Data Wrangling

Capstone Project # 1   - Predicting Popular Video Genres

Data Set

The data set used for this project comes from Kaggle. There are a number of csv and json files available on Kaggle under the name "Trending YouTube Videos Statistics". For this project, csv files are being utilized. The dataset includes data on trending videos. Data is included for several countries including the USA and Mexico. Data includes 16 columns (fields) in its raw form for both the USA and Mexico videos. There are a total of 40,949 records found in the USA csv file and 40,451 records found in Mexico csv files. The data for the most part is clean.

Data Wrangling Code

https://github.com/fariha23/Capstone1

Data Wrangling

Steps:

1. **Removing unnecessary columns**
   After loading files in Pandas as dataframes (namely, usa_df and mx_df) and looking at the output of ".info()" method on both dataframes (df) it is observed that no null records in either of the dfs. The attribute ".columns" shows that following are the columns included in the data.

   'video_id', 'trending_date', 'title', 'channel_title', 'category_id', 'publish_time', 'tags', 'views', 'likes', 'dislikes', 'comment_count', 'thumbnail_link', 'comments_disabled', 'ratings_disabled', video_error_or_removed', 'description'

   The aim of the project is to predict which genre (category) of videos is most popular in different countries. In order to make only relevant fields available for analysis, following fields(columns) were dropped from original dfs. To drop these columns ".drop()" method was used.

   'trending_date' ,'video_id','thumbnail_link','description','tags','channel_title'

2. **Adding "category_name" column**
   One important field missing from both csv files is category name ('category_name'). The reason is that every country's data doesn't have the same 1:1 translation between category_id and category_name. To solve this issue, a json file for the categories for the

USA and Mexico was downloaded from www.googleapi.com/youtube/v3/. After downloading the response using "requests.get()", the response was converted to a json nested dictionary using ".json()" method. The key "items" in the nested dictionary has the translation of interest i.e cateogory_id: category_name. A simple "for" loop function is written to extract that dictionary out. The id field is however a string and therefore it needed to be changed into integer in order to use .map() method on the data frame to add an additional column of "column_name" to the data frame.

After this change the dictionary with category_id and category_name was ready to be used with .map() method to the USA and Mexico data frames.

3. **Dropping rows with NaN values**
   After the new column was added to both dataframes, the USA data frame shows that there are no null columns. The Mexico data frame shows 252 rows of NaN. Upon investigating, it was found that the Mexico data frame has a category id of 29 whereas the translation dictionary does not contain that specific number. Therefore a NaN was written in those rows under column "category_name" of the data frame. After reviewing several of these 252 videos it is determined that these videos do not belong to one specific category. Therefore these rows will be dropped from the Mexico data frame. Since there are 40K+ rows of data in the data frame, the impact of dropping 252 rows will not be significant when the prediction model is built for the project. To drop these rows, ".drop()" method is used with a boolean condition for the specific null values in column "category_name". To find nulls, ".isnull ()" method is used.

4. **Outliers**
   Outliers are extreme values in a data set that makes the data pattern skew. These values are sometimes warranted . For example highly popular YouTube videos can acquire a couple of millions of views and for the purposes of this project those outliers can help determine the most popular category of videos. To find the outliers, boxplot was graphed using Seaborn Library,

   ```
   sns.boxplot(x=usa_df_clean['views'])
   ```

   Summary statistics were also collected on both cleaned data frames using .describe() methods. Both results can be found in the jupyter notebook saved under the github url provided above.

5. **Appending Mexico Data Frame with USA Data Frame**
   In order to keep things simple, appended Mexico's data frame under the USA's data frame. To do so a column 'country' was added to both data frames with value 'USA' and 'Mexico'. Following command was used to append the two data frames

   ```
   combined_usa_mx_df=usa_df_clean.append(mx_df_clean)
   ```