Capstone Project 1: Data Wrangling/Data Story Report

Springboard 2020

Joseph Tran

Data Wrangling

The data wrangling report encompasses information about the dataset used for the project of building a music recommendation system, and the steps taken to prepare the data for a machine learning model. The dataset chosen was the #nowplaying-RS dataset and was published in Jan. 2020. It was constructed by scraping song and hashtag information from Twitter with the intention of creating a dataset capable of context and content aware recommendation systems, was created by Eva Zangerle, from the University of Innsbruck Austria. The #nowplaying-RS dataset is comprised of three separate files: a listening event dataset that includes user id's, track id's, hashtag, and timestamp that contains roughly 17 million listening events. A sentiment file that contains sentiment scores for the hashtags. Lastly, the dataset also includes a file containing various information about context and content. Mainly, it includes the Spotify music attributes obtained from the Spotify API.

The main object required for a non-negative matrix factorization algorithm is the user-item matrix. This can be obtained solely from the listening event file. However, the three files will be merged so all information is contained in one dataframe. Merging the files will also help to reduce observations and thus, reduce the sparsity of the user-item matrix.

The main file is the listening event file that contains user id's, track id's, hashtags, and timestamps. I chose to work with this file first as it contains the information needed to create the user/item matrix necessary for a matrix factorization model. The data was inspected by calling some of the dataframe attributes and was found to have 138,223 unique users and 344,536 unique songs, with a shape of (17560113,4). Inspection of the head reveals identical rows. This was identified by looking at the 'created_at' column and seeing matching timestamps. These duplicate rows were generated from a 'one to many' relationship with a listening event and hashtags. The duplicate rows will be preserved until the datasets are merged. It is advantageous to save all the corresponding hashtags associated with a listening event. Dropping the duplicate rows at this time would discard relevant hashtags. Next, null values were removed as there was only a single null value in the hashtag column.

The next file that was worked on was the sentiment values dataset. The file contains sentiment scores from four popular sentiment dictionaries; AFFIN, Opinion Lexicon, SentiStrength, and Sentiment Hashtag Lexicon. Scores were only assigned to unique hashtags that were scored by at least one of the sentiment dictionaries. In addition to the score, minimum, maximum, sum, and average scores for the four dictionaries are included. Of the four dictionaries used, scores were only assigned to 5290 of the 32208 unique hashtags in the main file. Upon inspecting the head of this dataset, it was found that columns were mis-aligned and nested in a hierarchical index under the column name ‘hashtag.’ After inspection it was found that the first four columns did not have column names and somehow got thrown into the multi index. To fix this, column names were created with the dictionary name abbreviation and score, indicating that the column is the score generated from the dictionary. The new column names are assigned to the column in the order the dictionaries appear in the dataset. The multi-index column name ‘hashtag’ is renamed to ‘ss_score’ indicating the score for the sentiment dictionary. Afterwords, the new column names are assigned to the different levels of the multi-index, and the index is reset to create a standard range index. This is verified by calling type on the index. The only information that is desired is the hashtag, sentiment score, and average. Some of the sentiment scores have null values so the average score column is retained to impute those missing values row-wise, rather than imputing the mean, or some other transformation column-wise. A key-error was produced on the first attempt to drop the columns. Upon inspection, it was found that many of the field names contained leading white spaces. The columns are renamed to a standard form and then dropped from the dataset. When checking the null values of the resulting dataframe, it was found that the score columns for each dictionary contained 1423 null values. Instead of taking the mean of the column and imputing, the average score for each dictionary is imputed row-wise from the corresponding average score column. After the imputation, the null values are put into a bar chart and it is found that the ol_score resulted in the least amount of null values. This column is selected for our data and the remaining column dropped, afterwhich the null values are dropped from the data. The resulting dataframe consists only of the ‘hashtag’ and ‘ol_score’ columns, with all null values dropped from ‘ol_score,’ leaving 4831 unique hashtags, and their score from the Opinion Lexicon sentiment dictionary.

The last file to be worked on was the context/content file containing Spotify attributes and other contextual information. The columns: 'coordinates','id','place','geo','entities','time_zone' are dropped as they are not necessary for our model. The ‘tweet_lang’ and ‘lang’ columns will also be dropped, but first the dataset is filtered based on English. Once the filtering has been done these two columns are also dropped.

The next data wrangling process was to merge the three datasets. First, the listening event dataset is merged with the sentiment dataset with an inner join on ‘hashtag.’ This merged dataset is called ‘temp_df.’ Finally, all three sets are combined by merging the content/context dataset with temp_df with an inner join on 'track_id','created_at','user_id.'

With all the data in a single frame we can now preserve the hashtags and drop the duplicated rows. This is achieved by creating a dictionary to store the hashtags associated with a listening event. The values are added to the dataframe by mapping the dictionary keys to the user_id and track_id in the dataframe. After the new column containing the hashtags is added to the dataframe, the duplicate rows are dropped.

This is the point in the data wrangling process that we filter the track_id's so that each track appears no less than 10 times. A temporary dataframe is built from a value_count operation that limits the track count to be at least 10. The track_id count dataframe is then joined to the main dataframe on the track_id column, returning the track_id's with 10 or more plays.

Lastly, the user_id, artist_id and track_id are assigned category codes for sequential ordering.

Overall, the length of the final dataset is reduced from 17.6 million listening events to 4.4 million, and contains columns from all three files in the dataset. The final dataframe is saved as a csv titled 'NPRS_FINAL.csv.'

During the data wrangling process attempts we made to pull song metadata from Spotify API. The presence of the content features alerted me to the fact that these features were pulled from Spotify, and suggested I could use the track_id to pull the extra data. However, after spending several hours learning to connect to the Spotify API, I learned that the track_id and song_id were not Spotify labels, but hashed values intended to protect private information. Thus, I was not able to pull any metadata for the tracks. I reached out to the authors of the dataset, but they were unwilling to provide the hash keys.

Data Story

Since the only object necessary for a basic matrix factorization algorithm is the user-item matrix, it is difficult to weave a story with only that information. However, the final dataset from the data wrangling process contained numeric content features that we can plot as time series to look at the behavior over time. In addition, there is the categorical information we can plot and look at to find descriptive characteristics.

The first plot is a tabulation of total listening events per day. A vertical marking the day with the highest song count is plotted for reference. The plot has a lot of dips in song count over the year 2014 time period. It is hypothesized that the dip in song count is due to the web scraper breaking, being fixed, and collecting information again. I apply a 7 day rolling average to the data for smoothing in the next plot.

The next eight plots in the notebook are time series plots for the daily average value of the feature plotted over the time period. The indication is that the population listens to popular music. I would expect to see features such as 'instrumentalness' to have low values, as a value closer to 1 indicates a track with no vocals. This would mean genres like classical or jazz, that are not popular. I expect that the mean daily value for this feature is low. The rest of the plots for the Spotify features follow this logic and suggest what one may think about the listening habits of the masses: on the whole, there is no tendency to radical listening habits.

In addition to the content features, bar plots are made for the user_id, track_id, artist_id, and hashtag combinations. For the matrix factorization model, the user_id and track_id represent the most relevant information as these are the values in the user item matrix. I expected these charts to be boring and not reveal anything interesting, but I was mistaken. Plotting the counts for each user it was revealed that one user, user '303' was not an actual user, but some time of bot. Looking more closely at the user revealed that the time series plot of LE's generated by the 'user' was flat throughout the time period with around 303 LE's recorded per day. The user being assigned a code of '303' and the count per day of the user being 303 is a weird coincidence as the user code was generated from the hashed user ID. Being that a bot was discovered, this 'user' will be removed prior to constructing the user-item matrix.

The artist ID bar plot revealed that the artist with the most occurrences was artist '7582.' This artist appeared more than 55,000 times, with a unique song count of 79 different songs. It would be interesting to see who this is, but it is not possible with the data presented.

The bar plot of the hashtag combinations show that the single hashtag 'nowplaying' occurs, by itself, way more than any other hashtag. The single hashtag of now playing is removed to view the next highest count of hashtag combinations. Many of these combinations also include 'nowplaying' with another hashtag. The hashtag combinations show that the majority of the time users are listening to some type of rock related music, whether it be punk, classic, doommetal, or rockmusic.

Lastly, some correlation plots are made for selected content features that would intuitively seem correlated: energy/loudness, energy/danceability, energy,valence, and speechiness/instrumentalness. Of the plots generated, only one showed some type of correlation which was energy/loudness. This makes sense as loudness is proportional to energy. The correlation seems to be vaguely linear as both features increase in value.