ISE599 Final Report

Weihao Ding, Yiqing Fan

November 2019

Link: `https://github.com/WeihaoDing/FinalProject-ISE599.git`

1 Executive Summary

- **Problem Statement:**
  - **Motivation and Background**: Nowadays, there are more and more music applications in our life such as Spotify, Apple music and Pandora etc. The members in our team are both fond of listening to music and also viewing many different music playlists in those music applications. Those playlists are mostly been created by different classifications such as by genre, moods or places. Our team is motivated by the curiosity about how those playlists been made. Thus, we want to explore different methods to create different categories of music playlists based on Amazon Product Datasets(two different datasets inside it) and to learn more about the process regarding creating music playlists.
  - **Dataset:** We're using two datasets in this project. One contains product reviews and metadata from amazon, including 142.8 million reviews spanning Mat 1996 – July 2014. This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).The other is Metadata which includes descriptions, price, sales-rank, brand info, and co-purchasing links.
- **Description of Solution:**
  In our project, we mainly used two concepts of machine learning to achieve the best result in our case. First one is the rule-based machine learning, due to the special property of our project, we apply three different rule-based machine learning methods to create music playlists. The other is based on the natural language processing technique, we built TF-IDF model to do the data analyzing and then to create music playlists. In each method mentioned above, we build five different music playlists by choosing different music genre. Therefore, we created 20 music playlists in total in this project. We then calculate the average rating of each song and verifying the music genre of that song by using another dataset. After finding each song's average rating and music genre, we then have the way to decide which is the best method to get the good quality music playlists that belong to one certain genre and also have high ratings.
- **Summary Of Key Findings:**
  In the project, we created four music playlists inspired by rule-based machine learning method and natural language processing technique. After comparison, we find out music playlists based on by using *also bought* and calculating token weights, with scores 4.753 out of 5.0 and 4.746 out of 5.0 respectively, rank higher than music playlists based on other methods. However, 'also bought' method does not give good control on the category, which leads to the irrelativeness of some songs in the play list. So, we conclude that, the music playlist which benefited from natural language processing (i.e., TF-IDF), is the best one, 100% fitting the playlist's topic.Inspired by the finding, we highlight the significance of review text in helping create music playlists. Especially, using TF-IDF is conceiving to find keywords, which will give the clear explanation of what this song talks about. This is the most important step in the process of creating a high-quality music playlist.

2 Proposed Approach

1) **Random Pick:**
In this method, we're using Pandas dataframe.sample() method to generate 5 random different music playlists. Resulting output are as follows.
Explanation: The tables provided below are all in ProductID(s) of songs instead of the names of those selecting songs. Since we're using two different datasets in our project and we need to combine these two datasets via the productID(s), we found that ProductID(s) will be more meaningful in this stage. Also, the names of songs are not provided in both two datasets that we use, nevertheless, we will provide the names of songs in our final results.

Table 1: Random Pick Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B0011Z3DN4	B00136LUKO	B000005AM7	B0092YPJDI	B00000AGMC
2	B009D9L77Y	B008FOQH6Y	B008P5MLL8	B0002W4T4O	B003U7V8PG
3	B007DIQAXI	B008CWEPDG	B004H6QJVC	B000VT7AF8	B00123HPUC
4	B001AJFE5G	B00GD2L59O	B0027WNRNQ	B00005ICAW	B00014TQ7S
5	B00CAZOHDO	B001G93Z5Q	B00E9E4EC8	B00A0A5A6Y	B000W1MDAM
6	B007B6VOKQ	B00005A8MT	B0000026WD	B0006ZQ9BS	B008K9SG9K

2) **Search By Keyword:**
By just simply random pick songs to build playlists, we're trying to impose some restrictions on selecting songs. In this method, we decided to search by keywords(classic, cowboy ...) in each "reviewText" and then to randomly select songs to make five different categories music playlists.

Table 2: Search By Keyword Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B000X9O7IE	B000002W7W	B00006L852	B000WCDI5K	B000QPLFZ8
2	B000002KH3	B0015AEKGE	B005QJZ5FA	B0000026CZ	B006NO2WJ4
3	B000002KYU	B004BUFJS8	B00001OH7M	B006PZA7XY	B006NF68FC
4	B0000024IE	B005QJZ5FA	B0007TFI56	B005OH1IZ0	B00123D0CE
5	B004YMS6QU	B002REWLV8	B000058DXH	B00137MM8W	B00A70VWVE
6	B0013832A8	B00949YGM6	B002Q11RJO	B00GTZ6O2S	B00065BYAY

3) **Build TF-IDF Models:**

- **Usage of token weights:** In the field of Natural Language processing, each token is not of equal importance to classify a number of songs into a few groups. TF-IDF model is useful in this field, because we want to give different weights to different tokens in a song. If two songs are similar in geometry, we assume that they are on the same topic.
- **TF-IDF models:** Given the factors mentioned above, we need to build a model of how important tokens are based on the given keywords in a song. TF-IDF is a suitable model to distinguish two songs in different topics. In this model, more weight is given to a term

if it appears many times in a song, and less weight is given to a term if it appears in many documents. This is because some terms like "it", "is", "a" are not useful at all in terms of classifying songs.

- **Calulation of TF-IDF models:** Based on the TF-IDF models, we can calculate the weight of each term in a song. First of all, we need to build a vector space based on the tokens that appear in the songs. When we build the vector space, we need to preprocess the data. Specifically, what we have to do is consistent casing (map all characters to upper (or lower) case), unify or remove numbers (map all characters to upper (or lower) case), remove unnecessary spacing (duplicate white spaces) and word tokenization (split text into individual tokens (aka words)). After we have completed building the vector space, we are able to calculate the weights of each token based on the TF-IDF model. To narrow down the gap between the weights of different tokens, we have used log function instead of directly using the frequency of tokens. The formula used for calulation is

$$w(d,t) = log(1 + f(d,t)), iff(d,t) > 0; w(d,t) = 0, iff(d,t) <= 0$$

Given this formula, the weight of each token in a song can be calculated, thus we are able to get the vector of each song.

Table 3: TF-IDF Models Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B0026LWUCC	B000V66UV6	B000HBKCDC	B000EHQ80A	B0026EYVR6
2	B000V66H3M	B000SZDI42	B00BCYR8I0	B000VMYM5Q	B00H38MCFS
3	B00137RHT6	B000QNU0RY	B008A1W1SK	B00124HEFM	B00A3GQW76
4	B001EW8BAO	B002IEW504	B001NB6P3Q	B007XEHNUQ	B00BDNBZXY
5	B000W239UY	B00123HIPY	B00DYFCYSO	B00E9OGMPA	B001NCRRTG
6	B00171I5YO	B00B13OPB0	B0000025BA	B00GHJ6VME	B00005ICAW

4) **By Using *Also Bought* Attribute:**
When coming up with this solution, we were inspired by *Also Bought* attribute in Mete data. To be specific, there are mainly three steps in this method. Below we will illustrate this process by using the example of building **classic** music playlists. Then we do the same thing to make Cowboy, Dreamy, Rocks, Christmas playlists.

- **In reviews_Digital_Music dataset:** Firstly, we intend to find all rows that contain the keyword **classic** in *reviewText* attribute. Then we groupby this dataset by *asin*(which is the productID attribute) to find the mean of each song. By sorting their mean, we obtained the song that be rated highest that belong to the classic music.
- **In meta_Digital_Music dataset:** We write down the *asin* of the song we found, then we went back to the meta_Digital_ Music dataset to find the other songs related to this song by using *Also Bought* attribute in metadata. Then we write down the related songs' *asin*.
- **Back In reviews_Digital_Music dataset:** Sometimes, the number of related songs are exceed six. However, we're intended to make playlists consist of six songs. In order to create high-rating playlists, we used the *asin* in last part and then groupby each song's *asin* back in reviews_Digital_Music dataset to find songs with higher average ratings.

Table 4: By using *Also Bought* Attribute Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B0030G731A	B001NCINQW	B0050CJNJ2	B001KQICK8	B003WZ7QEI
2	B00137IL5U	B005XVBJHO	B002WIDRM6	B0064Y5ZBU	B000VRSXII
3	B001NB1M7A	B001NB1JA0	\	B0016OAM70	B001OGPVI0
4	B001NB6NO2	B00449PNXM	\	B0011Z0YR2	B001JQE9AG
5	B001NB8LV0	B0035QF0A6	\	B0043CKUAQ	B0016OAM70
6	B001NB1P7W	B0035QF02E	\	B0011Z1DPY	B000X6PY8E

3 Experimental Results

1) **Evaluate Random Pick Solution:**
In this part, we want to evaluate the average ratings of each playlists we built before to find out the quality of the playlists. We first find out the average rating of each song(i.e. find all rows in *reviews_Digital_Music* dataset that rating this song and then to calculate the average ratings of this song) in one specific playlist. After finding the average rating of each song in one playlist, we add them together to calculate the overall mean of this whole playlist to see whether this is a good quality playlist.

Table 5: Evaluate Random Pick Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B0011Z3DN4	B00136LUKO	B000005AM7	B0092YPJDI	B00000AGMC
2	B009D9L77Y	B008FOQH6Y	B008P5MLL8	B0002W4T4O	B003U7V8PG
3	B007DIQAXI	B008CWEPDG	B004H6QJVC	B000VT7AF8	B00123HPUC
4	B001AJFE5G	B00GD2L59O	B0027WNRNQ	B00005ICAW	B00014TQ7S
5	B00CAZOHDO	B001G93Z5Q	B00E9E4EC8	B00A0A5A6Y	B000W1MDAM
6	B007B6VOKQ	B00005A8MT	B0000026WD	B0006ZQ9BS	B008K9SG9K
Average rating	Classic	Cowboy	Dreamy	Rocks	Christmas
1	5.00	4.75	4.51	4.38	4.19
2	4.81	5.00	4.33	4.48	4.50
3	4.82	5.00	5.00	4.83	5.00
4	5.00	4.69	4.70	3.82	3.99
5	4.38	3.40	4.58	5.00	5.00
6	4.91	3.64	4.69	3.85	4.53
Overall Average	**4.82**	**4.41**	**4.64**	**4.39**	**4.53**

Since this method is randomly pick from our dataset, we decided to evaluate from another aspect to verify that each song's category, then to verify that whether this song belongs to the category in the table above. These songs do not all been provided their category in the dataset, so we fill this table by what we've found in dataset and what we can confirmed.

Table 6: Evaluate Random Pick By Category

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	\	\	Dance & Electronic	\	\
2	\	\	\	Pop Music	\
3	Rock	\	\	\	\
4	\	\	Dance Pop	Hardcore & Punk	\
5	Rock	\	Country Music	\	\
6	\	\	Blues	\	\

2) **Evaluate Search By Keyword Solution:**
We calculate the average rating of each song and each playlists by the same way explained above.

Table 7: Evaluate Search By Keyword Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B000X9O7IE	B000002W7W	B00006L852	B000WCDI5K	B000QPLFZ8
2	B000002KH3	B0015AEKGE	B005QJZ5FA	B0000026CZ	B006NO2WJ4
3	B000002KYU	B004BUFJS8	B00001OH7M	B006PZA7XY	B006NF68FC
4	B0000024IE	B005QJZ5FA	B0007TFI56	B005OH1IZ0	B00123D0CE
5	B004YMS6QU	B002REWLV8	B000058DXH	B00137MM8W	B00A70VWVE
6	B0013832A8	B00949YGM6	B002Q11RJO	B00GTZ6O2S	B00065BYAY
Average rating	Classic	Cowboy	Dreamy	Rocks	Christmas
1	5.00	4.43	4.75	4.29	5.00
2	4.75	4.50	4.52	3.67	5.00
3	4.87	5.00	4.37	5.00	5.00
4	4.76	4.52	4.53	5.00	4.80
5	5.00	3.71	3.92	5.00	5.00
6	4.61	5.00	5.00	4.19	4.30
Overall Average	**4.83**	**4.53**	**4.51**	**4.53**	**4.85**

3) **Evaluate Build TF-IDF Models Solution:**

Table 8: Evaluate TF-IDF Models Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B0026LWUCC	B000V66UV6	B000HBKCDC	B000EHQ80A	B0026EYVR6
2	B000V66H3M	B000SZDI42	B00BCYR8I0	B000VMYM5Q	B00H38MCFS
3	B00137RHT6	B000QNU0RY	B008A1W1SK	B00124HEFM	B00A3GQW76
4	B001EW8BAO	B002IEW504	B001NB6P3Q	B007XEHNUQ	B00BDNBZXY
5	B000W239UY	B00123HIPY	B00DYFCYSO	B00E9OGMPA	B001NCRRTG
6	B00171I5YO	B00B13OPB0	B0000025BA	B00GHJ6VME	B00005ICAW
Average rating	Classic	Cowboy	Dreamy	Rocks	Christmas
1	5.00	4.00	4.55	5.00	4.96
2	5.00	5.00	4.89	5.00	5.00
3	5.00	5.00	5.00	4.00	4.77
4	5.00	4.60	4.50	5.00	4.58
5	4.87	5.00	4.46	4.70	4.45
6	4.88	4.91	4.82	4.62	3.82
Overall Average	**4.96**	**4.75**	**4.70**	**4.72**	**4.60**

4) **Evaluate By Using *Also Bought* Attribute:**

Table 9: By using *Also Bought* Attribute Resulting Playlists

# of ProductID	Classic	Cowboy	Dreamy	Rocks	Christmas
1	B0030G731A	B001NCINQW	B0050CJNJ2	B001KQICK8	B003WZ7QEI
2	B00137IL5U	B005XVBJHO	B002WIDRM6	B0064Y5ZBU	B000VRSXII
3	B001NB1M7A	B001NB1JA0	\	B0016OAM70	B001OGPVI0
4	B001NB6NO2	B00449PNXM	\	B0011Z0YR2	B001JQE9AG
5	B001NB8LV0	B0035QF0A6	\	B0043CKUAQ	B0016OAM70
6	B001NB1P7W	B0035QF02E	\	B0011Z1DPY	B000X6PY8E
Average rating	Classic	Cowboy	Dreamy	Rocks	Christmas
1	5.00	5.00	5.00	5.00	5.00
2	5.00	5.00	4.56	4.75	5.00
3	4.91	5.00	\	4.73	5.00
4	4.89	4.94	\	4.55	5.00
5	4.57	4.38	\	4.35	4.75
6	4.33	4.00	\	4.25	4.50
Overall Average	**4.78**	**4.72**	**4.78**	**4.61**	**4.88**

4 Discussion

Figure1 shows the rating comparison of four methods that introduced before. We used the evaluation results above for each playlists in each method to see how they compare with each other.

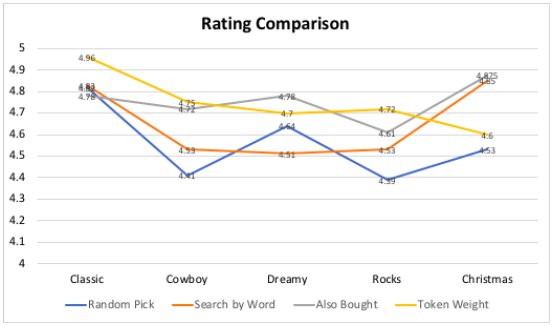

Figure 1: Rating Comparison

Figure 2: Overall Average Rating Comparison

After analyzing Figure1, we calculated the whole average rating of each method to see which method is the best(see Figure2). For example, For Token Weight Method, we add the ratings for classic, cowboy, dreamy, rocks and christmas playlists together, then divided by 5 to see how this method work overall.

- As mentioned in Evaluation part, In *Random Pick* method, in addition to calculate the average rating, we need to see whether the selected songs belong to the categories we chosen in this project. We can see that not only the overall rating for this method is the lowest, but also these randomly picked songs are not all belong to the categories we divided. Therefore, we label this method as a bad one.

- We can see that *Also Bought* method and *Token Weight* method has very close overall average rating in Figure2. However, by definition, the meaning of "also bought" is that people who buy this song also buying some other songs. But these other songs might not belong to the same music genre/style. People may like different kinds of music at the same time. Therefore, we cannot confirm that the 'also bought' directly indicated that those songs belong to the same music genre/category. Nevertheless, there's high possibility that those songs are quite similar.

- We find that the *Token Weight* should be the best way in our project to build music playlists. By implementing this model, we find the song that is closely to the music genre we want to find. In addition, they've been highly rated. To conclude, 100% of them have the average rating which is greater that 4.6. So we label this method as the best one.

- Although *Search By Keyword* method does not have the high overall average rating which is shown in Figure2. However, by limiting with the keyword, those songs are closely related to the category they belong to.

- Some other interesting finding: No matter in which method, the average ratings for *Classic* category are high.

5 Conclusion

- **Lessons Learned:**
  From starting to think about what project ideas we are interested in, to implementing our project step by step, we have learned a lot from this process. When we first decided to make music playlists, we thought it was a great and novel idea. Not only did the data processing and evaluation process, we could actually produce a thing. But when it was implemented, it was a bit helpless at first, because there didn't seem to be many existing library data processing tools suitable for our project, and we found it difficult to advance our project. After asking the professor's opinion after each presentation, we improved our project step by step. We decided to customize some meaningful and different types methods to analyze the data and make music playlists based on the perspective of rule-based machine learning. Personally, we think that our biggest gain from this project is to keep trying and improve the rule-based machine learning methods. This process is not as simple as we thought. It has a lot of freedom in customization, but it also has a lot of difficulty. When we came up with a good idea, it may not be implemented successfully. Therefore, we need to optimize our results under realizable conditions. In order to make our project more scientific, we also focus on using the TF-IDF model to build playlists. We think it is a good combination to use both existing models and to use rule-based machine learning.

- **Future Work:**
  To be continued, We need to learn more skills to improve our rule-based machine learning methods in this project. Sometimes, we have some better ideas to optimize our methods. Nevertheless, due to the limitation of our current skills and knowledge, we may not realize all of them. In the future, we want to to have some deep learning in machine learning and to have some deep understanding of those existing models to improve this project. For example, we may try to learn how to use Weka, which is an opening source for data mining and machine learning. Instead of using rule-based machine learning, KNN algorithm can be applied for classification. Further, we can calculate the sum of squared estimate of errors(SSE) to measure the accuracy. In addition, there are some wonderful recommendation system in our market that is related to music playlists. We are planning to learn something about that as well. In a word, this is a meaningful project and we've paid effort to it due to our current ability, but there are more things need to be done in the future.

- **Demo Demonstration:**
  After finding the best quality music playlists, we built classic and cowboy playlists using some UI technique for demonstration in the next page.

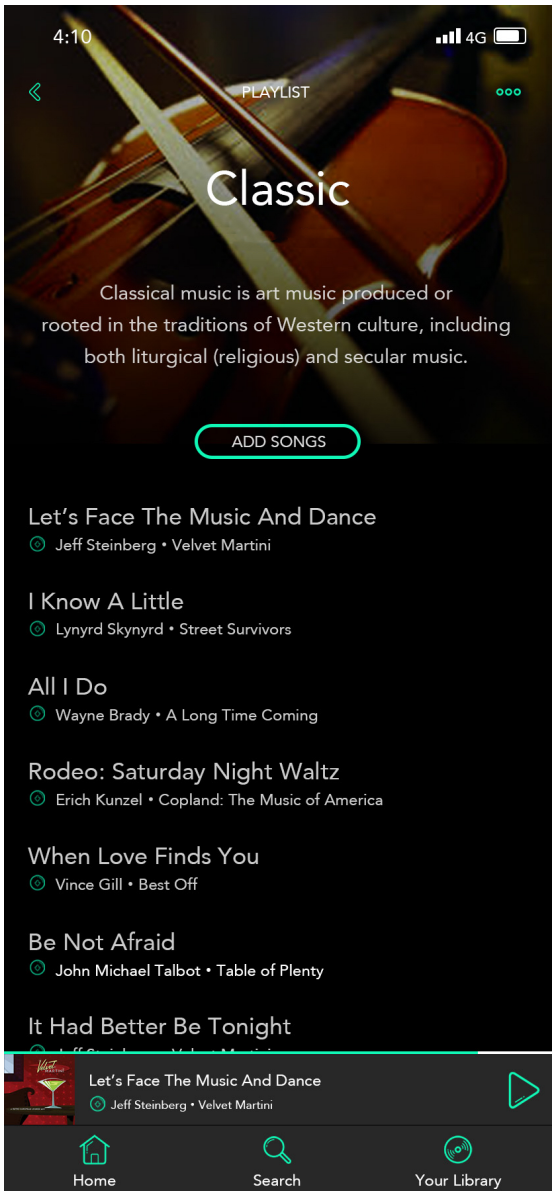

Figure 3: Classic Music Playlist

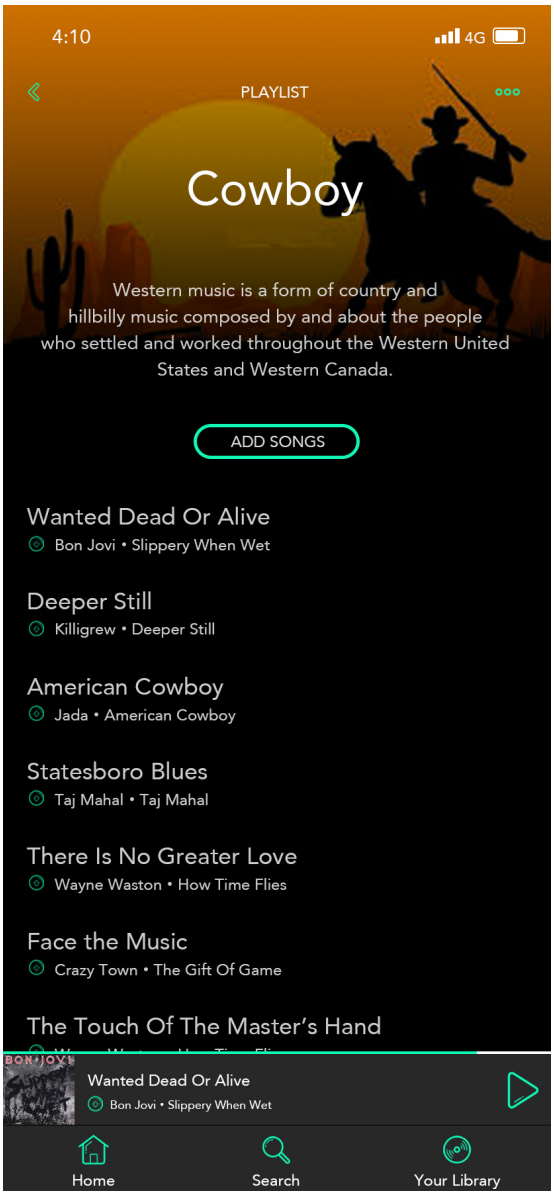

Figure 4: Cowboy Music Playlist