,000,000+
Aab e Hayat Full Novel : 100,000+
Aldiko Book Reader : 10,000,000+
Google I/O 2018 : 500,000+
R Language Reference Guide : 10,000+
Learn R Programming Full : 5,000+
R Programing Offline Tutorial : 1,000+
Guide for R Programming : 5+
Learn R Programming : 10+
R Quick Reference Big Data : 1,000+
V Made : 100,000+
Wattpad [U+1F4D6] Free Books : 100,000,000+
Dictionary - WordWeb : 5,000,000+
Guide (for X-MEN) : 100,000+
AC Air condition Troubleshoot,Repair,Maintenance : 5,000+
AE Bulletins : 1,000+
Ae Allah na Dai (Rasa) : 10,000+
50000 Free eBooks & Free AudioBooks : 5,000,000+
Ag PhD Field Guide : 10,000+
Ag PhD Deficiencies : 10,000+
Ag PhD Planting Population Calculator : 1,000+
Ag PhD Soybean Diseases : 1,000+
Fertilizer Removal By Crop : 50,000+
A-J Media Vault : 50+
Al-Quran (Free) : 10,000,000+
Al Quran (Tafsir & by Word) : 500,000+
Al Quran Indonesia : 10,000,000+
Al'Quran Bahasa Indonesia : 10,000,000+
Al Quran Al karim : 1,000,000+
Al-Muhaffiz : 50,000+
Al Quran : EAlim - Translations & MP3 Offline : 5,000,000+
Al-Quran 30 Juz free copies : 500,000+
Koran Read &MP3 30 Juz Offline : 1,000,000+

Hafizi Quran 15 lines per page : 1,000,000+
Quran for Android : 10,000,000+
Surah Al-Waqiah : 100,000+
Hisnul Al Muslim - Hisn Invocations & Adhkaar : 100,000+
Satellite AR : 1,000,000+
Audiobooks from Audible : 100,000,000+
Kinot & Eichah for Tisha B'Av : 10,000+
AW Tozer Devotionals - Daily : 5,000+
Tozer Devotional -Series 1 : 1,000+
The Pursuit of God : 1,000+
AY Sing : 5,000+
Ay Hasnain k Nana Milad Naat : 10,000+
Ay Mohabbat Teri Khatir Novel : 10,000+
Arizona Statutes, ARS (AZ Law) : 1,000+
Oxford A-Z of English Usage : 1,000,000+
BD Fishpedia : 1,000+
BD All Sim Offer : 10,000+
Youboox - Livres, BD et magazines : 500,000+
B&H Kids AR : 10,000+
B y H Niños ES : 5,000+
Dictionary.com: Find Definitions for English Words : 10,000,000+
English Dictionary - Offline : 10,000,000+
Bible KJV : 5,000,000+
Borneo Bible, BM Bible : 10,000+
MOD Black for BM : 100+
BM Box : 1,000+
Anime Mod for BM : 100+
NOOK: Read eBooks & Magazines : 10,000,000+
NOOK Audiobooks : 500,000+
NOOK App for NOOK Devices : 500,000+
Browsery by Barnes & Noble : 5,000+
bp e-store : 1,000+
Brilliant Quotes: Life, Love, Family & Motivation : 1,000,000+
BR Ambedkar Biography & Quotes : 10,000+
BU Alsace : 100+
Catholic La Bu Zo Kam : 500+
Khrifa Hla Bu (Solfa) : 10+
Kristian Hla Bu : 10,000+
SA HLA BU : 1,000+
Learn SAP BW : 500+
Learn SAP BW on HANA : 500+
CA Laws 2018 (California Laws and Codes) : 5,000+
Bootable Methods(USB-CD-DVD) : 10,000+
cloudLibrary : 100,000+
SDA Collegiate Quarterly : 500+
Sabbath School : 100,000+
Cypress College Library : 100+
Stats Royale for Clash Royale : 1,000,000+

GATE 21 years CS Papers(2011-2018 Solved) : 50+
Learn CT Scan Of Head : 5,000+
Easy Cv maker 2018 : 10,000+
How to Write CV : 100,000+
CW Nuclear : 1,000+
CY Spray nozzle : 10+
BibleRead En Cy Zh Yue : 5+
CZ-Help : 5+
Modlitební knížka CZ : 500+
Guide for DB Xenoverse : 10,000+
Guide for DB Xenoverse 2 : 10,000+
Guide for IMS DB : 10+
DC HSEMA : 5,000+
DC Public Library : 1,000+
Painting Lulu DC Super Friends : 1,000+
Dictionary : 10,000,000+
Fix Error Google Playstore : 1,000+
D. H. Lawrence Poems FREE : 1,000+
Bilingual Dictionary Audio App : 5,000+
DM Screen : 10,000+
wikiHow: how to do anything : 1,000,000+
Dr. Doug's Tips : 1,000+
Bible du Semeur-BDS (French) : 50,000+
La citadelle du musulman : 50,000+
DV 2019 Entry Guide : 10,000+
DV 2019 - EDV Photo & Form : 50,000+
DV 2018 Winners Guide : 1,000+
EB Annual Meetings : 1,000+
EC - AP & Telangana : 5,000+
TN Patta Citta & EC : 10,000+
AP Stamps and Registration : 10,000+
CompactiMa EC pH Calibration : 100+
EGW Writings 2 : 100,000+
EGW Writings : 1,000,000+
Bible with EGW Comments : 100,000+
My Little Pony AR Guide : 1,000,000+
SDA Sabbath School Quarterly : 500,000+
Duaa Ek Ibaadat : 5,000+
Spanish English Translator : 10,000,000+
Dictionary - Merriam-Webster : 10,000,000+
JW Library : 10,000,000+
Oxford Dictionary of English : Free : 10,000,000+
English Hindi Dictionary : 10,000,000+
English to Hindi Dictionary : 5,000,000+
EP Research Service : 1,000+
Hymnes et Louanges : 100,000+
EU Charter : 1,000+
EU Data Protection : 1,000+

```
EU IP Codes : 100+
EW PDF : 5+
BakaReader EX : 100,000+
EZ Quran : 50,000+
FA Part 1 & 2 Past Papers Solved Free - Offline : 5,000+
La Fe de Jesus : 1,000+
La Fe de Jesús : 500+
Le Fe de Jesus : 500+
Florida - Pocket Brainbook : 1,000+
Florida Statutes (FL Code) : 1,000+
English To Shona Dictionary : 10,000+
Greek Bible FP (Audio) : 1,000+
Golden Dictionary (FR-AR) : 500,000+
Fanfic-FR : 5,000+
Bulgarian French Dictionary Fr : 10,000+
Chemin (fr) : 1,000+
The SCP Foundation DB fr nn5n : 1,000+
```

The Book and Reference genre includes a variety of apps: software for processing and reading ebooks, various collections of libraries, dictionaries, tutorials on programming or languages, etc. It seems there's still a small number of extremely popular apps that skew the average:

```
In [31]: for app in google_final:
             if app[1] == 'BOOKS_AND_REFERENCE' and (app[5] == '1,000,000,000+'
                                                     or app[5] == '500,000,000+'
                                                     or app[5] == '100,000,000+'):
                 print(app[0], ':', app[5])
```

```
Google Play Books : 1,000,000,000+
Bible : 100,000,000+
Amazon Kindle : 100,000,000+
Wattpad [U+1F4D6] Free Books : 100,000,000+
Audiobooks from Audible : 100,000,000+
```

However, it looks like there are only a few very popular apps, so this market still shows potential. Let's try to get some app ideas based on the kind of apps that are somewhere in the middle in terms of popularity (between 1,000,000 and 100,000,000 downloads):

```
In [33]: for app in google_final:
             if (app[1] == 'BOOKS_AND_REFERENCE') and (app[5] == '1,000,000+'
                                                     or app[5] == '5,000,000+'
                                                     or app[5] == '10,000,000+'
                                                     or app[5] == '50,000,000+'):
                 print(app[0], ':', app[5])
```

```
Wikipedia : 10,000,000+
Cool Reader : 10,000,000+
```

Book store : 1,000,000+
FBReader: Favorite Book Reader : 10,000,000+
Free Books - Spirit Fanfiction and Stories : 1,000,000+
AlReader -any text book reader : 5,000,000+
FamilySearch Tree : 1,000,000+
Cloud of Books : 1,000,000+
ReadEra - free ebook reader : 1,000,000+
Ebook Reader : 5,000,000+
Read books online : 5,000,000+
eBoox: book reader fb2 epub zip : 1,000,000+
All Maths Formulas : 1,000,000+
Ancestry : 5,000,000+
HTC Help : 10,000,000+
Moon+ Reader : 10,000,000+
English-Myanmar Dictionary : 1,000,000+
Golden Dictionary (EN-AR) : 1,000,000+
All Language Translator Free : 1,000,000+
Aldiko Book Reader : 10,000,000+
Dictionary - WordWeb : 5,000,000+
50000 Free eBooks & Free AudioBooks : 5,000,000+
Al-Quran (Free) : 10,000,000+
Al Quran Indonesia : 10,000,000+
Al'Quran Bahasa Indonesia : 10,000,000+
Al Quran Al karim : 1,000,000+
Al Quran : EAlim - Translations & MP3 Offline : 5,000,000+
Koran Read &MP3 30 Juz Offline : 1,000,000+
Hafizi Quran 15 lines per page : 1,000,000+
Quran for Android : 10,000,000+
Satellite AR : 1,000,000+
Oxford A-Z of English Usage : 1,000,000+
Dictionary.com: Find Definitions for English Words : 10,000,000+
English Dictionary - Offline : 10,000,000+
Bible KJV : 5,000,000+
NOOK: Read eBooks & Magazines : 10,000,000+
Brilliant Quotes: Life, Love, Family & Motivation : 1,000,000+
Stats Royale for Clash Royale : 1,000,000+
Dictionary : 10,000,000+
wikiHow: how to do anything : 1,000,000+
EGW Writings : 1,000,000+
My Little Pony AR Guide : 1,000,000+
Spanish English Translator : 10,000,000+
Dictionary - Merriam-Webster : 10,000,000+
JW Library : 10,000,000+
Oxford Dictionary of English : Free : 10,000,000+
English Hindi Dictionary : 10,000,000+
English to Hindi Dictionary : 5,000,000+

So these apps seem to be dominated by software for processing and reading ebooks, as well as various collections of libraries and dictionaries. Because they are so similar it's probably not a good idea to build similar apps.

It seems that taking a popular book (perhaps a more recent book - maybe from the NY best-sellers list, etc) and turning it into an app could be profitable for both the Google Play and the App Store markets.

To make our app unique we would need to add some special features such as daily quotes from the book, an audio version of the book, quizzes on the book, a forum where people can discuss the book, etc.

## 11 Conclusions

In this project, we analyzed data from the Apple App Store and Google Play Store with the goal of recommending an app profile that can be profitable for both markets.

Our findings indicate that taking a recently popular book and turning it into an app could be profitable for both markets. However, these markets are already full of libraries so we need to add some unique features to draw customer attention such as an audio version of the book, quizzes, daily quotes, a forum where people can discuss the book, etc.

```
In [ ]:
```