Preliminary Report

June 6, 2019

1 COMP 499 Introduction to Data Analysis

2 Preliminary Report

2.1 Presented By: Genevieve Plante-Brisebois 40003112

2.2 Presented To : Greg Butler

2.3 Thursday June 6, 2019

2.3.1 Introduction of the dataset and report

This is the prelimminary report for the course Data Analysis in Summer 2019 presented by Genevieve Plante-Brisebois. The dataset that is chosen to do the project is Google Play Store Apps taken from this website: https://www.kaggle.com/lava18/google-play-store-apps . The information in this dataset covers each app in respect with their name, rating, reviews, number of downloads, type, price, file size, content rating, genres, last version/update made, current version and the android version needed in order to run the apps. The dataset contains 9660 records. It is comprised of the csv document for the general information of the applications as described above and a csv document comprised of the reviews and given ratings to the applications. The Review csv is taking the top 100 reviews from each application which has received reviews.

Using these datasets, there are multiple questions that I will attempt to answer and, as the answers are found, they will become the basis for the next one. This will be done in three stages.

1- What is more popular and fruitfull between free and paid applications and predict the present and upcoming trend?

To answer that question I will be making exploratory analysis of the data. I will be taking a look at the data and compare the number of downloads of each application and then try to see what is the one coming up on top in terms of poopularity by combining with the ratings.

For an information purpose, the revenue of the paid apps will be calculated as it could help to see if it is profitable. However, it is impossible to get the information about the revenue for the free applications. If it would be possible to find the data of the revenue from the free applications with the adds it would be a good way to enrich the dataset and add another feature to analyse the set.

2- What are the rising trends in the app type that will have won the previous question?

This will try to answer if there are categories and target audiences that are more proeminent and are the trend or upcomming trends. This will again be done by doing exploratory data analysis. Just like with the previous question, it will be used to narrow down the amount of data that

will be used in the next step. By further narowing our criterias, we will be able to bring ourselves to the last step of this process.

3- Predict the impact of the reviews and rating on the number of downloads.

From the selected apps in the previous questions, we will take a look at the reviews dataset. Using exploratory data analysis and natural language processing, we will try to create new feature that will hopefully allow us to see if there is a link between the reviews, the ratings and the success of an application in the trending categories. The success we are looking at for the apps in this case is the number of downloads. From the information found here we will try to predict the influence and potentially the growth of the applications that are present in these categories.

2.3.2 Steps to Data retrieval

In order to retrievee the data from this set, these are the steps to follow.

1-Go to the www.kaggle.com website and create an account if you do not have one yet.

2-Then there will be a need to go to the https://www.kaggle.com/lava18/google-play-store-apps page and click on the download button at the top of the page. If you desire to have more information about the data set, you can read the page and see the information and summary that has been provided for this data set.

3- Make sure that you are creating a data folder where you are having the jupyter book and in that data folder extract all the concerned information that is in the zip folder. You should be able to have the googleplaystore.csv and the googleplaystore_user_reviews.csv file.

Now in order to use the data directly via scripting, the data is also available by doing the following lines in python

```
In [1]: import urllib.request
        urllib.request.urlretrieve("https://drive.google.com/drive/folders/17Ad0l9iWsOooz4AGA6
        urllib.request.urlretrieve("https://drive.google.com/drive/folders/17Ad0l9iWsOooz4AGA6

Out[1]: ('googleplaystore_user_reviews.csv',
         <http.client.HTTPMessage at 0x1a095d36e10>)
```

2.3.3 Data Wrangling

In this section the data wrangling will be done in order to clean the datasets as much as possible and to be able to do our analysis later. I will only note that the natural processing of the review data set will not be done now as this will be the whole task of the question 3, building the features and make the prediction.

Importing Data, Data Types and Formatings the Dataframes

```
In [2]: import pandas as pd


        google = pd.read_csv("data/googleplaystore.csv")
        print(google.columns)
        review = pd.read_csv('data/googleplaystore_user_reviews.csv')
        #get data types
        print(google.dtypes)
        review.dtypes
```

```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
       'Android Ver'],
      dtype='object')
App                object
Category           object
Rating            float64
Reviews             int64
Size               object
Installs           object
Type               object
Price             float64
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

```
Out[2]: App                        object
        Translated_Review          object
        Sentiment                  object
        Sentiment_Polarity        float64
        Sentiment_Subjectivity    float64
        dtype: object
```

Now that we have the columns and the data types that we have to deal with, we can see if we need to cast some of the data types. From the data types here, we can see that we need to cast a few of them such as the price, number of reviews.

```
In [3]: google.Reviews.astype('int64')
        google.Price.astype('float64')
        google.dtypes
```

```
Out[3]: App                object
        Category           object
        Rating            float64
        Reviews             int64
        Size               object
        Installs           object
        Type               object
        Price             float64
        Content Rating     object
        Genres             object
        Last Updated       object
        Current Ver        object
        Android Ver        object
        dtype: object
```

From the data set of the reviews, we can see that there is the sentiment polarity and the sentiment subjectivity columns, however, there is no scale prrovided in the kaggle page so that we know what these data use as scales. As we do not know the scales, these columns will be dismissed from the data frame that we will be using for the project.

```
In [4]: review = review[['App', 'Translated_Review', 'Sentiment']]
        review.head()
```

```
Out[4]:                      App                                  Translated_Review  \
        0  10 Best Foods for You  I like eat delicious food. That's I'm cooking ...
        1  10 Best Foods for You    This help eating healthy exercise regular basis
        2  10 Best Foods for You                                                NaN
        3  10 Best Foods for You         Works great especially going grocery store
        4  10 Best Foods for You                                       Best idea us

          Sentiment
        0  Positive
        1  Positive
        2       NaN
        3  Positive
        4  Positive
```

Data Cleaning Now we can try to see if we have any nan values for all the columns

```
In [5]: print(google.isnull().sum(),'\n')
        print(review.isnull().sum())
```

```
App                  0
Category             0
Rating            1474
Reviews              0
Size                 0
Installs             0
Type                 1
Price                0
Content Rating       0
Genres               1
Last Updated         0
Current Ver          8
Android Ver          2
dtype: int64

App                     0
Translated_Review    26868
Sentiment            26863
dtype: int64
```

Now that we kow which values have some nan values, we can look into the means, for those that are applicable and see how we are going to handle the data in order to know how we are going to handle the nan values.

For the review dataset, the only way to really handle it will be to take out the rows that have nan values as there are too many nan values. Moreover, for the reviews, we will be focussing later on natural language processing, which means that we do not want nan values and we only want to evaluate the reviews where a comment was left behind. The other reviews were most likely simply someone who decided to only give a star rating and no commentary review. As both rating from the google dataset and the comment reviews left will be taken into account in order to try to make predictions, taking away the nan rows from the review dataset is ok.

For the general app dataset, it will be possible to use mean values or default values in order to take care of those nan values. Once we take care of the nan values we will be able to normalize.

```
In [6]: review = review.dropna()

        google.Rating.describe()

Out[6]: count    9367.000000
        mean        4.193338
        std         0.537431
        min         1.000000
        25%         4.000000
        50%         4.300000
        75%         4.500000
        max        19.000000
        Name: Rating, dtype: float64

In [7]: mean_r = google.Rating.mean()
        google['Rating'].fillna(value=mean_r, inplace = True)

In [8]: print(google.isnull().sum(),'\n')

App                0
Category           0
Rating             0
Reviews            0
Size               0
Installs           0
Type               1
Price              0
Content Rating     0
Genres             1
Last Updated       0
Current Ver        8
Android Ver        2
dtype: int64
```

```
In [9]: google.Type.describe()

Out[9]: count     10840
        unique        2
        top        Free
        freq      10040
        Name: Type, dtype: object

In [10]: google['Type'].unique()

Out[10]: array(['Free', 'Paid', nan], dtype=object)
```

For the last nan values in the set, the values cannot be assumed. If we do not have a version of the software or the version of Android needed we cannot simply guess those values. However, as there are only a few data that contain nan, 12, and that our dataset is quite large, 9000+ , it will not have a major impact to remove those nan values.

```
In [11]: google = google.dropna()
         print('Google Store General Information Dataset:\n')
         print(google.isnull().sum(),'\n')

         print(google.dtypes, '\n\n')

         print('Reviews of the applications dataset: \n')
         print(review.isnull().sum(), '\n')

         print(review.dtypes)

Google Store General Information Dataset:

App               0
Category          0
Rating            0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
Content Rating    0
Genres            0
Last Updated      0
Current Ver       0
Android Ver       0
dtype: int64

App                object
Category           object
Rating            float64
Reviews             int64
```

```
Size               object
Installs           object
Type               object
Price             float64
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

```
Reviews of the applications dataset:

App                 0
Translated_Review    0
Sentiment           0
dtype: int64

App                 object
Translated_Review    object
Sentiment           object
dtype: object
```

Now we are going to verify if there are duplicates since we only want to have the data once. This step will only be using the data from the google set. The review set has no record ID and there is no way of knowing if it is simply multiple reviews with similar or same comments for an application or if it is duplicates. As such, we will make the assumption that all the records are unique.

```
In [12]: print('Before droping the duplicates')
         google.App.duplicated().sum()
```

```
Before droping the duplicates
```

```
Out[12]: 1181
```

```
In [13]: #drop the dulicates

         google = google.drop_duplicates(subset='App', keep='first')
         print('After droping the duplicates')
         google.App.duplicated().sum()
```

```
After droping the duplicates
```

```
Out[13]: 0
```

```
In [14]: print(google.head())
         review.head()
```

```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                               Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite  FREE Live Cool Themes, Hide ...  ART_AND_DESIGN     4.7
3                             Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

   Reviews  Size     Installs  Type  Price Content Rating  \
0      159   19M      10,000+  Free    0.0       Everyone
1      967   14M     500,000+  Free    0.0       Everyone
2    87510  8.7M   5,000,000+  Free    0.0       Everyone
3   215644   25M  50,000,000+  Free    0.0           Teen
4      967  2.8M     100,000+  Free    0.0       Everyone

                      Genres Last Updated         Current Ver   Android Ver
0               Art & Design     7-Jan-18               1.0.0  4.0.3 and up
1  Art & Design;Pretend Play    15-Jan-18               2.0.0  4.0.3 and up
2               Art & Design     1-Aug-18               1.2.4  4.0.3 and up
3               Art & Design     8-Jun-18  Varies with device    4.2 and up
4    Art & Design;Creativity    20-Jun-18                 1.1    4.4 and up
```

Out[14]:

	App	Translated_Review	Sentiment
0	10 Best Foods for You	I like eat delicious food. That's I'm cooking ...	Positive
1	10 Best Foods for You	This help eating healthy exercise regular basis	Positive
3	10 Best Foods for You	Works great especially going grocery store	Positive
4	10 Best Foods for You	Best idea us	Positive
5	10 Best Foods for You	Best way	Positive

Normalizing Now that we have all the data structured and cleaned up, we can take a look at what needs to be normalized. This will help if we ever want to combine the sets or want to enrich the sets later on. By have the normalization done it will help lower the number of possible inconsistencies.

The sets will need a little bit of cleaning, as to make the text as uniform as possible for what we are trying to do in the next steps of the project.

```
In [15]: #this function can be used on names and sentences and also on categories such as sent
         def preprocess_text(name):
```

```
        name = name.lower()
        name = name.replace(',', ' ')
        name = name.replace("'", '')
        name = name.replace('&', 'and')
        name = name.replace('?', '')
        name = name.replace('_', ' ')
        #name = name.decode('utf-8', 'ignore')
        return name.strip()


google['App'] = google['App'].map(preprocess_text)
google['Category'] = google['Category'].map(preprocess_text)
google['Type'] = google['Type'].map(preprocess_text)
google['Genres'] = google['Genres'].map(preprocess_text)
google['Content Rating'] = google['Content Rating'].map(preprocess_text)

review['App'] = review['App'].map(preprocess_text)
review['Translated_Review'] = review['Translated_Review'].map(preprocess_text)
review['Sentiment'] = review['Sentiment'].map(preprocess_text)
```

```
In [16]: google.App.duplicated().sum()
```

```
Out[16]: 25
```

```
In [17]: #we have found more duplicates now that we have normalized the settings so we are goi

         google = google.drop_duplicates(subset='App', keep='first')
         print('After droping the duplicates created by normalizing')
         google.App.duplicated().sum()
```

```
After droping the duplicates created by normalizing


Out[17]: 0
```

Final restructuration of the datasets: As we are now done with normalizing and cleaning, we are going to finish up some last adjustments. In the annalysis that we wil have to do, we will not use the data from the size of the app, the current version, the last time it was updated nor the android version. As these data will not be used in the analysis, we are going to take them out from our working data frame.

```
In [18]: google = google[['App', 'Category', 'Rating', 'Reviews', 'Installs', 'Type', 'Price',
```

2.3.4 Dataset Descriptions

Now that we have cleaned and normalized our datasets, we can take a look at what they are.

Here is the set of the general information of the google store applications:

```
In [19]: google.head(10)
```

Out[19]:

	App	Category	Rating
0	photo editor and candy camera and grid and scr...	art and design	4.1
1	coloring book moana	art and design	3.9
2	u launcher lite free live cool themes hide ...	art and design	4.7
3	sketch - draw and paint	art and design	4.5
4	pixel draw - number art coloring book	art and design	4.3
5	paper flowers instructions	art and design	4.4
6	smoke effect photo maker - smoke editor	art and design	3.8
7	infinite painter	art and design	4.1
8	garden coloring book	art and design	4.4
9	kids paint free - drawing fun	art and design	4.7

	Reviews	Installs	Type	Price	Content Rating	Genres
0	159	10,000+	free	0.0	everyone	art and design
1	967	500,000+	free	0.0	everyone	art and design;pretend play
2	87510	5,000,000+	free	0.0	everyone	art and design
3	215644	50,000,000+	free	0.0	teen	art and design
4	967	100,000+	free	0.0	everyone	art and design;creativity
5	167	50,000+	free	0.0	everyone	art and design
6	178	50,000+	free	0.0	everyone	art and design
7	36815	1,000,000+	free	0.0	everyone	art and design
8	13791	1,000,000+	free	0.0	everyone	art and design
9	121	10,000+	free	0.0	everyone	art and design;creativity

In [20]:
```
google.dtypes
```

Out[20]:
```
App               object
Category          object
Rating           float64
Reviews            int64
Installs          object
Type              object
Price            float64
Content Rating    object
Genres            object
dtype: object
```

```
In [21]: print(google.App.describe(),'\n')
         print(google.Category.describe(),'\n')
         print(google.Rating.describe(),'\n')
         print(google.Reviews.describe(),'\n')
         print(google.Installs.describe(),'\n')
         print(google.Type.describe(),'\n')
         print(google['Content Rating'].describe(),'\n')
         print(google.Genres.describe(),'\n')
```

```
count                            9623
unique                           9623
top       skout - meet  chat  go live
freq                                1
Name: App, dtype: object

count       9623
unique        33
top       family
freq        1822
Name: Category, dtype: object

count    9623.000000
mean        4.176849
std         0.494663
min         1.000000
25%         4.000000
50%         4.200000
75%         4.500000
max         5.000000
Name: Rating, dtype: float64

count    9.623000e+03
mean     2.168584e+05
std      1.834069e+06
min      0.000000e+00
25%      2.500000e+01
50%      9.750000e+02
75%      2.949000e+04
max      7.815831e+07
Name: Reviews, dtype: float64

count           9623
unique            20
top       1,000,000+
freq            1415
Name: Installs, dtype: object

count     9623
```

```
unique       2
top       free
freq      8873
Name: Type, dtype: object

count         9623
unique           6
top       everyone
freq          7870
Name: Content Rating, dtype: object

count      9623
unique      118
top       tools
freq        820
Name: Genres, dtype: object
```

```
In [22]: review.head(10)
```

```
Out[22]:                         App                                Translated_Review  \
         0   10 best foods for you  i like eat delicious food. thats im cooking fo...
         1   10 best foods for you    this help eating healthy exercise regular basis
         3   10 best foods for you         works great especially going grocery store
         4   10 best foods for you                                    best idea us
         5   10 best foods for you                                        best way
         6   10 best foods for you                                         amazing
         8   10 best foods for you                              looking forward app
         9   10 best foods for you              it helpful site ! it help foods get !
         10  10 best foods for you                                       good you.
         11  10 best foods for you  useful information the amount spelling errors ...

            Sentiment
         0   positive
         1   positive
         3   positive
         4   positive
         5   positive
         6   positive
         8    neutral
         9    neutral
         10  positive
         11  positive
```

```
In [23]: review.dtypes
```

```
Out[23]: App                  object
         Translated_Review    object
```

```
Sentiment            object
dtype: object
```

```
In [24]: print(review.App.describe(),'\n')
         print(review.Translated_Review.describe(),'\n')
         print(review.Sentiment.describe(),'\n')
```

```
count          37427
unique           865
top       bowmasters
freq             312
Name: App, dtype: object

count     37427
unique    27866
top        good
freq        296
Name: Translated_Review, dtype: object

count        37427
unique           3
top       positive
freq         23998
Name: Sentiment, dtype: object
```

2.3.5 Summary

As a summary and ending note for this preliminary report here are the guidelines of the questions that we are going to be asked and that we are looking for answers to. For the questions that we want to answer they are:

1- What is more popular and fruitfull between free and paid applications and predict the present and upcoming trend?

2- What are the rising trends in the app type that will have won the previous question?

3- Predict the impact of the reviews and rating on the number of downloads.

Now we are using two main dataset, google play general information and reviews information. However, for the first two questions we wil be using solely the first dataset. The second dataset will be used duging the third phase and we will try to use natural processing to make new features and try to see how the comments and ratings might affect and lead to a more successfull application.

A note to keep in mind, as the exploratory data analysis will be done, the first dataset might be enriched with other datasets that were inspired by that dataset and that have even more data. Those datasets mght be used for testing or validation purposes but are not the main ones. Their use will be evaluated in more details as we go further in the analysis.

The dataset that might be used for enrichment, validation or testing is the following: https://www.kaggle.com/gauthamp10/google-playstore-apps